INTEGRATED FRAMEWORK FOR AIRCRAFT DESIGN AND ASSEMBLY TRADEOFFS

A Dissertation
Presented to
The Academic Faculty

By

Dat Huynh

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
Daniel Guggenheim School of Aerospace Engineering
Aerospace Systems Design Laboratory

Georgia Institute of Technology

December 2023

INTEGRATED FRAMEWORK FOR AIRCRAFT DESIGN AND ASSEMBLY TRADEOFFS

Thesis committee:

Prof. Dimitri Mavris, Chair
School of Aerospace Engineering
Georgia Institute of Technology

Prof. Shreyes Melkote
School of Mechanical Engineering
Georgia Institute of Technology

Prof. Daniel Schrage
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Adam Cox
School of Aerospace Engineering
Georgia Institute of Technology

Prof. Graeme Kennedy
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Steven Wanthal
Advanced Composites Research
The Boeing Company

Date approved: July 28, 2023

No Excuses.

Robert H. Liebeck

To my newborn and the one who came before you

ACKNOWLEDGMENTS

I would not have been able to get to where I am without significant help from others. I first want to thank my thesis committee members, Professor Dimitri Mavris, Professor Daniel Schrage, Professor Graeme Kennedy, Professor Shreyes Melkote, Dr. Steven Wanthal, and Dr. Adam Cox. Thank you, Professor Melkote and Dr. Wanthal, for being available to help me with clarifying manufacturing details and assumptions to make sure they are realistic. Thank you, Professor Schrage, for your insistence on selecting the correct model set, without which I may well have tried to model an aircraft I didn't have sufficient data for. Thank you, Professor Kennedy, for lending me your expertise on structural modeling and optimization. Thank you, Dr. Cox, for always being present to give me feedback on all of my ideas through this long and arduous process. And I must thank Professor Dimitri Mavris, my advisor, in particular for allowing me to be a part of the Aerospace Systems Design Lab (ASDL). It is the only lab I know of with enough research breadth for it to be possible to tackle as multidisciplinary a topic as mine. I am beyond grateful for your willingness to make so many accommodations for me, from giving me sponsored research work that matched my ideas to enabling me to finish my thesis remotely in California. Thank you as well for the boot camp experience that taught me so many fundamental skills as a researcher.

I could not have started or finished this journey without my colleagues at Boeing who helped me make this all possible. In particular I am deeply indebted to Mitch Petervary and Bob Liebeck for guiding me through my internships at Boeing, giving me initial feedback on the ideas that would turn into this dissertation, and helping me along in my student and professional career. I would like to extend my sincere thanks as well to Dan Humfeld, for his willingness to explain manufacturing and structural concepts, and to Krishna Hoffman, for being a very good team leader and allowing me to work on tasks that helped me acquire the required manufacturing and structural analysis skills for this thesis. I would like to

acknowledge Kim Linton too, who set off the creative spark that led to my examining of assembly sequence planning and coming up with the topic for this work.

I must thank the research engineer who led me down the manufacturing rabbit hole and kept me there, Dr. Olivia Fischer, without whom I would not have been able to work on the LCAAT project and even known manufacturing was a discipline. And thank you, Dr. Jason Corman, for being available every week over the summer of 2022 to answer questions about how to use the RADE framework. I would also be remiss in not mentioning the Simio company, which graciously allowed me to use the full student version of their software to finish off the needed simulations for my case study.

I have not forgotten all my friends in Weber 106, aka, The Dungeon. Thank you Steve, Mackenzie, Mac, Nathan, Taylor, Alexandra, Jimmy, Nikita, Maddy, and Akshay. You guys made the boring days entertaining, perhaps even too much so sometimes, and were always willing to talk and give advice. I am extremely grateful to those last 5 in particular, who helped me get housing when I was nearly homeless, made me feel connected and still a part of the lab while I was in California, and provided voices to assure me I wasn't alone in the darkest days of the pandemic.

And most of all, I must thank my family: my parents for devoting their lives to my siblings and me, my siblings Dzuy and Phuong and brother-in-law Chris for putting up with me, and my wife Professor Jacqueline Huynh, who was with me every step of the way, through the great ups and the terrible downs, listening to me on my long winded rants about my topic and everything else. Dear, you were the one who said I should get a Ph. D. and then insisted I visit Georgia Tech when they extended an offer during grad school applications when I would have otherwise blown them off due to their less-than-stellar application process. Without you, none of this would have ever happened.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

ACO Ant Colony Optimization

AEM Assembly Evaluation Method

ALB Assembly Line Balancing

ALBP Assembly Line Balancing Problem

APUC Average Per Unit Cost

ASDL Aerospace Systems Design Laboratory

ASP Assembly Sequence Planning

ATL Automated Tape Layup

CAD Computer-Aided Design/Drafting

CFD Computational Fluid Dynamics

CNC Computer Numerical Control

DES Discrete Event Simulation

DFA Design for Assembly

DFM Design for Manufacturing

DFMA Design for Manufacturing and Assembly

FAL Final Assembly Line

FEA Finite Element Analysis

FLOPS Flight Optimization System

GA Genetic Algorithm

GALBP General Assembly Line Balancing Problem

HLU Hand Layup

IML Internal Mold Line

IPLE Integrated Product Lifecycle Engineering

IPPD Integrated Product Process Development

KBS Knowledge-Based System

LFL Landing Field Length

MInD Manufacturing Influenced Design

MInDPRO Manufacturing Influenced Design Production Optimization

MODAPTS Modular Arrangement of Predetermined Time Standards

MOST Maynard Operation Sequence Technique

MTM Methods-Time Measurement

NDI Non-Destructive Inspection

OML Outer Mold Line

PLE Product Lifecycle Engineering

PLM Product Lifecycle Management

PSO Particle Swarm Optimization

RADE Rapid Airframe Design Environment

ROI Return on Investment

SALBP Simple Assembly Line Balancing Problem

SRI Stitched Resin Infusion

TIAPG Tentatively Inherited Associated Parent's Genes

TIOPG Tentatively Inherited Opposite Parent's Genes

TOFL Takeoff Field Length

TOGW Takeoff Gross Weight

TSALBP Time and Space Assembly Line Balancing Problem

VARTM Vacuum Assisted Resin Transfer Molding

SUMMARY

Aircraft passenger traffic is expected to increase and lead to demand for 40,000 new aircraft by 2040. Aircraft production rates have been rising to meet this demand, but delivery backlogs are growing at even faster rates. Large backlogs can lead to missed deliveries, canceled orders, and traffic congestion due to too few planes for too many passengers. This comes at a time when the two primary aircraft manufacturers, Boeing and Airbus, are competing for dominance in a market that a third competitor, Comac, is poised to enter as well. Increased production rates to better meet customer demand would thus also allow one of them to gain the edge over the others. Aircraft production rates must be increased and done so without inducing enormous costs to meet passenger demand and to stay competitive.

Changes to aircraft assembly, which constitutes up to 50% of total production time and up to 30% of total production cost, during the design process can address this. Current Design for Assembly methods addressing assembly changes during the design process range from Product Lifecycle Management techniques to various methods in Systems Engineering and have been used to great success. However, few such methods consider aircraft design in their analysis, which would enable further tradeoff capabilities and greater production rate and cost improvements. Those methods that do explicitly incorporate aircraft design analysis alongside the assembly analysis insufficiently consider several key assembly aspects such as assembly sequence planning (ASP) and more detailed assembly line balancing (ALB), which can be used to optimize the assembly line.

This work establishes a better connection between the aircraft design and assembly disciplines and joins them by accounting for geometry and material factors common to both using ASP and ALB. First, the correct analysis fidelity for the aircraft design process and ASP's geometric reasoning process is determined to allow geometry data to easily flow between the two, linking them. Then, the ASP and ALB analyses are combined and augmented to account for the novel materials traded during aircraft design and the manufac-

turing processes used to make them. Afterwards, the most promising assembly sequences are optimized for using metrics representative of both ASP and ALB so that sub-optimal assembly sequences are not line balanced, reducing the overall problem size to explore the large design space more efficiently. From all this an integrated aircraft design and assembly framework is made that strives to obtain higher production rates, lower costs, and better tradeoffs by leveraging the additional feedback loops produced via consideration of variables common to both disciplines. Finally, this framework is tested and compared with a state-of-the-art framework on a representative aircraft's wingbox and its production system.

The developed framework demonstrates it is able to obtain significantly higher throughput and lower cost values by simultaneously: sizing the aircraft to meet its performance requirements; accounting for the aircraft's geometry via ASP determining assemblability and ALB determining the consequent task time, cost, and space requirements; incorporating the aircraft's material system via usage of specialized sequences in ASP and identification of optimal line balances in ALB given the material's manufacturing process and its subsequent resource requirements; and flowing all this manufacturing information back upstream to maximize the aircraft's manufacturability during its sizing. The developed framework is thus able to make tradeoffs such as what size the aircraft should be for a given performance, throughput, and cost requirement, what the maximum production rate is given a design, material, manufacturing process, and spatial constraint, and what costs are incurred given a desired production rate. This provides the designer with a greater understanding of the problem and its constraints and allows them to see what factors can help them increase production rate as well as what the associated costs are.

CHAPTER 1
INTRODUCTION

1.1 Motivation: Need for Higher Production Rates

Over the next two decades, it is projected that airline passenger traffic will increase by 4.6% annually [1]. To meet this demand, the current in-service aircraft fleet will have to more than double and expand its size by 3.4% annually [1]. Accounting for the replacement of older aircraft in the coming years, this will result in the need for about 40,000 new aircraft by 2038, worth about US $6.8 trillion [1]. Due to the large and ever-increasing demand Boeing and Airbus, the two primary manufacturers of commercial aircraft, have been constantly increasing their aircraft production rate for the last few decades, as can be seen in Figure 1.1. However, the projected demand in aircraft translates to an average yearly production rate of 2,000 aircraft: as can be seen in Figure 1.1, Boeing and Airbus are only able to total to roughly 1,500 aircraft per year. The current demand has already eclipsed the production rate increases enough to create an enormous backlog. Future demand will almost certainly completely overwhelm it. Aircraft production rates must be further increased.

Problems regarding insufficient production rates are not limited to just the ever-increasing backlog. Airbus and Boeing have each been vying to control a larger portion of the aircraft manufacturing market than the other for the last few decades and are currently neck and neck, as shown in Figure 1.2 [5]. This equilibrium of sorts is liable to end not only because of the increased throughput demands of the future changing the equilibrium conditions but also because another aircraft manufacturing company, China's Comac, is entering the market to compete with them [6]. With three companies in this market now, the competition will become even more cutthroat than it already is and each company will need to do everything it is capable of to stand out above the others. It will not be sufficient for these

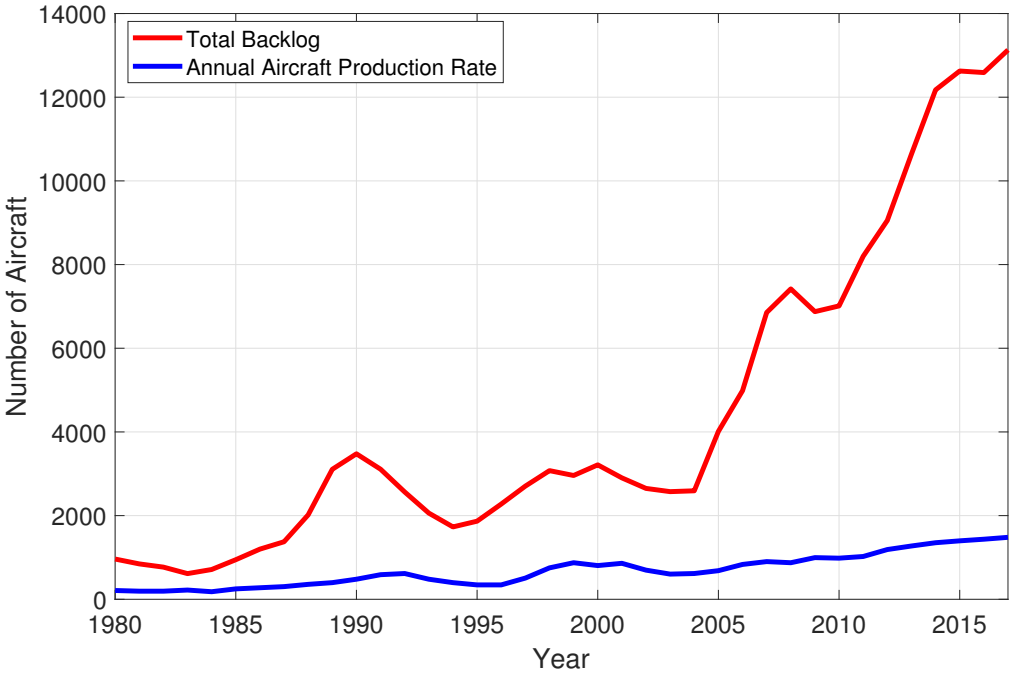


Figure 1.1: Combined Airbus and Boeing Annual Production Rate and Backlog from 1980-2017, Adapted from Hexcel Corporation, Boeing, Airbus [2, 3, 4]

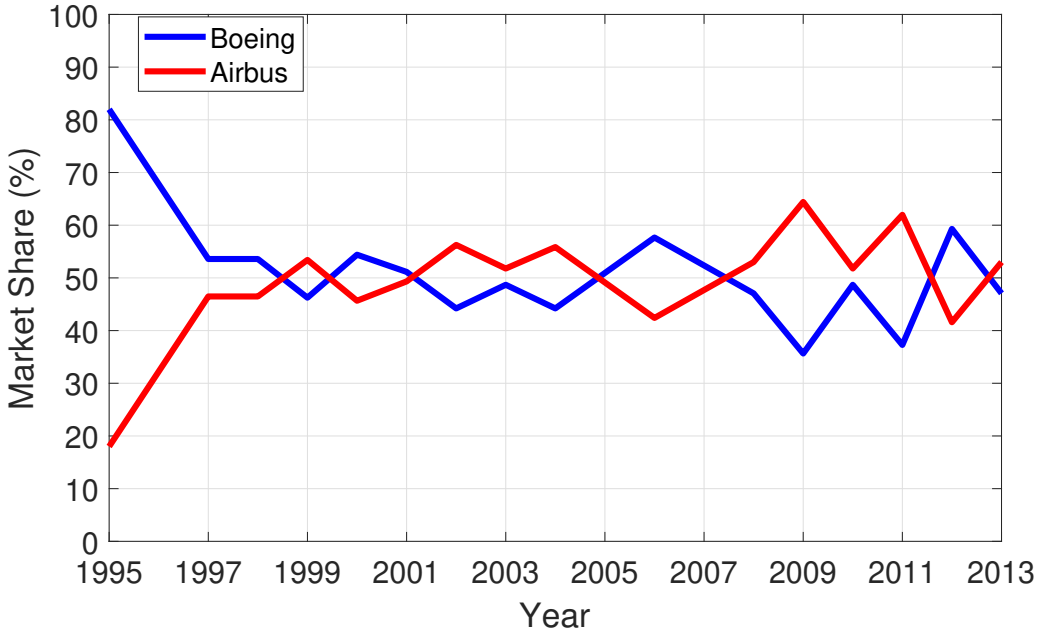


Figure 1.2: Boeing and Airbus World Market Share, Adapted from CAPA Centre for Aviation [5]

companies to together just barely meet future throughput demands if they want to maintain or increase market share; they will each need large jumps in throughput capability to have the edge and convince customers to not go to their competitors for airplanes.

To do otherwise in Boeing and Airbus's case is to cede market share to the newcomer or the other competitor and in Comac's case be completely overshadowed and possibly driven out by the established companies. Additionally, the current economic climate's focus on affordability in all aircraft programs no longer allows for such capabilities to come "at any cost" unlike in the past [7]. Boeing and Airbus themselves have recently basically acknowledged all this, with Airbus creating their Wings of Tomorrow program to industrialize their wing-making abilities [8] and Boeing working with NASA on the High Rate Composite Aircraft Manufacturing (HiCAM) to maximize their wing production rates [9]. Aircraft production rates must thus be further increased and increased cost effectively, both to not have the customers balk at the price tag and to not accidentally go bankrupt in the process.

This all leads to this work's motivating question:

Motivating Question *How can aircraft production rates be further increased cost-effectively?*

Despite this, current methods to increase production rate are unlikely to be sufficient in the future. As an example, the majority of future aircraft demand will be for single aisle aircraft, with a projected demand of about 32,400 aircraft worth US $3.75 trillion [1]. This means that roughly 1,620 single aisle aircraft must be produced per year, or 135 per month. Boeing and Airbus currently have a maximum monthly single aisle aircraft production rate of 63 and 57, respectively [10]. However, the maximum production rates for both companies took quite some time to attain, Boeing requiring 6 years to increase from 42 to 57 [10, 11] and Airbus projecting 3 years to increase from 55 to 63 [12, 13], with neither so far being able to sustain those rates. Additionally, the large amount of time needed

to ramp up production rates only serves to further compound both companies' backlog issues. Even if the two companies could, together, produce 135 single aisle aircraft per year, assuming future order trends are similar to the present, the backlog time will still stay at a dismal 9 years, which is undesirable for customers. Other options include significant infrastructure investments or reallocation of existing infrastructure. However, the former is prohibitively expensive while the latter would reduce production rates for other aircraft types, only further amplifying the problem. Production rates thus need to increase and do so cost effectively, but current industry practices are not practical for achieving that goal.

A promising avenue to address this problem is to alter either the aircraft's design or its manufacturing system's design, or both, to be as amenable to high throughputs as possible. Design has many meanings in different fields and disciplines but is generally defined as the blueprints for making something with a purpose or as the process of making those blueprints. In this instance, for aircraft design it can refer to the configuration, or the physical features that impact performance, of the aircraft such as number of engines, fuselage length, and wing sweep. Likewise, for manufacturing system design it can refer to how the factory is laid out or how workers are distributed among different workstations. An aircraft or manufacturing system design based on large production rates would not need reallocation of or additional infrastructure and would be significantly easier to ramp up if desired, eliminating the issues with current industry practices.

Unfortunately, perhaps even more so than the aforementioned practices, altering the design during production is extremely expensive; all such design decisions have already been essentially set in stone and are dependent on one another. Making changes to one system will require a chain of changes to other systems. Therefore, anything more than a handful of such changes quickly snowballs to trying to alter the entire aircraft and its production systems. This is best summed up in Figure 1.3. During the typical aircraft design process, knowledge about the design is obtained by sacrificing design freedom, the ability to make further changes to it, throughout the design phases leading up to production.

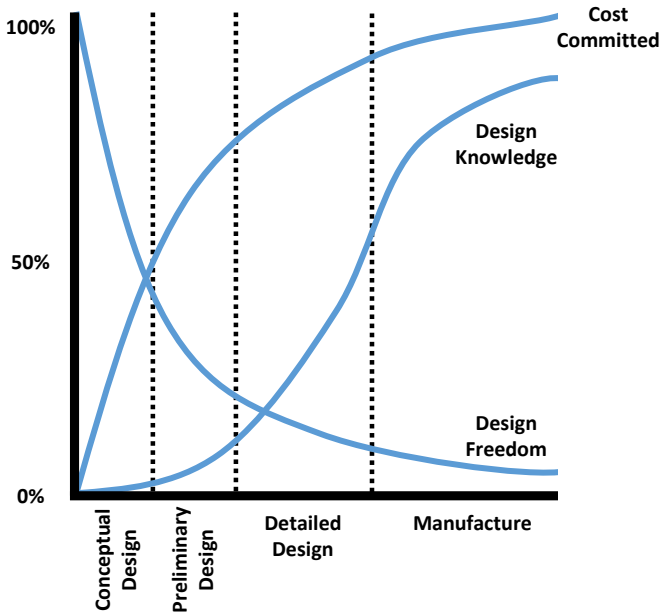


Figure 1.3: Cost Commitment and Design Knowledge vs Design Freedom in Different Aircraft Design Stages, Adapted from Mavris, et al. [7]

This locks in the aircraft's cost as the design freedom vanishes, leading to almost no feasible and affordable changes after production has started because most decisions have already been made and the costs set. Design changes during production are therefore out of the question and the only recourse is to incorporate high throughput, low cost considerations into the aircraft's design from the beginning and solve the problem before it ever appears. In other words, it is not feasible to try to make current aircraft meet future throughput requirements–new aircraft must be designed with those requirements already in mind. This entails less focus on production and post-production and a higher focus on pre-production, design, and planning: benefits will be measured less on physically implemented changes and more on improvements to models based on and close to reality.

The most well-known methods to try to improve production rates and reduce cost by making changes to the design are the Design for Manufacturing (DFM) and Design for Assembly (DFA) methods. DFA is a formal analysis procedure for determining the ability

and ease with which a design can be assembled [14]. DFM is a set of methodologies whose goal is to reduce the cost of fabricating the parts that make up a product [15, 16]. They were both created as part of a bid to be able to consider design choices in all stages of a product's development, a concept known as "concurrent engineering" [15]. The aim was to not have designers throwing their designs "over the wall" to manufacturers and instead have the two communicate and consider producibility and assemblability up front to reduce cost, improve quality, and increase manufacturability. According to Leaney, both DFA and DFM revolve around lowering costs by "design[ing] right [the] first time" and addressing fabrication and assembly issues such that "wrong" products are never made in the first place, instead of trying to fix defective products [17]. DFA does this and incorporates design with assembly by designing the product in such a way as to simplify the act of assembling, primarily by reducing part count and making assembly operations easier to perform [16]. DFM does this and incorporates design with fabrication via multiple avenues. According to Holt, this includes leveraging "general guidelines for features to include or avoid," using manufacturability handbooks, and having Knowledge-Based System (KBS) provide feedback on the product during its Computer-Aided Design/Drafting (CAD) implementation [15].

Both DFA and DFM have proven to be capable methods over the last few decades with substantial production time and cost decreases. DFM has been used to replace the Apache helicopter's sheet metal angles and extruded stiffeners with machined parts, saving $43,000 per item. Similarly, DFA has been used to reduce the number of parts in Bombardier's CRJ-200's engine nacelle torque box, reducing its recurring costs by $450,000 and assembly time by 30 hours [16]. They are so effective Boothroyd and Dewhurst, in fact, combined the two of them together into one tool and called it Design for Manufacturing and Assembly (DFMA). This leads to the following observation:

Observation 1: In the past, considering assembly and manufacturing/ fabrication with design has yielded large time and cost improvements.

However, in the scope of this work, there is only enough time to focus on one of the two methods: DFA and the discipline of assembly, or DFM and the discipline of fabrication. Assembly was chosen as the more promising of the two because, as both Leaney and Holt noted, DFM is too focused on the component-level aspects of the product whereas DFA, and assembly as a discipline in general, is more holistic and looks at the entire product [15, 17]. Looking at assembly allows for more cross-system interactions to be examined and thus provides more high level and impactful improvement opportunities, which is why it is preferred. And indeed, the assembly by itself is still very significant: assembly traditionally takes up to 50% of the total production time [18] and 10 to 60% of the total production cost of a product [19]. Changing how the aircraft is assembled and making other decisions based on its assembly can therefore have a large difference in how long it takes to produce it, leading to Observation 2:

Observation 2: Assembly has a significant impact on production time and cost.

It is due to this that DFA and the assembly discipline will be examined more closely to answer the question of whether current DFA methods are able to address the problem. The two prior observations combined with this question leads to a more refined version of the motivating question:

Refined Motivating Question *Can current methods combining assembly and design cost-effectively increase current aircraft throughput?*

1.2 Current Methods to Address Higher Production Rate Needs and Their Insufficiencies

The original DFA methods look at the design of the product's assemblies and individual parts primarily by limiting the product's part count without adversely affecting its functionality, increasing assemblability. This is done in a variety of ways, with the most common

being to have the designer answer a series of questions regarding the complexity of the part movements during assembly, the product's theoretical minimum number of parts, and the materials the parts are made from [17, 16, 20]. They resulted in improvements such as those mentioned in the previous section as well as allowing the McDonnell Douglas F/A-18 Hornet to have 45% fewer parts and weigh 1000 lbs. less as it was being redesigned into the Super Hornet [16].

However, while the original DFA methods are able to incorporate the product's design into its assembly before production starts, they do so in the detail design phase from Figure 1.3. Most of the design freedom and cost has still already been locked in, making the original DFA methods insufficiently able to provide a large throughput increase and cost decrease.

Conceptual design DFA methods were developed in response to the original's shortcomings. True to their name, conceptual design DFA methods operate in in the conceptual design phase and, instead of examining the details on the product's parts that were only available in detail design, look at modules related to the product's functions. These functional modules are available as soon as the product's functions are defined and are used as a proxy for the parts themselves, with the goal being to minimize the number of said functional modules or use them to generate assemblability scores [21, 22, 23]. This has allowed for larger assemblies to be optimized considering more systems, with such examples as optimizing a fuselage cabin component to have higher degrees of preassembly to reduce assembly time [24].

But while conceptual design DFA methods work in the conceptual design phase, they specifically only work in conceptual assembly design; assembly design and aircraft design are distinct and conceptual design DFA methods operate in detail aircraft design, just like the original DFA methods. This severely limits the degree of throughput improvements possible because major aircraft configuration trades still cannot be made. Additionally, being carried out in the conceptual assembly phase generally prevents conceptual design

DFA methods from having enough detail to be able to estimate production rates. These two flaws cause the conceptual design DFA methods to also be insufficient.

The current state of the art, most advanced methods are based on the concept of DFA but do not explicitly use it in the same way as the original and conceptual DFA methods. This class of advanced methods includes techniques such as product life cycle management and systems engineering that look at all the different systems of the product. They examine both the conceptual and preliminary design phases and their hallmarks are the traceability of all the design decisions and the free flow of information between the different design phases. With this they can make trades such as figuring out the best number of stations for an aircraft final assembly line, verifying which product features incurred the most cost via its resource usage, and seeing which sequences and line balances yield the highest production rate for a light aircraft [25, 26, 27].

Despite this these advanced methods still prove insufficient because, while they are able to consider conceptual and preliminary assembly design, they still do not incorporate aircraft design or make any trades on the aircraft's configuration. Changing the aircraft's geometry, the materials it is made out of, and the tolerances with which it is built has large effects on the aircraft's performance. These aspects can then be flowed down to also impact assembly results such as production rate and cost. Unfortunately this is not done, missing many opportunities to use as many tradeoffs as possible to maximize throughput cost-effectively.

In response to this, advanced methods that do incorporate aircraft design were developed. Contrary to previous methods whose primary focus was on assembly and then tried to branch out to other disciplines to implement the concurrent engineering concept, these methods focused on aircraft design and then branched out to assembly. As a result of this they are able to change the aircraft's planform, overall configuration, structural materials, and manufacturing process to try to optimize throughput and cost while at the same time meeting the necessary aircraft performance requirements [28, 29].

However, the initial focus on aircraft design and then branching out to the assembly discipline afterwards came at a price. These aircraft design-considering advanced methods do not account for as many assembly-based analyses nor in as much detail as the other class of advanced methods. They thus also miss out on many opportunities to utilize the maximum number of throughput improving tradeoffs, just that the tradeoffs missed are now from the assembly discipline. From this, it can be seen that no methods exist that are able to fully incorporate early assembly design with early aircraft design to have as high of a chance as possible of meeting future production rate demands in a cost-effective way.

1.3 Overarching Research Question and Summary

Because there are currently no methods in the literature that sufficiently address the problem of future high throughput demands in an affordable context, the following overarching research question is posed that is the focus of this work:

Overarching Research Question *How can early assembly design be better integrated with early aircraft design to increase production rates and reduce cost?*

The rest of this work is organized as follows. Chapter 2 goes over the literature review leading into the overarching research question in greater detail to gain additional context and better set up how the problem could be tackled. The overarching hypothesis, which is based on using assembly sequence planning and line balancing to better integrate the two disciplines, is developed and then presented.

In Chapter 3 the literature around assembly sequence planning and line balancing is reviewed to gain a better understanding of how they can be implemented to prove the overarching hypothesis. This results in the identification of various implementation issues for each analysis type, which must be addressed to actually carry out the framework being proposed by this work.

Chapter 4 focuses on a fidelity issue regarding the implementation of the assembly se-

quence planning. The correct fidelity for the input data to be used in the assembly sequence planning must first be identified because what can be outputted by aircraft design may not necessarily match what is needed for assembly sequence planning. Afterwards, the correct fidelity of the geometric reasoning techniques used by assembly sequence planning to extract the geometric information must be identified. This is because the techniques must be of the correct fidelity to not take too long to carry out and properly match with the fidelity of the input data as well. The end result is that the proper fidelity for geometric reasoning input data and techniques is arrived at.

Chapter 5 looks at how to implement a line balancing approach that incorporates assembly sequence planning as well as multiple problem types identified as necessary in Chapter 3. These problem types are integrated along with assembly sequencing to produce a multi-objective framework able to look at different assembly sequences, spatial constraints, and manufacturing processes to produce a Pareto front of optimal production rates and costs that a designer can choose from. An experiment is performed where this framework is compared with the most similar and capable approach in the literature both to demonstrate it has more and superior capabilities and to observe the types of tradeoffs it can provide.

Chapter 6 is dedicated to addressing the combinatorial expense of trying to perform both assembly sequencing and line balancing in a framework with aircraft design. Conventional techniques to accelerate the solving of the integrated NP-Hard problems are first examined such as increasing computational speed and using surrogate models. A method revolving around reducing the problem size to solve it more efficiently is then discussed. This involves trying to find the ideal assembly sequence and aircraft design using preliminary metrics before assembly line balancing is performed. This eliminates the extra effort and time spent in performing line balancing on a sub-optimal sequence and aircraft design. An experiment is afterwards carried out to determine how feasible this is, the quality of the results in the reduced-problem size approach versus the full size one, and how much time is saved.

Chapter 7 ties everything together into one integrated framework to fully answer the overarching research question and prove that all of the discussed factors, combined, can yield improved throughput and cost estimations as compared to current works in literature. This is done via comparison with another method in the case study presented in this chapter, which is on the analysis of an aircraft wingbox factory. Other advantages with using the proposed framework as opposed to current ones are discussed as well.

At the end of each chapter, save for Chapter 7 and the conclusion, is a logic diagram summarizing the main points of that chapter. The roadmap of this entire work and the combined summary of all the end-of-chapter logic diagrams is presented in Figure 1.4.

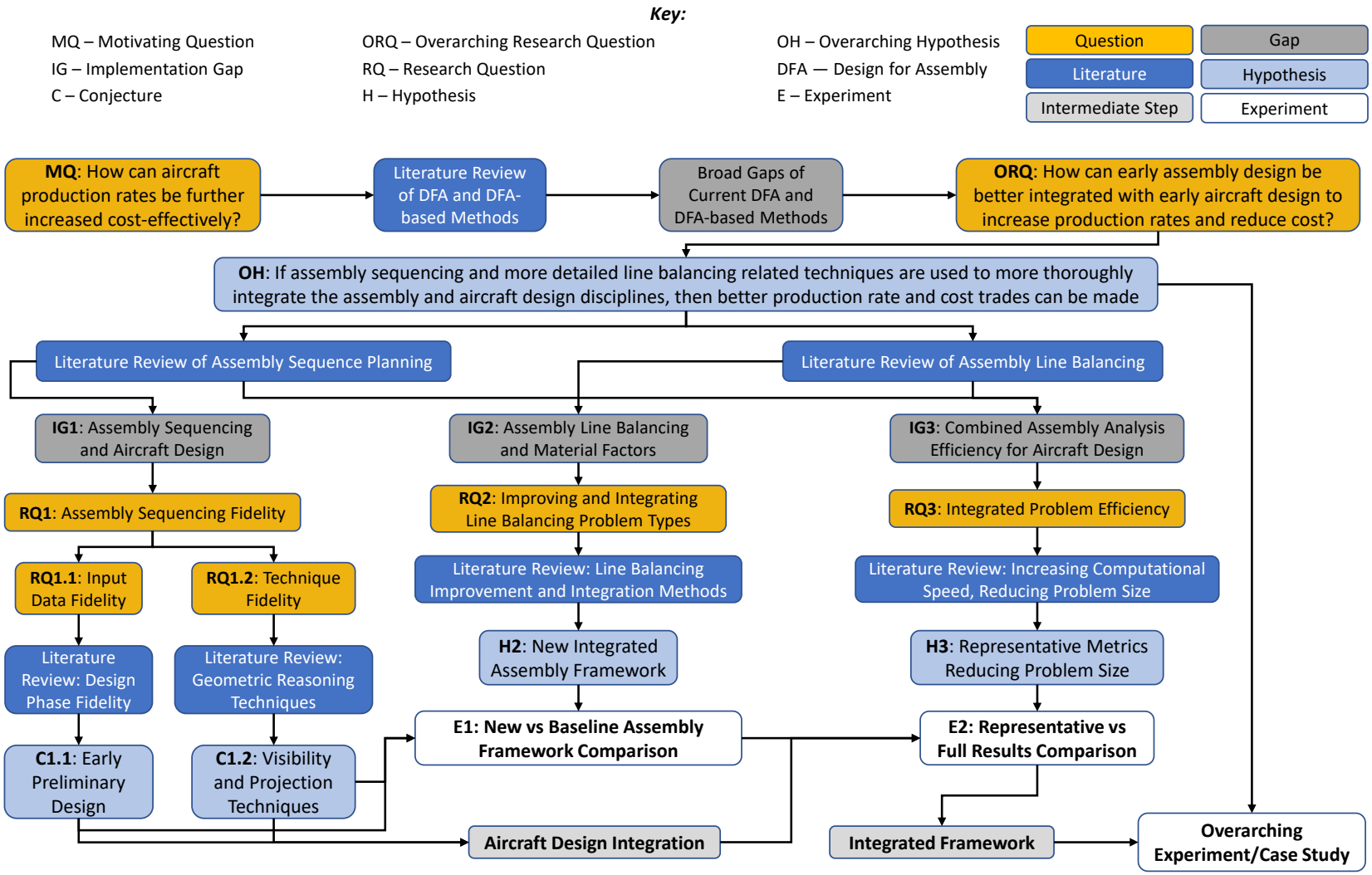


Figure 1.4: Roadmap of This Entire Work

CHAPTER 2
INTEGRATION OF AIRCRAFT DESIGN WITH FABRICATION AND ASSEMBLY

The literature review leading up to the overarching research question in the previous chapter was kept short for brevity. It summarized the general gaps in current methods but did not go into the intricacies surrounding them. These intricacies are needed to develop a framework that is able to precisely fill in those gaps and appropriately address the overarching research question. The first part of this chapter tackles this by going into more detail when looking at currently available methods, starting with the DFA ones, and then moving on to further define the problem.

But before going any further, the terms "manufacturing" and "assembly" that are so commonly used must first be better defined because there is a large amount of ambiguity around them, with different disciplines using the same terms to mean different things. Boothroyd and Dewhurst define manufacturing as the "[fabrication] of the individual component parts of a product or assembly" and assembly as "the addition or joining of parts to form [a] complete product" [16]. In the context of aircraft, this definition of manufacturing involves the process of creating the individual parts and components using the appropriate raw materials, with a component being defined as "a basic part where no assembly operation occurs" [14]. Similarly, assembly in the context of aircraft consists of putting together components like wing spars and ribs to form assemblies like wing boxes. It also consists of combining separate assemblies like wing boxes and fuselages to form larger assemblies like the entire airplane, this being called the final assembly line. Boothroyd and Dewhurst and the majority of those in the more focused assembly discipline thus consider assembly and manufacturing distinct. However, the majority of other disciplines use the two terms more interchangeably, with assembly usually considered as part of the broader discipline

of manufacturing. Because the scope of this work is not limited only to the discipline of assembly and to avoid ambiguity, Boothroyd and Dewhurst's definition of manufacturing from here onwards will be referred to as "fabrication." The term "manufacturing" and terms related to it in this work will always refer to manufacturing as a broad discipline encompassing the smaller fabrication and assembly ones except for special cases, typically names of fabrication discipline-oriented concepts such as "Design for Manufacturing."

2.1 Integrating Design with Assembly

2.1.1 Design for Assembly

The most significant and industrially-proven DFA methods are the Hitachi Assembly Evaluation Method (AEM), the aforementioned Boothroyd-Dewhurst DFA method, and the Lucas method [14].

The Hitachi AEM integrates design with assembly with its use of an assemblability evaluation score, E, and an assembly cost ratio, K. The E index reflects a design's quality by seeing if the motions of the required assembly operations are simplistic and efficient straight-down movements, with lower scores obtained for non-simplistic motions associated with worse designs. The K index is a ratio of the estimated cost of the operations from the previous step and the historical cost of a similar product [17]. It can be seen based on these two scores that the Hitachi AEM is primarily focused on insertions of components, making it somewhat limited.

The Boothroyd-Dewhurst DFA technique is carried out primarily by calculating two different indices. The first index is related to part counts and obtained by asking the engineer/designer the following detailed questions about the functionality of each component [16]:

- Does the part move relative to the other parts?
- Is the part a different material than the other parts?

- Is the part separate to allow other parts to be assembled or disassembled?

The answers to these determine if the parts are needed, can be consolidated with another part, or are outright removed. From there the component count of the final product is compared with the theoretical minimum component count and an efficiency value determined to see how well the final product was designed. The producibility and assemblability index of the product is determined by having the engineer/designer answer questions related to the operations involved in assembling the product, such as how heavy the product is, which operations are being used, how accessible the joint is, and how the component will be gripped and inserted [17]. The answers are used to calculate an estimated assembly time and cost metric based on detailed tables in Boothroyd and Dewhurst's book. The second design efficiency index is calculated based on this estimated assembly time and cost. This generalizability, detail, and data traceability is what has made the Boothroyd-Dewhurst method the go-to one whenever DFA is mentioned.

The Lucas method focuses on using penalty factors to calculate three assemblability indices: design efficiency, feeding ratio, and fitting ratio [20]. The design efficiency index is calculated by determining which parts are considered non-essential and comparing the actual number of parts with the theoretical minimum of only essential parts. This information is obtained via designer-supplied answers to questions similar to those in the Boothroyd-Dewhurst method, but in less detail. The feeding ratio is calculated by having the designer answer questions about the part handling such as if it is abrasive, does it tangle, is it sticky, etc. [20]. The fitting ratio is calculated by answering questions about the operations involved in the assembly such as clamping, inserting, and fastening parts. These indices are then compared with thresholds, obtained from historical experience, to determine the design's general assemblability [20]. The Lucas DFA method is essentially an intermediate method between the Hitachi and Boothroyd-Dewhurst methods, requiring more detail and being more flexible than the former and less in both than the latter [17].

These DFA techniques have shown they are able to incorporate design considerations

into the aircraft assembly discipline, unlike the current post-production industry practices. Despite all this, there are several limitations in attempting to use them to tackle the aircraft production rate and cost issues that will arise in the next two decades. These limitations ultimately cause them to be insufficient in addressing the problem. To elaborate further on these limitations, the design phases from Figure 1.3 must be further defined. Specifically, that figure shows that aircraft design has three distinct phases: conceptual, preliminary, and detail design. In the conceptual design phase, the overall configuration of the aircraft is determined, its size and weight set, and its performance evaluated; in the preliminary design stage, the aircraft's physical shape becomes more defined and specialists in disciplines like structures, controls, and aerodynamics are able to perform more in-depth studies and tradeoffs; and in the detail design stage all the parts and minor systems and subsystems, along with the rest of the aircraft, are developed and tested [30].

What is not shown in the figure is that the different design phases are not limited to just aircraft design but also apply to manufacturing design, which has its own conceptual, preliminary, and detail design phases [31]. These manufacturing design phases tend to take place during or after the aircraft detail design phase. The specific activities that occur in these manufacturing design phases differ greatly between disciplines, analyses, and authors, however. This is because there is often an interplay between the aircraft design phases and the manufacturing design phases, with the two often becoming intermixed. This is almost never explicit because aircraft design is known by the more general term of product architecture design, defined by Ulrich as the mapping of a product's function to its physical components as well as the specification of the interfaces among interacting physical components [32]. As an example of this interplay, analyses from the detail aircraft design phase on a specific part can end up being used in the conceptual manufacturing design phase. And in a different scenario, material definition analysis from the conceptual aircraft design phase could also end up being analyzed in the conceptual manufacturing design phase. This intermixing can cause the terms to become muddled, on top of many

specific types of manufacturing analyses having their own separate and conflicting definitions of their respective design phases.

To try to be consistent, the activities that describe the manufacturing design phases will be based on those agreed upon by general consensus among the many different manufacturing analyses, that roughly match Pahl et al.'s definitions of the different product design phases, and that are closest in nature to the definition of the aircraft design phases. Therefore, the conceptual manufacturing phase revolves around "the specification of the principle solution," including creating the overall manufacturing definition and identifying what processes will be used, the level of mass production to be implemented or lack thereof, and a rough idea of how the product will be made. In the preliminary manufacturing phase the "specification of the layout" becomes the center of attention and the factory itself becomes better defined with more details in how the workstations are balanced, the floor laid out, and the tasks scheduled, while the processes used to manufacture the product also become more fleshed out. And in the detail manufacturing phase the complete specification of the "production, assembly, transport, [and] operating" processes are laid out and "elaborate detail drawings and parts lists" provided [33]. This is summarized in Figure 2.1.

Given those design phases and definitions, the primary issue with the DFA techniques is that they are only able to operate in the aircraft and manufacturing detail design phase, as stated by both Favi and Bouissiere [31, 23]. This means that high level product configuration trades can only be implicitly addressed due to the fact that these DFA methods are "inherently capable of handling individual parts only," according to Abdullah [14]. During aircraft design multiple configurations, such as wing position, tail type, engine type, etc., are considered and evaluated when addressing the customer's requirements. Different configurations have a profound influence on both the aircraft's performance and its cost and production rate, making selection of the correct configuration of paramount importance. Being able to evaluate and provide improvements to the configuration, and by extension the product itself, when there is an issue with the overall assembly therefore carries far

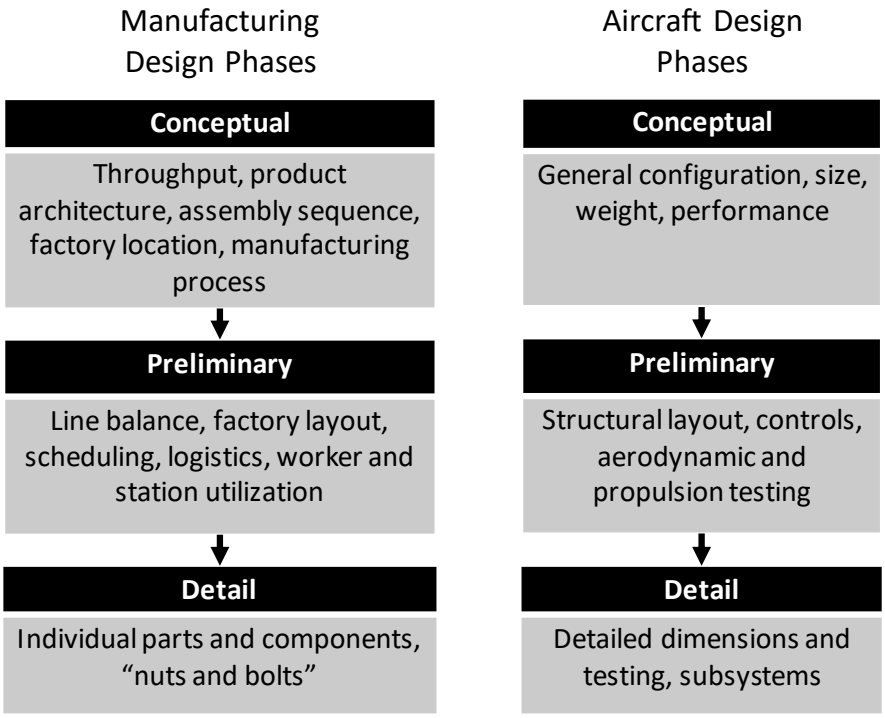


Figure 2.1: Different Manufacturing and Aircraft Design Stages and Example Activities in Each

more weight than doing the same with the configuration's individual parts, which is what the DFA methods are limited to. Using an aerospace example, it is pointless to design the minimum number of parts that satisfy the functions of a canard when a conventional tail, or even no tail at all, could yield better aircraft performance and throughput. These "conventional DFA" methods, as De Fazio calls them, also have significant difficulty dealing with complex assemblies that rely on specific geometries and manufacturing processes to carry out their function, which makes them resistant to the typical conventional DFA redesign suggestions [34]. Said complex assemblies are omnipresent on aircraft due to the emphasis of having as minimal a weight as possible for better performance, causing every part to have extremely specific geometries and processes used to create said geometries.

This is all to say that though conventional DFA methods acknowledge the importance of design and from there incorporate design with assembly, the design phases they operate on are so close to production and post-production that the majority of the costs, through-

puts, and trades are still already locked in. This is on top of the fact that the conventional DFA methods have been very well studied and implemented since the 1980s. If they were able to solve future throughput problems, they would have done so already and no such issues would exist. Current conventional Design for Assembly methods are thus insufficient to solve the issue in Section section 1.1 because they can generally only be used during manufacturing and aircraft detail design. In those phases, there is little design freedom left to make affordable changes. This leads to the first broad gap in existing capabilities:

Broad Gap 1: Conventional Design for Assembly techniques are limited to the detail design phase, where most design decisions are already defaulted.

2.1.2 Conceptual Design for Assembly

One possible way to mitigate this issue is to eschew asking the designers detailed questions and use Design for Assembly guidelines earlier in the design phase, such as during conceptual design. Examples of these include designing parts to be as symmetric as possible and easily handled in bulk from Boothroyd [16], maintaining independence of functional requirements and minimizing information content from Suh [35], and standardizing, simplifying, and ensuring quality of assembly operations and interfaces from Pahl, et al [33]. These guidelines do not need detailed geometry or components to already exist to be used and can be used early on. However, as noted by Abdullah, these guidelines are too general to be practical and are unable to provide a distinct methodology that can be easily followed. Furthermore, they are focused on already existing assemblies rather than on brand new designs [14]. In essence, where the implementations in the detail phase provided quantitative information but were too narrowly focused on individual parts, the conceptual design implementations have the opposite problem and are not as tunnel-visioned but only provide qualitative guidance and so are too general. The guidelines are therefore unable to provide the same types of benefits as the methods applied during detailed design and so do not solve

the issue.

They did, however, lead to more methodical implementations of DFA methods earlier in the manufacturing design phases, particularly conceptual design. Some of the first to do this were Stone and McAdams, who developed a DFA method focused on functional modules that can be applied during the conceptual design phase. They define functional modules as clusters of sub-functions that convert "flows" of material, energy, and information from input to output [21]. These functional modules are representative of the minimum number of parts in an assembly and can be made into assembly modules to become physical parts. By not requiring physical geometry or detailed geometry, this functional module method can be used during conceptual design and allow for a pseudo-calculation of the minimum number of parts in the same way conventional DFA methods try to minimize part count [22].

Favi and Germani noted several issues with Stone and McAdams's method related to the product architecture, mainly that the latter's method lacked consideration of the modules' spatial layouts and assembly sequences, or order the components are assembled in [36]. Favi and Germani addressed this by identifying the different interfaces between modules, which are mechanical, optical, magnetic, and ordering them in terms of priority. This information, combined with the number of interfaces each module has, was then used to generate an assembly sequence for the modules before they were converted to physical components and the conventional DFA methods applied.

Favi and Germani later on made further improvements to their method and added manufacturability aspects such as material and fabrication process selection [31]. They additionally noted that product design consisted of more than just assembly and so incorporated DFM and Design to Cost methods into their approach. To implement this, they started out with Design for Assembly considerations, because those are more compatible with high level decisions, and simplified the product structure and selected the most economic materials and processes. The DFM aspect was then used to design the components, in detail,

for minimum fabrication costs. Throughout this process, the Design to Cost methodology was used to calculate the cost of the product and allowed the DFA and DFM methods to communicate, making their approach truly multi-disciplinary.

Bouissiere et al. made further improvements to Favi and Germani's original method by making it applicable for complex products with numerous functions and interfaces and allowing it to compare different architectures [23]. They did this by using a simplified Digital Mock Up (sDMU), a simplified Bill of Materials (sBoM), and ergonomic and assembly analysis documents. These three items encode information about the overall shape and volume of the functional modules, the nature of the connection between their interfaces, the accessibility of the working areas, and the complexity of the assembly operations. The data from these new additions are normalized and combined with Favi and Germani's functional modules procedure to perform what Boussiere et al. call "modules analysis." In this analysis the complexity of all the modules is evaluated and scored in a component, assembly, and ergonomic context. This score provides an assemblability rating that designers can use to compare with other architectures, analyze why certain modules are so complex as well, and create design rules to prevent them from being so.

Krause and Halfmann [24] performed very similar work to Favi and Germani involving modules and interfaces. Their method differs from Favi and Germani's in that less time is spent finding different clusters of sub-functions to create functional modules and more emphasis is placed on the assembly sequences, the assembly time estimation, and the consolidation of interfaces to reduce part count. In particular, they demonstrated this with the redesign of an aircraft fuselage passenger cabin. The cabin was originally composed of 38 separate parts and 25 different interfaces and subsequently consolidated into just 1 assembly module and 5 interfaces, allowing them to more easily preassemble components and significantly reducing the assembly time.

These particular works were examined because they explicitly studied aircraft assemblies, but there are many other conceptual-level DFA methods and, for brevity, they will

not be reviewed. Regardless, from this myriad of different methods, it can be seen that it is possible to apply Design for Assembly principles in a more quantitative way than was originally done during conceptual design with conventional DFA's guidelines. With the methods just discussed in particular, the main goal of minimizing parts for better assemblability is able to be applied with almost the same results to functional modules. This results in Observation 3:

Observation 3: Conceptual DFA methods address the detail design shortcomings of conventional DFA methods.

Despite these benefits conceptual DFA methods, like conventional DFA ones, ultimately prove to be inappropriate in addressing future throughput issues. One of the weaknesses of these functional modules methods is that they almost completely rely on the product being modular, which is not always possible because it does not consider other disciplines. Structural feasibility and manufacturability in particular often combine to create insurmountable stumbling blocks. Favi and Germani attempted to address this in their later work [31], but manufacturability and structural feasibility both require more detail than can be provided with just functional modules and interfaces to be accurately considered.

There are two other larger issues. The first is that although conceptual DFA methods work with product architecture, they only operate on architectures in the aircraft detail design phase. Krause and Halfmann and Bouissiere et al.'s works, despite having aircraft assemblies as part of their use cases, only analyze the aircraft's subsystems, which typically do not get addressed until very late in the aircraft design process. They do not analyze major assemblies in the earlier phases like the wingbox or fuselage, which are significantly more likely to be throughput bottlenecks. This is as opposed to the ideal situation of analyzing said assemblies and trading off desired assembly characteristics versus their impact on the aircraft's overall performance. Doing the latter would involve making changes in the earliest design phases for both aircraft and manufacturing design, providing the highest

chances of taking advantage of every design decision possible to achieve high throughputs, but it is not currently done with conceptual DFA. This is in fact common to both conceptual and conventional DFA methods. Neither truly operate in the early design phases of aircraft design, effectively still throwing designs "over the wall" from the aircraft side to the manufacturing side. This results in the following broad gap:

Broad Gap 2: Current conceptual and conventional Design for Assembly methods only look at detail aircraft design and do not address the earlier aircraft design phases.

With the current emphasis on affordability combined with a much more cutthroat economic environment than in the 1980s and a demand for ever larger production rates, all possibly beneficial design decisions must be leveraged; early aircraft design phases cannot be left on the table and must be considered alongside the earlier manufacturing design phases in trying to maximize throughput.

The second large issue also applies for both conventional and conceptual DFA. Conceptual and conventional DFA methods only operate on the product and provide little input on the manufacturing systems that implement and manufacture the product. These manufacturing systems and the decisions related to them, such as deciding where the factories are located, what the logistics of obtaining the supplies are, and how the assembly lines should be laid out, define the resources to physically implement the product. Consideration of resources is needed to turn production and assembly times into production rates because times are distinct from rates. Production time looks only at the sum total time to fabricate and assemble an assembly and its components. It does not consider physical resources and so lacks any connection to the physical world. As an example, a product with a very large production time could nonetheless theoretically attain an infinite production rate, or how much of a product can be produced in a specific time interval, by having infinite space and an infinite number of factories, equipment, and workers. Production times need a connection to the physical world via resources to be turned into production rates.

Conceptual DFA methods occur too early on to look at resources adequately. Even conceptual DFA works like Favi and Germani's later attempts [31] that consider the beginnings of resource analysis by examining different material definitions and manufacturing processes are inadequate because no resources are actually allocated. This makes it difficult to predict an actual production rate. Tables of historical data like for conventional DFA can be used to try to estimate a production rate given a production time, but these are only accurate for tried and true processes and practices. If any new processes or practices are used, which is likely given that large changes would be needed for large production rate increases, said tables can quickly become inaccurate, inappropriate, or completely inapplicable. On the other hand, conventional DFA methods occur too late in the design process, where the resource allocation has been long defaulted. Little guidance is thus mentioned or provided on how these systems should be implemented when parts are being redesigned to reduce part count [37]. Though the conventional DFA methods were not intended to address those factors, that they are essentially left untouched leads to early design decisions that can be leveraged to reduce cost and increase throughput not being made. Both conceptual and conventional DFA methods have shown they can only predict production times, not rates, leading to the following broad gap:

Broad Gap 3: Conceptual Design for Assembly methods occur too early and conventional Design for Assembly methods occur too late in the design process to determine or change resource allocations and cannot adequately predict or address production rate changes by themselves.

These limitations make both DFA methods inappropriate to solve future throughput issues. To fix this, how the product can be embodied must be determined, leading to the requirement that the manufacturing embodiment, or preliminary, design phase must be considered when trying to improve production rates. This is because that exact design phase is where these resource definitions and allotments all occur. And as stated earlier,

such methods must also be able to operate in the early aircraft design phases. The next section goes over the current state of the art in DFA-based methods that do this.

2.2 Design for Assembly in Early Design: Current State of the Art

2.2.1 Product Lifecycle Management, Systems Engineering, and Other Methods

With the ever increasing market globalization at the end of the 20th century, companies had to be innovative and do so quickly and effectively to remain competitive. All the different entities in an organization had to be able to communicate with each other to get the product to market as soon as possible, akin to concurrent engineering but across all different fields, disciplines, areas, and systems. The new focus was on meeting the new requirements by trading as many variables as needed, variables such as ones related to the assembly process, material, and factory. The subsequent emphasis on combining all the different systems of a product can be best described as "systems-thinking," with one of its characteristics being a focus on data traceability. These systems-thinking methods, with their basis on efficiently designing and embodying a product and then getting it to market, filled in the voids left by conceptual and conventional DFA through incorporation of both the embodiment and conceptual design phases.

The Product Lifecycle Management (PLM) concept is one of the best-known examples of the systems-thinking methods. Product Lifecycle Management is defined as "a strategic business approach for the effective management and use of corporate intellectual capital" [38]. One of the main goals of using PLM is to, according to Sudarsan et al., integrate "all the information produced throughout all phases of a product's life cycle to everyone in an organization" [39]. Modern PLM techniques concentrate on building a support framework for product information that can access, store, distribute, and reuse all of the product's data [39]. This, in particular, includes support for data traceability during the conceptual and preliminary design phases so that designers can cost effectively decide the features to include at that point as opposed to during the procurement or manufacturing phases.

The PLM concept thus does not specifically implement Design for Assembly methods like the functional module techniques do, but is instead inspired by it to incorporate assembly considerations early in design. For this reason many different authors have come up with different techniques to implement the PLM concept. However, only ones more related to aircraft design will be reviewed because, otherwise, they would not consider the aircraft design phases at all and be generally inapplicable to the purpose of combining aircraft design with manufacturing.

Demoly et al. proposed a web-based approach that they call Assembly-Oriented Design [40]. Their framework is centered on designing the product and determining its assembly sequence concurrently. To do this they had product-assembly lifecycle stakeholders manage the product structure, define the product geometric information, work on the assembly sequences, define the assembly processes, and analyze the assembly operator activities. Their algorithm called Assembly Sequence Definition Algorithm (ASDA) is then used to compile all this information and generate and evaluate candidate sub-assemblies and assembly sequences. After the final sequence is chosen, the algorithm converts the information into files that can be used in other engineering domains and further define the product. Demoly et al. have since further expanded on this framework such as by working on a skeleton geometry model in CAD that describes design intents in a top-down manner [41], creating view models (functional, structural, geometric, technological, etc.) that represent the viewpoints and concerns of various stakeholders [42], incorporating tolerance analysis during processing of the assembly sequences [43], and developing a product relationship management approach that controls the flow of information between the product and assembly process domains [44].

Agyapong et al. looked at a manufacturing and cost-specific implementation of PLM in the form of the Product-Process-Resource-Cost (PPRC) methodology [26]. In their methodology they defined the possible processes, manufacturing operations, and sequences able to meet the product's requirements and then mapped the available resources and the

product's features onto said processes. Cost estimates are afterwards obtained by tracking the consumption of resources used to carry out the manufacturing processes. Agyapong et al. justified using product features instead of separate engineering activities such as Finite Element Analysis (FEA) and Computational Fluid Dynamics (CFD) because there is a more direct link between processes and product features via geometry than there is between processes and FEA/CFD. With this setup any changes brought about by the engineering activities are reflected in changes to the product's features and directly translated to cost. This allows the cost of alternative methods to be easily determined in early design and can provide justification for why a specific product, process, or resource needs to be changed. This ability to trace the cost back to the product allows for information to be available at all stages of product design and is why the PPRC method is the backbone of many PLM software such as DELMIA.

Gomez et al. created an Industrial Digital Mock-Up (IDMU) that is comprised of geometric and technological information on the product and its processes and resources [25]. Their efforts were aimed at using PLM techniques to allow designers to explore aircraft Final Assembly Line (FAL) alternatives in the early design stages. The method is split into five tasks: define the as-planned structure and group the aircraft's components and joints into sub-assemblies; define the supply chain and logistics plan and select the aircraft's assembly plant locations and transportation methods; define the final assembly line and determine the assembly sequence of the joints and aircraft components, assign the assembly operations to workstations, and allocate the various resources to each assembly activity; define the final assembly line layout to place the workstations and obtain the manufacturing plant's space requirements; and evaluate the FAL solutions on a cost, cycle time, and efficiency basis using fuzzy logic to be able to consider vague or ambiguous inputs. The product, process, and resource information and data utilized in all of these tasks are represented in an object-oriented meta-model using Unified Modeling Language (UML). Due to a desire to not be reliant on experience and rules of thumb and to be more methodi-

cal, the information needed for the different processes and resources are implemented as knowledge bases in the UML model [45]. Gomez et al. demonstrated their methodology by performing a use case with CATIA/DELMIA V5 and comparing a FAL with two assembly stations and with three assembly stations, determining that the latter had a higher production rate but cost far more [25]. Though they considered their work to be at the conceptual level, their efforts in embodying the assembly line via workstation assignment and plant layout indicates they actually covered the entirety of both the conceptual and preliminary manufacturing design phases.

Ortegon et al. used a methodology that is very similar to Gomez et al.'s and also incorporated an IDMU [46]. Their methodology involves using Measurement Time Methods (MTM) to estimate the time spent during the assembly activities and Value Stream Mapping (VSM) to identify improvements that can be made to the assembly line. MTM estimates manufacturing time by treating each activity as a set of basic motions. VSM is a lean manufacturing tool used to identify which activities do not add value to the product and can be potentially improved. This is done by mapping the characteristics and parameters of a product (size, weight, etc.) to the parameters of the processes in that product (drilling speed, cure time, etc.) and then identifying areas where performance is subpar, such as a bottleneck, and tracing it back to its cause [47]. Ortegon et al. demonstrated this methodology by drastically increasing the efficiency and production rate of the assembly system of a historic aircraft, the Bleriot XI, via eliminating its manufacturing bottlenecks and reducing the wait times [46].

PLM is not the only systems-thinking method. Another one is systems engineering, which is centered on requirements analysis, simulating all parts of the system, synchronizing development of the product's components, and validating all the subsystems and components against the initial requirements. Systems engineering is extremely similar to PLM, with the primary difference being that PLM is more physical and mechanical in nature while systems engineering is more overarching and able to handle the meta-physical

as well. This includes the complexity of all the digital models and data generated over the course of the product's lifecycle [48]. Yet other systems-thinking methods exist as well that are less formalized than either systems engineering or PLM methods but with similarities to both. This next part thus reviews some aircraft-related non-PLM works to see other ways systems-thinking methods can be applied to consider assembly considerations in the conceptual and preliminary phases.

Li et al. used the systems engineering concept of RFLP (Requirement, Functional, Logical, Physical) to model the complexities of finding the correct sequence to install and test different aircraft systems during the FAL process such that all requirements are met [49]. In RFLP, Requirement refers to what the system must do, Functional to what the system actually does, Logical to how the functions are carried out, and Physical to the components carrying out the function. Li et al. used RFLP models to create a test breakdown and check which functions are dependent on a specific assembly environment. The functions in these breakdowns are traced to their logical components and linked with their 3D physical CAD models. The assembly sequences considering testing and integration of systems are afterwards generated using this association of functions to physical parts. They tested this method on a use case of a student design project aircraft's avionics via the CATIA/ENOVIA V6 software. Their method shows that systems engineering methods nowadays are extremely similar to PLM, especially in the fact that ENOVIA v6 is able to implement the RFLP process and is a part of CATIA, which itself is a PLM-driven program.

Polacsek et al. described an approach similar to Demoly's Assembly-Oriented Design but focused specifically on aircraft and with little emphasis on the PLM aspect [50]. They proposed a model-based framework where an integrated model is shared between the disciplines, allowing different manufacturing and design aspects to be traced through different levels of abstraction. Designers and manufacturers can then use this integrated common reference model to observe the effects of their disciplines' decisions on the others' at different levels of abstraction and, essentially, design phases. Polacsek et al. later expanded

on this work to be able to predict production rates and reduce manufacturing bottlenecks by tackling the Resource Constrained Project Scheduling Problem [51]. The tool resulting from solving this problem allows designers and manufacturers to quickly see the results of design changes and reduce lead times.

Caggiano et al. implemented a digital manufacturing approach revolving around simulating the assembly and fabrication process in a virtual environment [27]. Their methodology is inspired by the DFMA approach and is similar to Gomez's but incorporates line balancing and has less emphasis on the PLM concept. The methodology starts with determining the FAL's assembly operations. The assembly sequence for those operations is then optimized using value stream mapping and the manufacturing time for the best sequence calculated. Assembly line balancing is afterwards done to estimate how many workstations should be used. This is verified using a Discrete Event Simulation, which is also leveraged to manage resources such as the number of operators at each workstation to further improve production efficiency. Caggiano et al. tested this methodology on a use case involving improving the production rate and efficiency of a Skycar light aircraft. Their method requires more detail than Gomez's and so is unlikely able to be used in the conceptual phase but is still adequate for the preliminary phase.

A large number of other systems-thinking works based on incorporating assembly considerations into the design step exist, but the above works are sufficiently representative of them that they will not be reviewed. A significant commonality they all have is they are able to trace all fabrication and assembly information back up to the product, its functions, or its requirements. This allows for the product to be redesigned based on the identified trouble-causing areas, enabling efficient design and evaluation of different product configurations that was not possible before. Similarly, the works all incorporated preliminary and conceptual manufacturing configuration decisions as well. The preliminary level decisions in particular included tradeoffs regarding line balancing, assembly sequencing, factory layout, job scheduling, and logistics, which all have a significant impact on the final manufactur-

ing performance of the product. Both of these aspects were not present in conventional or conceptual DFA methods and can be utilized to take advantage of early design decisions to improve cost and throughput as much as possible. This is summarized in Observation 4:

Observation 4: State-of-the-art DFA-based methods address the shortcomings of conventional and conceptual DFA methods.

However, despite displaying the capability to link assembly considerations back to design, none of the above works have actually addressed the design itself, instead focusing only on the assembly considerations. In other words, all of the discussed works assumed that an aircraft had already been designed and sized and the only step left was to determine the different fabrication and assembly variables. In contrast to this, the design and optimization of the product being made is just as important as the processes and resources following it and has direct effects on it, so the product cannot be assumed to be set. This leads to the following broad gap in capabilities:

Broad Gap 4: Many state-of-the-art methods in the assembly literature that consider DFA techniques and address their deficiencies only look at the assembly aspect and are still silo'ed off from aircraft design.

The next section addresses the works that do not have this dilemma.

2.2.2 Incorporating Early Aircraft Design

At the same time that PLM, systems engineering, and other systems-thinking methods were taking off due to the emphasis on affordability, many aircraft designers also realized they could no longer design airplanes with amazing performance "at any cost" like they could in the past–they had to consider affordability too. They thus concentrated their efforts on better implementing concurrent engineering to explicitly link the aircraft design process with fabrication and assembly considerations. They did so to more accurately estimate

manufacturing costs early in the aircraft design process though. Because of this, the majority of the following works are from the point of view of the aircraft literature branching out into the manufacturing one using systems engineering principles. Consequently, due to the focus on cost estimation, assembly analysis was often treated as a part of the overall manufacturing analysis and usually not as a major driver in itself.

One of the first major works in the area was from Schrage et al. in their design of a high speed civil transport aircraft. They examined the wing in particular detail because conceptual aircraft design primarily revolves around sizing an aircraft wing, making its in-depth analysis representative of designing the entire aircraft [52]. Schrage et al. used the Integrated Product Process Development (IPPD) approach to simultaneously size and perform multidisciplinary optimization of the wing and obtain the associated lifecycle cost to estimate the manufacturer Return on Investment (ROI). The IPPD approach itself is defined as "a systematic approach to the integrated, concurrent design of products and their related processes, including manufacture and support." The ideal IPPD approach is visualized in Figure 2.2 and is essentially the ultimate goal of what all aircraft-considering systems-thinking methodologies strive for. In it, there is a feedback loop such that the necessary design trades can be made at each design phase while also allowing the information and implications from said trades to easily flow up or down different phases or across disciplines as needed. This results in not throwing decisions "over the wall" and connecting the two disciplines during all of their different design phases, enabling parallel iterations for maximum design improvements. The strong concurrent engineering approach in this work was emphasized, more so than works before it, because usage of increasingly advanced materials for improved aircraft performance led to manufacturing limitations that also affected said performance, creating a loop. Feedback between manufacturing and aircraft design thus had to exist to accurately capture the effects of the novel materials.

Schrage et al. implemented a limited version of this approach that stops before the detail design phases by first performing conceptual design to initially size the aircraft. Prelimi-

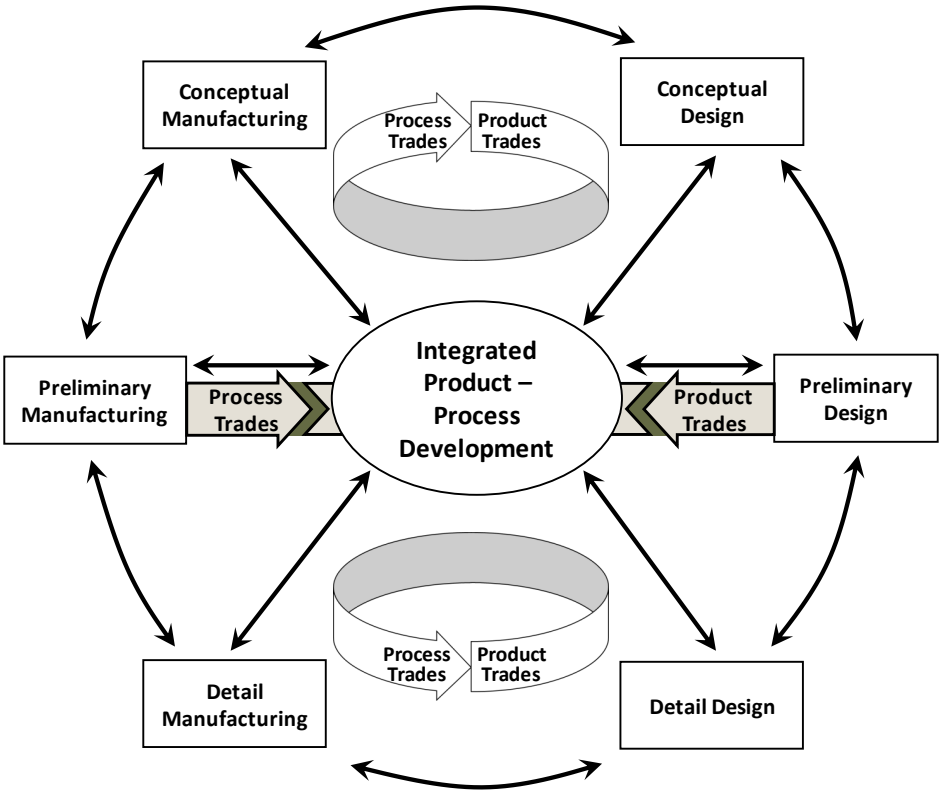


Figure 2.2: Ideal IPPD Approach, Adapted from Schrage et al. [52]

nary aerodynamic, propulsion, and structural analyses are then carried out. The preliminary structural analysis in particular provided the necessary context regarding the different materials, processes, and complexities examined in the manufacturability and cost analysis. The costing was performed using several different programs that based the costs on weight regressions, manufacturing complexity factors, and parametric relationships with previous aircraft. The IPPD approach thus allowed for multidisciplinary cost estimation at both the conceptual aircraft design and preliminary manufacturing levels, but the emphasis on cost meant that manufacturing time and the assembly discipline were barely examined, on top of little mention of production rates.

Marx et al. continued Schrage et al.'s work and refined it [28]. The KBS they developed reads in the material and manufacturing constraints used and produces a set of suggested manufacturing processes. Their wing production cost model does not rely as much on historical weight regressions as previous models and focuses more on labor and material

cost, subsequently allowing them to better estimate production time and examine different manufacturing processes. And the lifecycle cost model they implemented takes advantage of both of these developments to directly relate conceptual and preliminary level aircraft design decisions to manufacturing cost and time. They demonstrated this by explicitly analyzing the impact of three different structural design concepts for the high speed civil transport aircraft in terms of aircraft performance, production cost, lifecycle cost, and manufacturer ROI. Their work provided a way to obtain better design justification for aircraft designers, enabling quantitative metrics from more detailed design phases to be attached to early design concepts to compare them. By using their KBS they were also better able to incorporate more detailed assembly considerations than Schrage et al.'s work and focus more on production time. However, production rate considerations were still mostly neglected.

Work further refining Schrage's IPPD approach by focusing on systems engineering, PLM, and Product Lifecycle Engineering (PLE) with an aircraft design-centered application was carried out through the combined efforts of Schrage's Integrated Product Lifecycle Engineering (IPLE) laboratory, the Aerospace Systems Design Laboratory (ASDL), and Boeing. The framing of the IPPD approach and tradeoff environment in Figure 2.2 as a decision-making aid resulted in the IPPD methodology, shown in Figure 2.3. This methodology aids designers in generating and selecting the appropriate alternatives to best meet the customer's needs. It does so by combining together quality engineering methods with systems engineering methods and breaking the latter down into discrete steps, akin to the 11 steps in Boeing's systems engineering process [53].

One of the more notable demonstrations of the IPPD methodology is its use in the notional redesign of an F-86 Sabre with modern technology to improve its combat capabilities and producibility as part of an experiential-learning class project [55]. The F-86 Sabre was selected in particular due to data on how it is sized and manufactured being easily obtainable. The redesign was done with a PLM approach to ensure the aircraft's

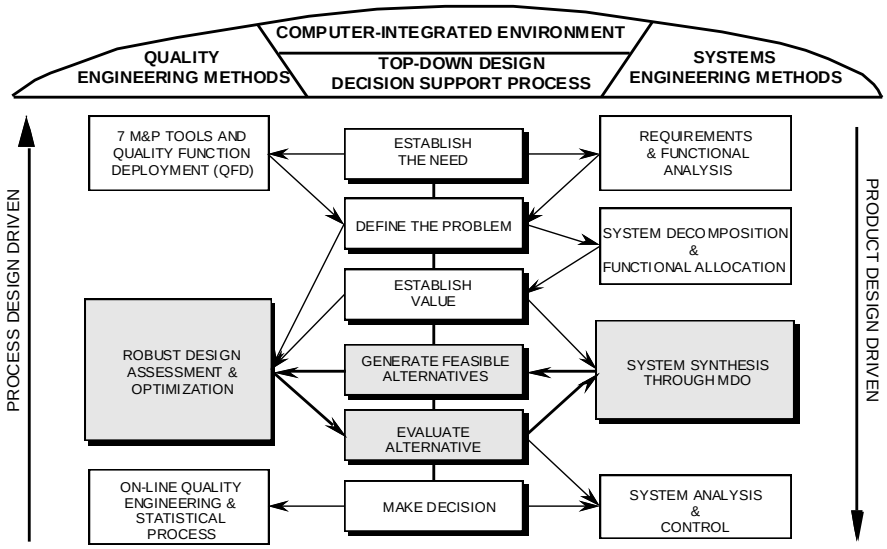


Figure 2.3: Generic IPPD Methodology, Adapted From [54]

capabilities were sufficient for its entire lifecycle. Different design alternatives were characterized primarily by changes in the aircraft's configuration, including changes in metallic materials, payloads, and engines used. Several more classroom iterations of this were carried out to demonstrate the advantages of performing integrated design and manufacturing trade studies using PLM tools such as CATIA and DELMIA. In these, students teams were asked to "define the problem, establish value, generate feasible alternatives, evaluate the alternatives, and make a decision" [56]. The decision-making was enabled by tools such as "Quality Function Deployment (QFD) matrices, morphological matrices, a Pugh evaluation matrix, and a Multi-Attribute Decision Making (MADM) technique, designated the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS)" [56]. The first of these iterations examined usage of metallic versus composite materials [56]. Later ones used NASA's Advanced Subsonic Technology Program, in cooperation with Boeing as a primary subcontractor, to create a knowledge base to dive further into trading different composite manufacturing processes [57]. Despite all these uses of an integrated design and manufacturing environment, production rate was nigh never the focus, only the aircraft's cost.

Sirirojvisuth did work similar to Marx et al. while basing his efforts on the IPPD approach for the F-86 and created his own, more granular cost model that was also able to incorporate production rate requirements [58]. Sirirojvisuth's cost model is hybrid in nature and has varying levels of fidelity, using weight-based regressions for better flexibility and activity and process-based techniques for more detailed cost and time estimates. The activity and process-based estimates allowed for greater sensitivity in material and manufacturing process trades. In particular they enabled more accurate analysis of composite materials in early design since composite materials carry many manufacturing restrictions that invalidate traditional weight regressions. Sirirojvisuth obtained the necessary inputs to use these later-design techniques by first performing conceptual design with NASA's Flight Optimization System (FLOPS) software and then linking it with the CATIA and DELMIA CAD packages to help define, store, and analyze the detailed part geometry as well as carry out manufacturing studies. He showcased all these capabilities by evaluating five different structural concepts for an F-86 wingbox, which also involved comparing traditional metallic material concepts versus newer composite-based ones. His usage of CATIA and DELMIA, both PLM-based packages, allowed him to be able to estimate production rates, though like the previous IPPD works it was not the focus.

A series of works that combined all these IPPD tradeoff capabilities demonstrated on the F-86 that did focus on production rate is the Manufacturing Influenced Design (MInD) methodology [59]. The MInD methodology's goal is to move away from traditional cost estimates relying on historical, weight-based regressions during aircraft design; these are not accurate for emerging technologies such as composite materials, which require more detailed manufacturing models to better capture their improvements and limitations' effects on cost, throughput, and aircraft performance [60]. In the methodology, the aircraft is first sized to meet the customer performance requirements. The resulting Outer Mold Line (OML) is used by a structures-to-manufacturing translation tool to provide data for the time and cost estimating tool, SEER-MFG. From this the number of lines and shifts and work-

ers, the production schedule, and the cash flow model are created and the manufacturer ROI calculated. Afterwards the lifecycle costs in the form of sustainment, research, development, and operational cost estimates are calculated with the SEER-H tool. The process is ultimately repeated many times with different aircraft configuration and manufacturing variables to generate data for surrogate models, which significantly speeds up the design space exploration. Cost, production time, and production rate estimates are thus easily traced back to the product features that generated them while conceptual and preliminary aircraft design is explicitly done, unlike the other works. Notably, MInD uses mathematical relationships to convert the vehicle's OML into the internal structure and geometry data needed for its process-based costing tools. This is done instead of using parametric CAD tools like CATIA that require substantial user experience and are difficult to parameterize for large configuration changes. The use of the specified process-based costing tools additionally allows for a user-friendly way to build-up and model a large variety of processes by combining different individual activities together.

There are many iterations of the MInD methodology. Sundaresan et al. added a more detailed structural sizing module and performed a case study on the manufacture of a redesigned composite F-86 wingbox [61]. Siedlak et al. added a more detailed production flow model using Discrete Event Simulation (DES) to obtain more accurate cost and throughput estimates [62]. This is supplemented with value stream mapping to identify and address manufacturing bottlenecks to balance the production line. Siedlak et al. then compared different manufacturing concepts by evaluating the baseline conventional composite wingbox against a machined aluminum design and an advanced composite design [63]. This allowed insight into details such as how the increased performance of a composite wing fared against the reduced cost and improved manufacturability of an aluminum one. The cash flow formulation is also refined and a multi-objective optimization algorithm implemented allowing customer requirements to be compared with cost. The added emphasis on production optimization caused this iteration particular iteration to be called

Manufacturing Influenced Design Production Optimization (MInDPRO). Siedlak's latest update added additional detail to the costing models and incorporated the concept of a digital thread, which is the linking of models from various disciplines using common inputs to speed up design and enable multi-disciplinary trades in previously disconnected disciplines [29].

Butterfield et al. moved away from directly basing their work on Schrage et al. when examining early aircraft design and manufacturing together. Their methodology is similar to Gomez's [25] but emphasizes CAD and PPRC tools more while also explicitly incorporating the aircraft design process [64]. The methodology is similar to the ones above in that it starts with determining conceptual parameters such as thrust and wing loading. These are then used to develop and automatically generate CAD geometries for FEA and CFD simulations, which serve as input for the manufacturing analysis. Said input information is all tracked and integrated within the DELMIA Process Engineer planning environment, which itself uses various DELMIA suites to simulate different manufacturing process plans and track their flow and the flow of related production line resources. Butterfield et al. demonstrated their methodology by varying the number of fitters and riveters during the fuselage assembly process and observing the resultant effects on efficiency, cost, and production rate. The methodology's integrated information flow allows all information to be easily tracked from the early design stage all the way to production and assembly, making it possible for aircraft design decisions to be easily brought downstream.

Jin, Curran, and Butterfield tried to estimate assembly times early in design to bring more detailed assembly analyses upstream in the design process [65]. They did this by using Maynard Operation Sequence Technique (MOST) to estimate time, similar to Sirirojvisuth, with the difference being that Jin et al. also incorporated a KBS to incorporate expert domain knowledge into the estimations. The KBS allows for estimation of an assembly time and its associated setup time based on parameters such as a part's material, size, location, and operation type performed. Jin et al. integrated this method with the

Delmia Process Engineer software and demonstrated it by performing time analysis on an aircraft panel. Though they did not perform an explicit assembly analysis, they claimed their method could be used to generate assembly times for analyses such as line balancing in a more automated way in early aircraft design than with the current method of consulting assembly handbooks.

Friel, Marzano, and Butterfield used a digital mock up similar to Gomez et al. [25] to consider part tolerance information early in design [66]. Tolerances are an important link between the product's design and manufacturability consideration but are normally set early in aircraft design and only analyzable in later manufacturing design. Friel et al. combined cellular modeling of tolerance zones with digital mock ups in CAD software to analyze and visualize the impact of component tolerances on assemblies. The implementation of digital mocks ups in particular allows the tolerance information to be traced back up to the design configuration that generated it, providing insight into assembly feasibility, time in early aircraft design. Friel et al. examined an aircraft panel and its holding fixture using this method and proposed further research in the topic. A notable omission, however, is how their tolerancing information could be obtained early in aircraft design.

A large number of other works in the literature discuss integrating manufacturing with aircraft design in order to obtain better cost estimates for the purposes of concurrent engineering. A good summary of them is in Price et al.'s survey paper, where they reviewed the state of the art in incorporating all the different aircraft design disciplines together through a systems engineering viewpoint, with many of them authored by Price himself [67]. All these works, and those derived from them, generally have the same flavor as the ones already reviewed above. Because of this, these works will not be further analyzed.

Overall, the discussed works are able to incorporate fabrication and assembly modeling while also being able to link that information back to the aircraft design process, which itself is leveraged to be able to further affect cost and production time. They improve on the systems-thinking works in the previous section by considering the conceptual and the

preliminary aircraft design phases. In this way they are truly multidisciplinary. Despite this difference, the two types of methods do share similarities. A major one is that the most capable methods can easily extend their analysis to the detail design level for both manufacturing and aircraft design. For example, there are works in both groups of systems-thinking methods that use CAD-based software and link it with manufacturing software such as DELMIA. The linkage with the CAD-based software allows for individual, detailed parts to be modeled, enabling usage of detail design manufacturing analysis with the manufacturing software. Similarly, the ideal IPPD approach in Figure 2.2 indicates connections with detail aircraft design, which can be achieved with multifidelity models that look at both large assemblies and individual parts for areas such as structural analysis. Or, copying the methods that don't look at aircraft design, CAD and manufacturing software could be leveraged such as in Sirirojvisuth's work to not just perform detailed manufacturing analysis but detailed aircraft analysis at the part level too. Put simply, detail design analysis with systems-thinking methods is possible should it be desired. But because the majority of improvement opportunities lie in design phases earlier than detail design, it is more important to focus on the analyses in the earlier phases first. If the earlier phases settle on an overall non-optimal design due to lack of capability, it does not matter how good the detail design optimization is, the overall design will still suffer. For this reason the scope of this work is limited to the conceptual and preliminary aircraft and manufacturing design phases.

Another similarity between systems-thinking methods that do and don't examine aircraft design is that neither are perfect. The aircraft design-considering works all share the same deficiency: due to originating from the aircraft design discipline and then branching out to the manufacturing one, the works miss important analyses in the conceptual and preliminary manufacturing phases. This is made more complicated by different works missing different aspects. The earlier works based on Schrage et al.'s efforts lack throughput considerations; a large portion of works based on Butterfield and Price's efforts lack consideration of different, particularly composite, materials, and works related to Price's

efforts tend to not explicitly examine throughput as well; and all of the works reviewed have very basic to nonexistent implementations of important assembly analyses such as line balancing, assembly sequencing, and the physical layout of the factory floor. Similar to how the works in the previous section missed throughput-improvement opportunities by not considering the early aircraft design phases, the works in this section missed such opportunities by not taking advantage of major analyses available in the early manufacturing design phases. These current methods have essentially imperfectly copied Schrage et al.'s ideal IPPD approach back in Figure 2.2 and underutilized some of the feedback loops, interdisciplinary communications, and multidisciplinary tradeoffs by not executing important assembly analyses to their fullest potential. This leads to the fifth broad gap:

Broad Gap 5: State-of-the-art methods that consider DFA techniques and aircraft design insufficiently consider assembly topics important to throughput improvement.

It can be seen that systems-thinking methods of both types have their own respective deficiencies. In this present state, they may or may not be able to solve future production rate issues, since many have not been fully implemented and tested in industry yet; however, they would have a much better chance of doing so if they eliminated their deficiencies via one being able to make the other's tradeoffs. In other words, the methods without aircraft design would need to incorporate aircraft design, or the methods with aircraft design would need the manufacturing analysis capabilities of the methods that do not. The best approach to address future production rate issues therefore does not lie solely in one group of methods or the other, but in some framework that combines the tradeoffs both can make and better integrates the two disciplines.

Assembly considerations are one of the primary candidates to address the production rate and cost issues the aircraft industry will face in the next two decades. This is because the area of assembly has a significant influence on both. However, addressing assembly issues during an aircraft's production phase is unlikely to yield significant benefits. More

benefits can be found considering assembly during the design of new aircraft. Conventional and conceptual DFA methodologies have yielded significant positive results in the past but are still insufficient. They are currently unable to provide large enough improvements because they can only be used during the detailed design phases, where the majority of decisions have still already been made and costs and production rates essentially locked in. State of the art systems-thinking methods remedy this by operating in the earlier aircraft design and manufacturing design phases. There are two distinct groups of these systems-thinking methods, one that has more comprehensive coverage of important assembly topics and one that directly incorporates early aircraft design. Despite their potential, neither of these groups of methods fully utilize all of each other's capabilities that could cost-effectively improve production rates. This lowers their chances of being successful in addressing future aircraft production rate issues. To have higher chances of being successful, one group needs to incorporate the other's capabilities and better integrate early aircraft design with early assembly design. And because methods in both groups can easily increase their analysis fidelity to the detail design level, the scope of this work in trying to obtain better methods will not consider the detail design phases. This leads to the overarching research question from section 1.3.

2.3 Better Integration of Early Assembly Design and Early Aircraft Design

Current state-of-the-art methodologies that incorporate aircraft design and assembly all have some deficiencies in trying to address future throughput issues. Their capabilities need to be improved in order to remedy the situation. To carry this out, a baseline must be chosen to improve. The options are either systems-thinking methods that more thoroughly consider the early manufacturing design phases but lack aircraft configuration trades, or systems-thinking methods that consider both the early aircraft and manufacturing design phases but are less thorough regarding the manufacturing one. In terms of ease of development, the latter are most likely the better baseline. This is primarily because the latter

already have manufacturing considerations, all that is needed to get them up to par is to make comparatively small additions to what is already there. This is as opposed to the former, which do not at all consider aircraft configuration design and so have basically already locked in the configuration and are in the aircraft detailed design phase. It means an entirely new module from essentially an earlier design phase would need to be developed and fitted to the existing frameworks. Doing so is more complex and time consuming than adding a few additional types of assembly analysis. Aircraft design-considering systems-thinking methodologies are thus the simplest-to-implement baselines to try to answer the overarching research question and so will be used.

The easiest way to identify the assembly-related capability gaps in the aircraft design-considering systems-thinking methodologies, in essence to find which interdisciplinary feedback loops need to be established or strengthened, is to first examine all the different ways the two disciplines can be linked to each other. According to Joneja, the design discipline in general can only affect manufacturing by making decisions on three different areas: tolerancing, geometry, and material considerations, as shown in Figure 2.4 [68]. The linkage comes from the fact that these areas are present and have ramifications in both disciplines, and so changing an area in one discipline will affect the other. Due to this how they affect each discipline is further elaborated on.

Geometry is heavily emphasized on the aircraft design side of matters. This is because defining the aircraft's geometry is one of the main goals and products of early aircraft design, with the size and shape of the aircraft being almost the sole aircraft performance determinant in conceptual design. Similarly, aircraft design is essentially a highly specialized version of what is called product architecture design for most other products, where a product's functions are given physical form. As geometry is necessary to be given a physical form, it is a fundamental linchpin of aircraft design. On the assembly side, geometry has a large influence on the assemblability of the product. The conventional and conceptual DFA methods, the first to even try to integrate manufacturing with early design, are

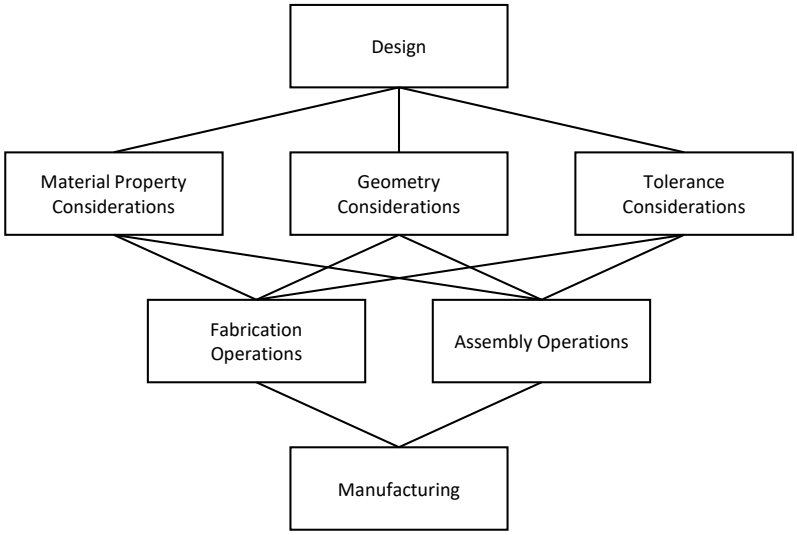


Figure 2.4: Ways Design and Manufacturing can Affect Each Other, Adapted from Joneja [68]

almost solely dedicated to ensuring the geometry is optimal to obtain the lowest assembly costs, times, and throughputs. And similar to how geometry is what gives physical form to an aircraft's function, geometry is what is being operated on when performing assembly. All assembly revolves around geometry in some way, shape, or form, which speaks to its importance in the discipline.

Material considerations, on the aircraft design side, take the form of constitutive relations used to structurally size the parts. They help determine the structural integrity of the physical form of the aircraft. In particular, materials help determine the weight of an aircraft, which is an important performance indicator. On the assembly side, material considerations are used to select the manufacturing processes that create the part from a specific material and then combine them. These processes affect how expensive the assembly operations are and how quickly they can be done and so are a defining assembly variable. They even spill over back into aircraft design by being able to impose geometry limits that can impact the aircraft's performance, depending on the processes used. Material considera-

tions also affect the handle-ability of the product since some materials are more difficult to assemble than others due to their properties. All these factors mean that variables related to materials can have large effects on cost and production rates on the assembly side and performance on the aircraft design side.

Tolerance considerations have a wide variety of effects on aircraft design. As reviewed by Friel et al. and shown by Curran et al. and Bhachu et al., which are representative works on how tolerancing affects design, accommodating for it produces effects ranging from enlarging parts and increasing weight and reducing performance, to having to deal with additional aerodynamic drag from improperly fitted parts, up to needing to redesign the parts entirely because they are unable to be assembled [66, 69, 70]. Tolerance considerations thus share similarities with manufacturing process considerations in that the majority of its impact on aircraft design is a result of trying to accommodate it. Its effects on assembly design are comparatively more direct and there are many more of them too. According to Choi et al., Curran et al., and Bhachu et al., these impacts include the amount of overtime, inspection, and rework labor spent checking and fixing parts, the amount of scrapped parts that need to be remade, and the cost of tooling and equipment since tighter requirements need more specialized capital equipment [71, 69, 70]. These all have significant effects on how much the product costs and how quickly it can be made, making tolerancing an especially important area for the assembly discipline.

With the significance of all three areas on each discipline explained, they now need to be examined in greater detail to aid in identifying and addressing capability gaps in the current baseline methods. However, it is not necessarily possible or optimal to examine all three since together they encompass a very large number of topics. Therefore the determinants of which of the three to look at in more detail are their potential impacts on cost and throughput and their availability in early design. The potential impacts of all three, based on the explanations above, are undeniable. It is doubtless that tradeoffs involving all three will have significant effects on the aircraft's cost and production rates given how fundamental

all three are, especially on the assembly side. Due to this, potential impacts are a non-factor because all three are impactful.

The same cannot be said about their availability in early design. All the systems-thinking works in subsection 2.2.1 and subsection 2.2.2 were able to incorporate geometry and material considerations, and so those two areas and the information needed to examine them are guaranteed to be available early in design. It is much more difficult to do so with tolerancing considerations. Friel et al. tried to incorporate tolerancing as a link between the product's design and assemblability, but provided no mention on where the tolerancing data was to come from [66]; Curran et al. was able to demonstrate the impacts of tolerance relaxation on aircraft performance and direct operating cost, but had to build and obtain tolerance data from an actual engine nacelle to do so [69]; and Bhachu et al. had to obtain data from an already built wing spar lap joint for their analysis [70]. While tolerance considerations can be used to link aircraft and assembly design, as shown by these works, it is incredibly difficult to acquire the necessary tolerance data before detail design has already occurred. This is even acknowledged by Zhao et al., whose goal was to specifically implement tolerance considerations in early aircraft and manufacturing design [72]. In their work they were forced to turn to historical sources, manufacturing handbooks, and subject matter experts to try to obtain sufficient data. But tolerance data, much more so than just simple geometry and material cost and time data, is very part and process-specific and is likely not applicable for newer processes or different geometries. Additionally, baseline labor rates for inspection, rework, overtime, and scrap activities are already difficult to obtain in early design. Acquiring such rates as a function of tolerance requirements is exponentially more difficult. This is doubly so for capital equipment costs and production rates as a function of tolerance; suppliers already rarely provide baseline values and will definitely not provide tolerance-dependent ones unless they are assured that their equipment will be used in the final design, which does not typically happen until basically everything is cemented in detail design. While it is possible to link aircraft design and assembly with tolerance consider-

ations, it is currently far too difficult to do so in the early design phases. For this reason, geometry and material areas will be examined more thoroughly while tolerance ones will not be due to the present impracticality of doing so in early design, which is summarized in Observation 5:

Observation 5: Geometry and material considerations can be used to better integrate early assembly and aircraft design.

Examining the geometry area more carefully, the most notable gap in aircraft design-considering systems-thinking methods is that none of them elaborate on how the aircraft geometry affects assemblability. Many of the methods detail how part size can influence the amount of material needed, thereby changing the material cost and making fabrication and assembly operations take more or less time depending on the size. However, none of them specifically mention how the shape of the parts and assemblies affect the way and the ease with which they can be put together. This is a noteworthy topic because, on the assembly side, some shapes can cause part assembly to be subpar or physically impossible due to lack of space, obstruction, or interference. And on the aircraft design side, some shapes provide better structural benefits such as increased stiffness and rigidity. As mentioned earlier, this is one of the main ways geometry affects assembly and so is an important capability to add. It is thus desirable for systems-thinking methods with aircraft design to have a formal way describe the impact of part and assembly geometry on how they can be assembled and the ease they can be assembled with.

In fact, it turns out that this is more than just desirable but actually a need. The formal way to do all this is Assembly Sequence Planning (ASP) and it is an assembly analysis step performed in nigh all of the non-aircraft design systems-thinking methods and was present in most of the ones reviewed in subsection 2.2.1 and absent in all the aircraft design systems-thinking methods. This makes ASP one of the analyses that allows the non-aircraft design methods to be more comprehensive in the assembly sense than the aircraft design

ones. By incorporating ASP, an additional feedback loop can be established relating the impact of a part's shape on both how well the aircraft performs to how much it costs and how quickly it can be made, better embodying the ideal IPPD approach. ASP also enables additional tradeoff capabilities which can result in increased throughput and reduced cost because it is one of the first activities performed when optimizing an assembly; selecting the correct sequences when starting out allows the other assembly topics downstream to be optimized in a more global manner, making them less likely to be trapped trying to find the best solution for an inferior sequence. Li et al. emphasized this by stating there is no foundation with which to balance the assembly lines or determine the throughput without a well-designed assembly sequence [49]. Incorporating ASP will, at the least, provide a quantitative design justification on why a sequence was chosen and enable better data traceability, an overarching goal of systems-thinking methods. This is as opposed to current aircraft design systems-thinking methods choosing their sequences at random or based on industry expert input, like with Butterfield et al.'s work or the various MInD works: random sequences do not guarantee optimality and experts may or may not exist and their inputs may or may not be applicable for new designs. Consideration of ASP will at least partially address the overarching research question of this work and so be added, leading to Observation 6:

Observation 6: Assembly sequence planning can be used to better integrate assembly geometry considerations with aircraft design.

Examining the geometry area resulted in an identified capability that can be infused with current aircraft design systems-thinking methods to help answer the overarching research question. The material area will be looked at next. As mentioned when it was first introduced, its biggest impacts are related to the materials and corresponding manufacturing processes used to perform the fabrication and assembly. In regard to the materials specifically, composite materials like carbon fiber and fiberglass are currently used

much more often due to their many performance benefits over traditional metallic materials. These aircraft design benefits also come with a significant assembly benefit: parts can be fabricated in such a way that they do not require assembly afterwards, possibly reducing part count similar to the conventional DFA methods and, from there, possibly increasing production rate and reducing cost. A carbon fiber stringer, as an example, can be co-bonded or co-cured to a spar during fabrication instead of being riveted to it after both are fully fabricated. This ability for separate parts to be made in a single fabrication step as opposed to requiring assembly afterwards will from here on be referred to as "integral fabrication" for lack of a better, material-agnostic term. While traditional metallic materials also sometimes have this ability, such as milling out a spar with integral stiffeners as opposed to making the stiffeners separately and then riveting the two together, the caveat with composite materials is that their handling is more complicated. This results in the number of steps and the accompanying complexity involved in trying to make an integrally fabricated part vary depending on the process used. This means it is not always necessarily the case that making an integrated part with a lower part count will cost less or take less time than making a non-integrated one. A part with a lower part count could actually be more expensive due to the high complexity or number of steps involved in getting it to said lower part count. Due to this, tradeoffs regarding whether parts should be integrally fabricated or not carry a large amount of weight. In spite of that, integral fabrication tradeoffs are currently not explored in any of the aircraft design systems-thinking methods. This makes integral fabrication an important material factor to incorporate to link the performance benefits of using composite materials on the aircraft design side with their tradeoffs on the assembly side.

With the materials aspect of the material area covered, attention is now given to the manufacturing processes aspect. As was explained earlier, they have a large influence on the assembly time, rate, and cost. It makes sense, then, to incorporate the ability to compare multiple processes with each other and see how each affects production rate and cost to select the most desirable one. A vital component in ensuring a fair comparison is to look at

the space requirements of the equipment sets used in each of the manufacturing processes. Space is an important resource in defining throughput as each manufacturing process uses different activities that require different sets of capital equipment, which each need a certain amount of space. Available space can thus both limit and improve throughput, depending on the process. On the more constraining end, a manufacturing process's equipment must fit in the available space lest the entire process be written off as not feasible. On the other hand, a seemingly inefficient process that uses little space can be leveraged to have significantly higher throughput than an efficient process that requires a very large factory. These considerations play a large role in optimizing production rates.

These considerations are also often overlooked because most works deal with smaller assemblies and so are more limited by the desired throughput and cost than by the available space. Similarly, it is often feasible to scale up the needed space given a desired throughput for those smaller assemblies because they are small enough that it is unlikely they will reach a spatial limit. This is not the case with the aerospace industry because aircraft parts and assemblies are very large and factories and buildings can only be so big, making available space much less scalable and so a much more limiting factor.

Despite this, spatial constraints are still sometimes not accounted for in the aerospace field. As an example, the MInD-based work of Siedlak et al. [29] has many of the ideal IPPD approach's capabilities, such as being able to analyze aircraft configurations produced with different materials and manufacturing processes. However, it makes the assumption that its factory is spatially unconstrained. This simplifies the throughput and cost analysis but is unrealistic because no actual factory has unlimited space. Their work attempts to mitigate this by assuming that no more than a set number of each type of workstation is allowed in the factory. However, setting such a limit is quite arbitrary and subjective; what prevents one type of workstation from having more stations than the other, and if the factory is spatially unconstrained, what prevents there from being an infinite number of both? Adding spatial considerations eliminates this question and those similar to it as

well as making it explicit whether a manufacturing system is physically feasible. It adds data traceability and justification on why only a certain number of workstations was used. Spatial constraints must therefore be accounted for when optimizing production rates cost-efficiently.

Another consideration is that the best throughput and cost may not necessarily be the absolute highest throughput or lowest cost. This is because, especially when considering equipment costs, the highest throughput design/process combinations can tend towards very high costs, while the minimum cost combinations can likewise tend towards very low throughputs. The designer may not want either or may not be able to afford either and prefer a more middling solution. This is why this work is focused on increasing throughput cost-effectively instead of "at all costs." And even if only one metric was considered important and the other completely ignored, combinations with the absolute highest throughput or lowest cost may still not be desirable. This is because those combinations could be so optimized that any unforeseen issues, such as inaccurate modeling of aircraft performance, fluctuating labor rates, or manufacturing process defects, could cause the actual output to be significantly lower than expected. It could be to the point where said combinations are inferior to more robust options that initially had lower predicted production rates. In particular, given the needed spatial considerations mentioned previously, the absolute highest throughput is likely to include a fabrication and assembly system whose equipment uses up every last bit of space available. This is not necessarily ideal because if less space is available for whatever reason, whether it be due to inaccurate space estimates or extra space being needed for handling, the highest throughput options would suffer large performance setbacks. This is as compared to options that have slightly less than the highest possible throughput by using slightly less than the maximum amount of space, thus being more robust and likely to have some leeway in regard to space. In essence, only selecting the absolute most optimized combinations for either throughput or cost can lead to solutions that fare badly in the other metric or are less than robust. For these reasons, when trying

to get the ideal throughput, the throughput and cost objectives should be kept variable so the designer can see both the maximum/minimum values possible as well as everything in between to be able to make a more informed decision.

Examining the materials area revealed four different factors that are important to cost-effectively increasing production rates: integral fabrication, comparison of different manufacturing processes, space usage, and variable production rate and cost. Performing assembly analysis and making the subsequent tradeoffs on any of these four would create an additional feedback loop relating the impact of the aircraft's choice of material system and subsequent aircraft performance to how much it costs and how quickly it can be made. This better embodies the ideal IPPD approach. To take full advantage of these factors and give even further impact to those additional loops, they all need to be considered simultaneously because they are all able to compound on each other. As an example, the ideal combination of integrally fabricated parts via usage of a composite material could allow one of several manufacturing processes to stand out above the rest by allowing it to leverage its superior space usage, leading to it having lower cost and higher throughput options than other processes and integral fabrication combinations. All four factors can be and are connected. Of these four factors, current aircraft design systems-thinking methods have only been able to incorporate the variable production rate and cost and the different process-comparison factors. No works are currently able to consider all four at the same time.

Despite this, there is still a way forward. Integral fabrication is extremely similar to the alternative subgraphs problem proposed and addressed by Capacho et al. [73]; alternative manufacturing processes are similar to the equipment selection problem put forth by Bukchin and Tzur [74]; spatial limitations are considered by Chica et al.'s time and space problem [75]; and variable throughput and cost trades imply multi-objective optimization, which are covered by a variety of works including Oesterle et al.'s [76]. And what these all have in common is they are all part of Assembly Line Balancing (ALB). ALB is examined more extensively in the non-aircraft design systems-thinking methods than the ones

with aircraft design. Almost all of the former include and carry out line balancing in their frameworks while the latter mention it but do not carry it out, like Butterfield et al. [64], or only analyze an extremely simplified version of it, like with Siedlak et al. [29]. Incorporating ALB therefore has a similar effect to incorporating the previously mentioned ASP. It can better integrate early assembly and aircraft design by establishing additional feedback loops based on the identified factors in the material area, as well as add missing capabilities that were only present in systems-thinking methods without aircraft design. These considerations culminate in Observation 7:

Observation 7: Assembly line balancing and its relevant problem types can be used to better integrate assembly material considerations with aircraft design .

On top of this, the ASP capabilities to be added are on their own only able to operate on production time since ASP does not generally consider resources. Incorporation of ALB alongside ASP provides the resource considerations to turn the production time estimates into production rate estimates. Because of that ALB serves dual purposes and so, when combined with ASP, provides aircraft design systems-thinking methods with significantly more capability, data traceability, justifiability.

The data traceability and justifiability in particular serve as an important complement to the additional capability gained. This is because obtaining higher production rate or lower cost estimates can be accomplished by merely reducing analysis fidelity and making enough assumptions until the desired values are reached; it does not mean that those estimates are necessarily accurate. Incorporation of ASP and ALB eliminates a large number of said assumptions by allowing the designer to see how changing geometry and material variables such as the assembly sequence, degree of integral fabrication, manufacturing process, and spatial constraints affects the final throughputs, costs, and aircraft performance. They provide insight as to why the optimized settings for those variables are indeed the best ones. Not only does this yield better designs through the additional capabilities, but the re-

sults also instill more confidence by making fewer assumptions and removing the typical total dependency on subject matter expert input, which may not even be available. The goal to have better throughput and cost trades is not to just have higher or lower numbers in the results, but also to be able to explain why and be confident that they are close enough to reality to be useful. Adding ASP and ALB analyses will enable that.

No further capabilities will be sought for from the geometry and material areas to infuse with aircraft design systems-thinking methods. The reasons for this are many-fold. The first is that important considerations from both areas have already been identified and addressed via incorporation of ASP and ALB. This denotes a large-enough potential jump in capability that it must be verified before trying to add more, lest the foundations of further additional capabilities be on shaky ground. The second is that incorporation of ASP and ALB to aircraft design systems-thinking methods brings them to roughly equal parity with non-aircraft design systems-thinking methods. This is able to occur because some of the main analyses the former lacked that the latter had are related to ASP and ALP. The disparity between the two classes of methods was one of the main motivators of the overarching research question and so addressing that also plays a large role in addressing the overarching research question. If just ASP and ALB are enough, then no further analyses need be added until a new overarching question is formulated. The third reason is more complicated. A large portion of the remaining high level capability gaps involve analysis that is far outside the knowledge of the typical aircraft designer and so is beyond the scope of this work. Of the capability gaps that remain in scope, ASP and ALB form the first few key steps and so it is imperative to implement them correctly before tackling other insufficiencies. This is further explained below.

One of the main characteristics of this work is that it is essentially an extension of the aircraft design process into the assembly discipline. The designer trying to use this is, subsequently, most likely an aircraft designer of some sort. The scope of this work should therefore reflect this. This means that, on the assembly side, the scope should be limited

to just a single factory or plant. To increase the scope any further requires inclusion of considerations not directly related to factors the designer has control over, and to ignore those extra considerations while increasing the scope is likely to cause gross inaccuracies as compared with reality.

A strong example is multi-factory and plant location selection. Works like those of Tseng, Chen, and Huang [77, 78], and Gomez et al. [25] assume that the only factor concerned with different plants and factories in different locations is the transportation time and accessibility compatibility with one another. Aircraft, however, are physically enormous products. Due to their scale, their manufacture is either focused in one particular factory or plant, or spread huge distances across multiple U.S. states or even in different countries altogether. The latter situations bring up many factors otherwise not considered. One of these is geography, which has a significant effect on travel time and cost and, due to its nature, is unlikely to be generalizable enough to be efficiently modeled such that a variety of different options can be explored. Another is, surprisingly, politics. Factories located so far apart are likely to be in different countries, meaning the relations between different nations can affect the production and movement of the product between factories. This impacts the cost of the aircraft due to differences in material prices and taxes, transportation time due to differences between how nations handle imports and exports, and even the ability to make the product itself depending on whether the nations become hostile in the future. And regarding transportation, the fact that the factories are so spread apart affects the mode of transportation between factories. Due to the physical distance, multiple modes of transportation are available such as transportation by air, rail, road, and even sea, with each mode also carrying its own geographic and political considerations and quirks. Ignoring any of these aspects is liable to doom a production system if it hinges heavily on a location that becomes unavailable or has much fewer benefits than expected as a result of these aspects.

All these factors are generally outside the scope of knowledge of the designer. These

factors and others similar to them tend to most heavily affect assembly analyses that look at scopes beyond a single factory, plant, or small cluster of factories in the same location. The latter can essentially be treated as a single oversized factory or plant. This is why the scope of this work is limited to just a single factory or plant. This requirement rules out other similar assembly analyses, such as logistics and supply chain, on top of the already mentioned multi-plant and factory location. Because these areas will not be examined by the designer, a simplifying assumption is made that all the components and supplies will be available when needed and that, if multiple factories are required, they will be located close enough to each to be treated as a single very large factory. If the areas outside of the desired scope must be examined later on during design, then the presented work considering only a single factory can simply be scaled up to accommodate for multiple factories distant from each other. It is easier to size a single factory in this manner and be able to scale it up later than to start out only being able to size several and be forced to downsize later because one of the locations becomes unavailable.

The remaining scope focusing on just a single factory is best described by Battaia et al. as being part of an assembly flow line [79]. The different analyses in an assembly flow line are: product design, process selection, line balancing, line layout design, and production scheduling. The chronological workflow of those areas, how they relate to each other, and the order in which their analyses are carried out is depicted in Figure 2.5.

It can be seen from Figure 2.5 that, in the context of this work so far, product design refers to the aircraft design discipline, process selection refers to a combination of assembly sequencing and choosing a manufacturing process, and line balancing is self explanatory. Because ASP and ALB are part of the first few steps of the remaining possible in-scope gaps, they must be implemented correctly before any further capabilities are added. To do otherwise could result in situations where expensive later-design changes are liable to occur due to deficiencies that were locked in during earlier design like in the Motivation section. This is the third reason why no other gaps are searched for or capabilities added

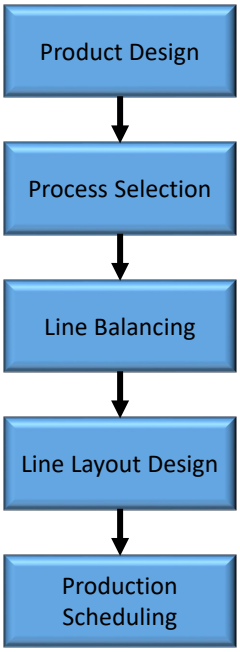


Figure 2.5: Different Assembly Analyses in Assembly Flow Lines and the Relative Order They Are Carried Out, Adapted from Battaia et al. [79]

until ASP and ALB have been incorporated.

2.4 Overarching Hypothesis

To better integrate early assembly design and early aircraft design, a baseline group of methods must first be chosen to improve. Aircraft design systems-thinking methods were selected due to the overall lower complexity of using them as the baseline. To actually carry out an improved integration of the two disciplines, first the areas in which they are able to affect each other were found: geometry, material, and tolerance. The tolerance area will not be used due to the impracticality of obtaining the relevant data for it early in design. The remaining areas were then examined more carefully for missing assembly capabilities that could be used to infuse into the aircraft design systems-thinking methods. These capabilities were identified and it was determined that incorporating ASP and ALB analyses would address them all. The linkage created by performing these analyses using information common to both assembly and aircraft design establishes enough new feedback

loops in the aircraft design systems-thinking methods that the resulting framework is likely able to obtain both higher throughput and lower cost estimates. It is likely to do so with more data traceability and design justification than before as well. On top of this, the addition of ASP and ALB to aircraft design systems-thinking methods gives it most of the assembly analysis capabilities of the non-aircraft design methods. The lack of said capabilities was the main motivator for the overarching research question. As augmenting aircraft design systems-thinking methods with ASP and ALB analyses is believed to be sufficient to address the overarching research question, no further gaps are searched for nor will additional capabilities be added at the present moment. That leads to the overarching hypothesis of this work:

Overarching Hypothesis *If assembly sequencing and more detailed line balancing related techniques are used to more thoroughly integrate the assembly and aircraft design disciplines, then better production rate and cost trades can be made.*

To tie all the material covered in this chapter together and summarize the train of logic that led to this point, how all of the questions, observations, and gaps identified so far, combined with the literature review, led to the overarching hypothesis is shown in Figure 2.6.

The next chapter will review ASP and ALB in more detail to obtain the specifics of what their analyses entail, to ascertain the current state of the art, and to see how they should be combined with aircraft design systems-thinking methods.

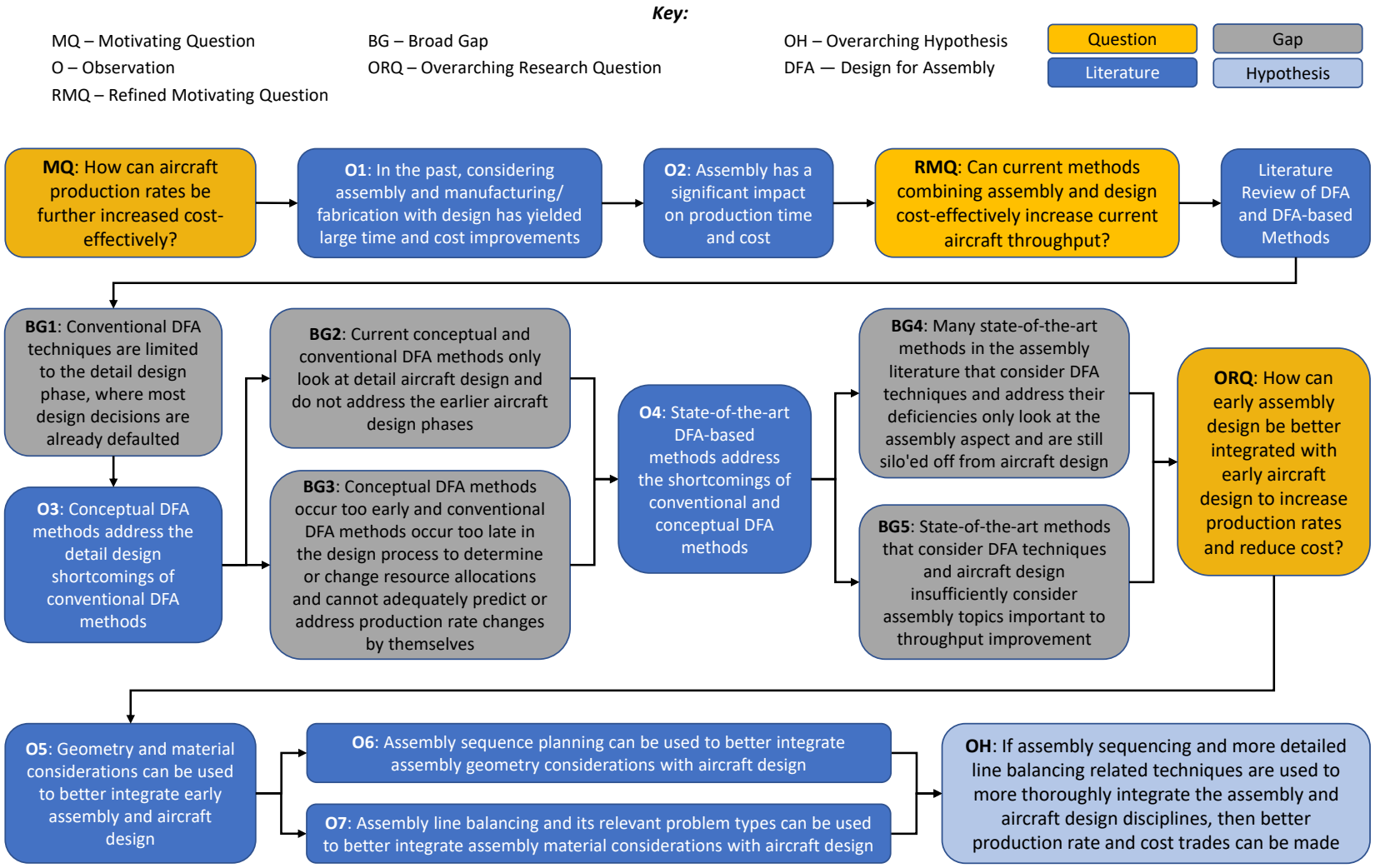


Figure 2.6: Summary Logic Diagram Starting at Motivating Question and Leading Up to and Including Overarching Hypothesis

CHAPTER 3
ASSEMBLY SEQUENCE PLANNING AND LINE BALANCING LITERATURE REVIEW

ASP and ALB have been identified as being able to better integrate early assembly and aircraft design. They both thus warrant further review to examine what how to perform them, what their capabilities are, and what is the current state-of-the-art. The following question is asked to help guide the ASP part of the literature review:

Guiding Question 1: How is assembly sequence planning carried out and what is the current state-of-the-art?

A separate question will be asked later to guide the ALB review.

3.1 Assembly Sequence Planning

ASP is, more formally, an "ordering of collision-free operations" that combines components together "given a geometric description of their positions" in the final product [80]. It is an important step in the more overarching problem of Assembly Process Planning (APP), which also includes assembly tool and fixture planning, resource allocation, and addressing issues regarding tolerances. This importance is due to ASP affecting the efficiency of the entire assembly process, being the locus of information that is used later on to determine the correct line layout, balance the assembly line, and lock in the majority of the cost of the system. An assembly sequence is therefore fundamental to the design itself and affects more than just a design's assemblability and manufacturability [80].

An assembly in an assembly sequence is composed of several different parts: components, component features, the boundaries between components and/or features, and subassemblies. Assembly sequences can be modeled either as a sequence of parts that are put

together or a sequence of operations that puts said parts together [81]. These assembly sequences cannot be made from just any combination of parts or operations, however; they must fulfill geometric feasibility constraints, which state that parts or subassemblies must have a collision-free trajectory during an assembly operation that brings them together; and they must follow precedence constraints, which are constraints on the ordering of the operations relative to each other based on the manufacturing process being used as well as other outside factors. An example of a precedence constraint is riveting only being possible after drilling is performed. The general procedure to create an assembly sequence therefore consists of three steps: define the geometric and precedence constraints, generate all feasible sequences, and selecting the best one, which can be done with various measures of optimality to enable selection of the desired alternative for the appropriate problem [80].

To perform assembly sequence modeling, information about the parts and assemblies being operated on must be obtained. This information must then be presented in a way or form useful to the designer. The next section presents the most notable forms that assembly sequences and their assembly information have taken throughout the years.

3.1.1 Sequencing Representations

Assembly sequencing representations exist to communicate constraints on the assembly sequence, whether those constraint be based purely on geometry or are process-related in nature. They quickly communicate information needed to determine which sequences are feasible and allowed and which are not. There is no one unifying representation for these sequences and the pertinent information contained in their assemblies, however, because there are numerous product architectures and optimization schemes used by and for assembly sequencing. Despite this, there are many common representations that are used again and again that serve as the basis for almost all the others. Only these particular ones will be reviewed.

According to Sanderson and De Mello, there are two major classes of representations,

explicit and implicit ones [82]. The former explicitly allows for an assembly sequence to be generated using the elements shown in the representation, while the latter has this information encoded in it and essentially shows only partial orderings. In essence, explicit representations are able to show all possible feasible sequences up front, while implicit representations primarily show which steps a specific operation can be done before and which it must be done after. The former is less plentiful and, nowadays, less common due to the large number of possible sequence combinations often making it impractical to be represented. It will thus be covered first. Meanwhile, there are numerous versions of the latter due to it being less susceptible to combinatorial explosions, an issue with being forced to represent too many separate combinations. Consequently, the variety of implicit representations is much greater.

The primary representation that is a general superset of all the other representations, explicit and implicit, is the liaison graph, sometimes called the graph of connections or liaison diagram. Created by Bourjault, it depicts the surface contact between mating parts in an assembly and is meant to show that there is a significant relationship between those parts [83]. When depicted as a graph, the nodes represent the individual parts while the edges/hyper-arcs represent the liaisons, or contact connection, between them. The edges tend to be labeled as well. A liaison matrix can be generated from a liaison diagram, with each row or column corresponding to a part. The elements in the matrix are 0 if there is no connection between the two parts and 1 if there is a connection. The diagonal elements are zero because a part cannot be connected to itself. An example assembly and the resulting liaison representations are shown in Figure 3.1, Figure 3.2, and Figure 3.3.

One of the main explicit representations is the directed graph of feasible assembly sequences, which is a graph used to represent the set of all feasible assembly sequences [82]. A graph is directed in the sense that the arcs or edges that connect the nodes have an associated direction with them and point from one node to the next. This is to allow designers to see which way information is being communicated. The nodes in the directed graph of

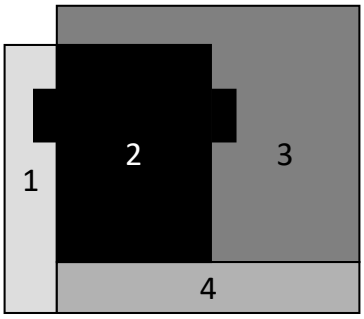


Figure 3.1: Example Assembly for Liaison, Directed, And/Or, and Graphical Liaison Representations

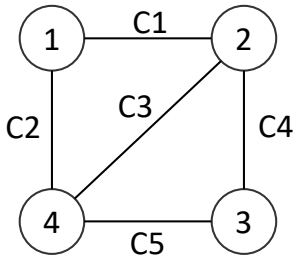


Figure 3.2: Liaison Graph Based on Figure 3.1

$$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

Figure 3.3: Liaison Matrix Based on Figure 3.1

feasible assembly sequences correspond to stable partitions of the set of parts while the arcs represent feasible assembly tasks. In other words, each node is a collection of parts that have and have not been assembled together, with parts that are assembled together being in the same set and parts that are not being in different sets. The arcs, or assembly operations, are used to traverse between different collections where parts are being assembled together and put into the same set. In this way, the first node has as many sets as there are parts, while the last has only one set indicative of the product being fully assembled. This representation is helpful in that all the different sequences can be visualized but, for the same reason, is difficult to create for assemblies with more than five parts. An example is shown in Figure 3.4.

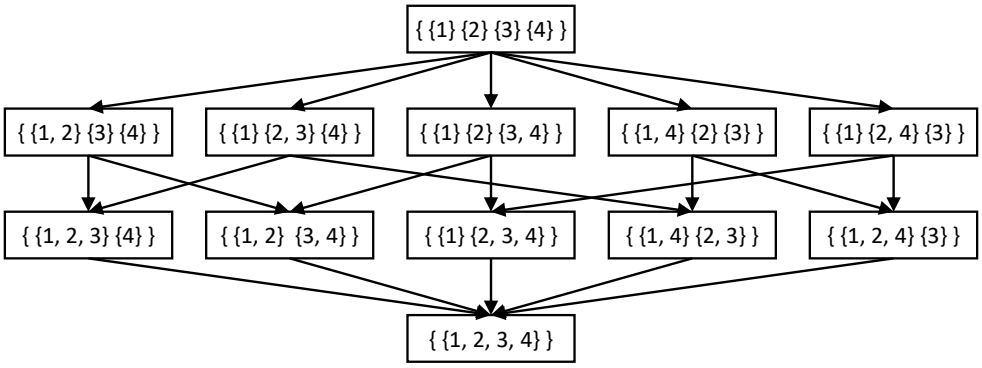


Figure 3.4: Directed Graph of Feasible Assembly Sequences Based on Figure 3.1

A representation very similar to the direct graph of feasible assembly sequences is the AND/OR graph, also created by De Mello and Sanderson [84]. In this graph each node represents a stable subassembly and each hyper-arc represents a feasible assembly operation used to get to that subassembly. The full assembly is typically the topmost node, while the bottom-most nodes are the individual parts. Due to this, tracing the hyper-arcs and assembly operations from the bottom up depicts possible assembly sequences, while from the top down it depicts disassembly operations. The AND/OR graph is overall a more efficient representation than the directed graph because there are no duplicate subassemblies

or assembly operations, leading to much fewer nodes and hyper-arcs if the assembly has more than five parts. However, the AND/OR graph is still liable to have a combinatorial explosion problem if there are too many parts. An example is shown in Figure 3.5.

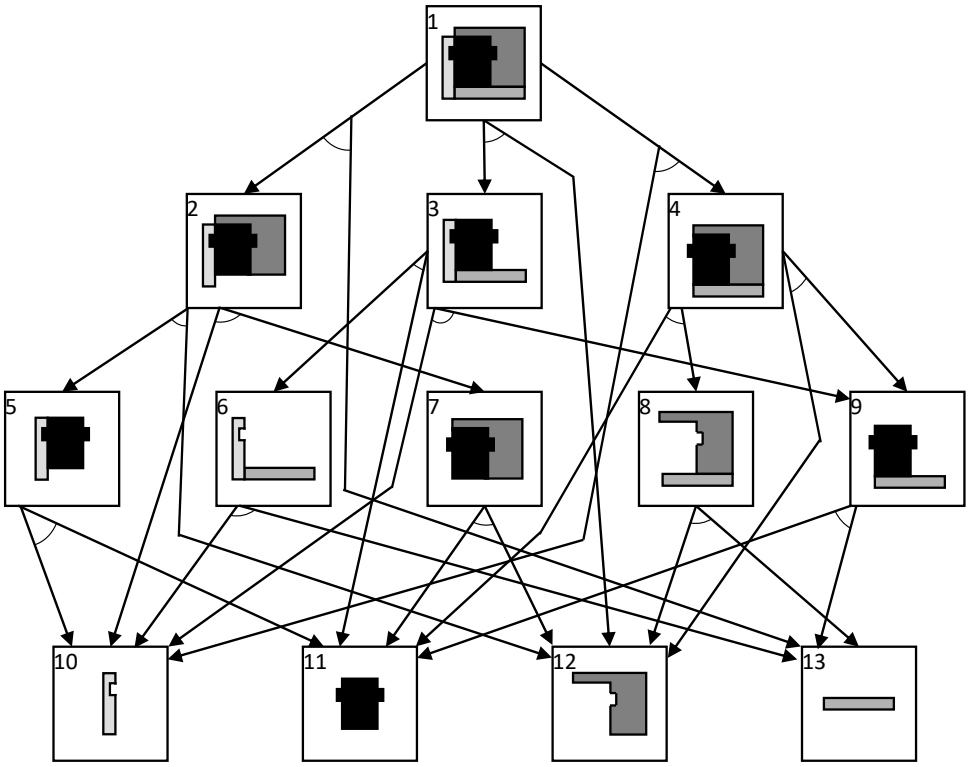


Figure 3.5: And/Or Graph Based on Figure 3.1

Another version of the directed graph, and the last of the explicit representations to be covered, is presented by De Fazio and Whitney, who called it the graphical representation of liaison sequences [85]. It depicts virtually the same information as the directed graph except instead of showing a partition of sets of parts, it instead has True/False values as to whether the connections in each of the subassemblies are present or not. This is done so that the graph is more easily computerized. Additionally, it is also easier to keep track of which parts are in subassemblies since each node's depiction of parts being connected evokes similarities to the AND/OR graph. An example is shown in Figure 3.6.

Perhaps the most important implicit representation of assembly sequences and their

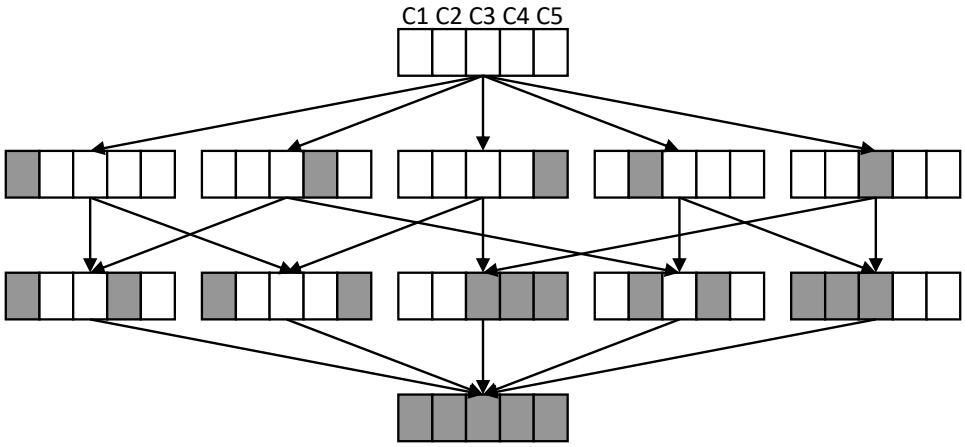


Figure 3.6: Graphical Representation of Liaison Sequences Based on Figure 3.1

information is the precedence graph because, as stated by De Mello and Sanderson, all implicit representations are based on it in some way [82]. Some of the first authors to mention it are Bullinger and Ammer, who described it as essentially a network plan depicting all the necessary tasks and their relative orderings in an assembly [86]. It is called a precedence graph because it shows which tasks and parts precede other tasks and parts. Although originally developed for assembly line balancing, it has gradually become one of the main tools to aid with assembly planning and sequencing. In Bullginer and Ammer's work it is depicted as a graph with circular nodes connected with edges, with each node representing a task and each edge representing the connection between tasks. The graph is read from left to right, the edges not being directed, with a task or part in a node only able to be accessed if all of the operations in the nodes connected to it on the left have already been performed. In the nodes are two numbers, the one on top being the task number and the one on the bottom denoting how long the task will take. Despite its usefulness and ubiquity, few actually use the depiction described by Bullinger and Ammer. Instead, more modern precedence graphs are not limited to the nodes representing tasks, instead being able to represent parts as well. Similarly, the edges thus represent assembly activities. Additionally, for easier reading, the edges are nowadays directed as opposed to undirected

and, because precedence graphs are not only used for line balancing anymore, the second number in the node representing task time is generally not included. The precedence graph can be presented as a matrix as well, as shown by Wang and Liu [87]. In the precedence matrix, the element e_{ij} is equal to 1 if part p_i must be assembled before part p_j, -1 if part p_i must be assembled after part p_j, and 0 if there are no precedence constraints between the two parts. Many variations of precedence graphs and their respective matrices exist, generally developed by authors specifically for their own methodologies and not used by others, and so they will not be further discussed. An example assembly for the precedence graph is shown in Figure 3.7, where parts 5 and 6 must be placed first and where parts 3 and 4 need to be placed before part 2. The resulting precedence graph and its matrix are shown in Figure 3.8 and Figure 3.9.

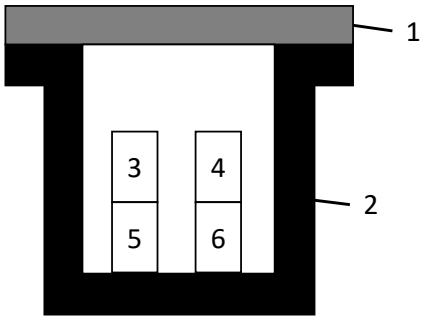


Figure 3.7: Example Assembly for Assembly and Disassembly Precedence Graphs

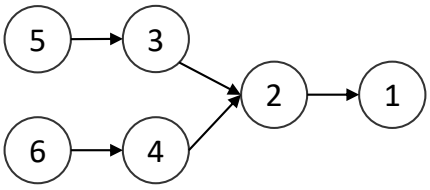


Figure 3.8: Assembly Precedence Graph Based on Figure 3.7

Related to the precedence graph is the disassembly precedence graph; where the former refers to the process of assembly, the latter refers to disassembly. Disassembly is often considered during assembly sequence analysis because it is generally easier to obtain geomet-

$$\begin{bmatrix} 0 & -1 & -1 & -1 & -1 & -1 \\ 1 & 0 & -1 & -1 & -1 & -1 \\ 1 & 1 & 0 & 0 & -1 & 0 \\ 1 & 1 & 0 & 0 & 0 & -1 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Figure 3.9: Precedence Matrix Based on Figure 3.7

ric constraints by analyzing the disassembly of a completed product and then reversing that process to get the assembly information. Disassembly and assembly operations are usually reversible if only geometric constraints are concerned, the parts are rigid, and tolerancing and forces like gravity and friction are ignored [88]. Disassembly precedence graphs, as described by Lambert, are directed graphs where the nodes are disassembly operations and the edges represent precedence relationships [89]. Similar to regular precedence graphs, the tasks in a node can only be performed if all the other tasks with edges pointing into this node are completed. And just like precedence graphs, there are many variations of disassembly precedence graphs as well. As an example, Li et al. created a disassembly constraint graph that uses directed and undirected edges, with directed edges being precedence constraints and undirected edges being disassembly constraints, and with edges being disassembly operations and nodes being parts [90]. They developed this to incorporate both precedence information and disassembly constraint information into a directed graph structure. A demonstration disassembly precedence graph is shown in Figure 3.10.

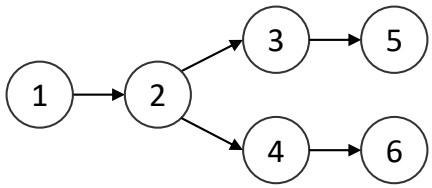


Figure 3.10: Disassembly Precedence Graph Based on Figure 3.7

One of the most common ways to represent the precedence relations in a more mathematical and easily interpreted way that incorporates more geometric data is Dini and San-

tochi's interference matrix [91]. It is an N x N matrix, where N is equal to the number of parts, and works on the assumption that assembly and disassembly operations are reversible. Being an N x N matrix, it is suitable to large assemblies with a high number of parts and numerous edges. Using the assembled product as the starting point, the elements e_{ij} of the matrix are equal to 1 if part p_i is translated in a direction and collides or interferes with part p_j along that trajectory. The matrix elements are equal to 0 along the diagonals or if there is no collision. A respective part's row and column are removed as it is disassembled, and once all the parts are disassembled the operations are done in reverse to obtain the assembly sequence. There are thus at least three sets of matrices, one for each principal Cartesian direction, with the negative direction generally able to be obtained by transposing the matrix of the positive direction. Due to its mathematical and implicit representation and its easily generalized procedure, it is currently one of the most popular ways to represent assembly information, especially when working with assemblies in 3D CAD systems. An assembly created to demonstrate this is shown in Figure 3.11, with the interference matrix in Figure 3.12 shown as an example.

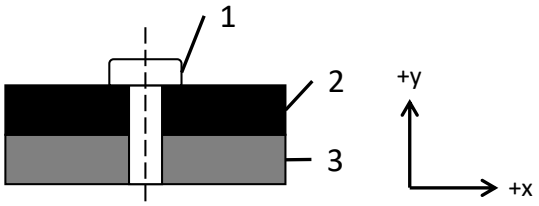


Figure 3.11: Example Assembly for Interference Matrices

A representation using a similar concept as the liaison matrix but with possible assembly sequences being made more overt along with embedding information on how the parts are connected is the connector-based assembly sequence diagram made by Tseng and Li

$$IM_{+x} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \qquad IM_{+y} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}$$

Figure 3.12: Interference Matrices for +x and +y Directions Based on Figure 3.11

[92]. This type of graph is focused on the concept of connectors, such as riveting, welding, bolt-nut-washer, etc., and making them the focus of the graph. In this type of graph, the components are represented as numbers inside circular nodes and are each associated with at most one connector via an undirected edge. The connectors are represented as squares with the identifier C_p^i inside, where p is the order in the sequence and i is the name of the connector. The connectors are connected via directed edges that indicate which connector is to be assembled first. The sequence is determined by performing interference analysis between the different connectors and their components. As an example, say C^1 is the base connector and its components and interference with C^2 is being checked by translating C^2: if C^1 does not interfere with C^2's translation, then C^1 is first in the sequence; if it does interfere, C^2 is now treated as the base and C^1 is translated, and if it does not interfere then C^2 is first in the sequence; and if there is no interference with either order, then both sequences can be done in parallel. An example is depicted in Figure 3.14.

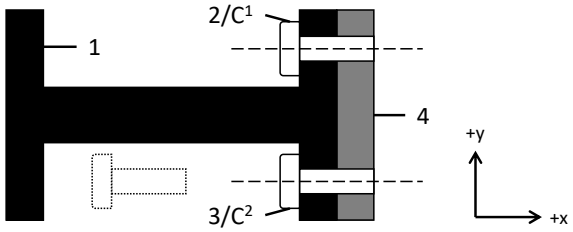


Figure 3.13: Example Assembly for Connector-Based Sequence Diagram, Initial Feasible Direction for Part 3 Denoted

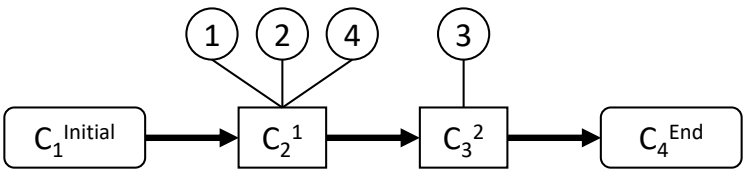


Figure 3.14: Connector-based Assembly Sequence Diagram Based on Figure 3.13

An implicit representation similar to the liaison graph is the assembly cut-set decomposition created by De Mello and Sanderson [93]. This representation is used for determining different cut-sets or partitions of subassemblies so that all possible feasible assembly sequences can be considered. The cut-sets are made along the minimal set of edges that, if removed, disconnect the entire graph. Every time the graph is cut, there can only be a maximum of two resulting sub-graphs. The cut-set representation overall resembles a liaison diagram except that there are dashed lines indicating where cuts are made to separate the graph into sub-graphs. The main purpose of this is to visualize how the sub-graphs are divided up and then to determine whether the sub-graphs, or cut-sets, are feasible subassemblies. An example assembly cut-set based on the assembly from Figure 3.1 is shown in Figure 3.15.

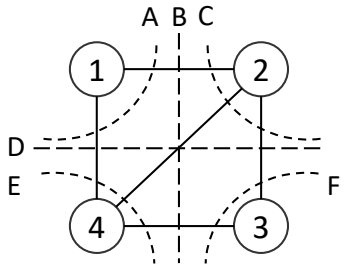


Figure 3.15: Assembly Cut-Set Based on Figure 3.1

Another implicit representation closely related to the liaison graph is the Datum Flow Chain (DFC). Created by Mantripragada et al. it is a directed liaison graph made to aid designers in selecting an assembly sequence while considering tolerancing [94]. Unlike a liaison graph, the arcs between the nodes of a DFC are only solid lines if there is a mate between the parts and are dashed if the two parts only make contact. This is to differentiate between parts that provide mates and key dimensional location information and are therefore more important as opposed to parts that provide contacts and only contribute strength and support to the assembly [95]. The arcs are pointed from the node/component providing the locating information to the node/component being located. Additionally, there is gen-

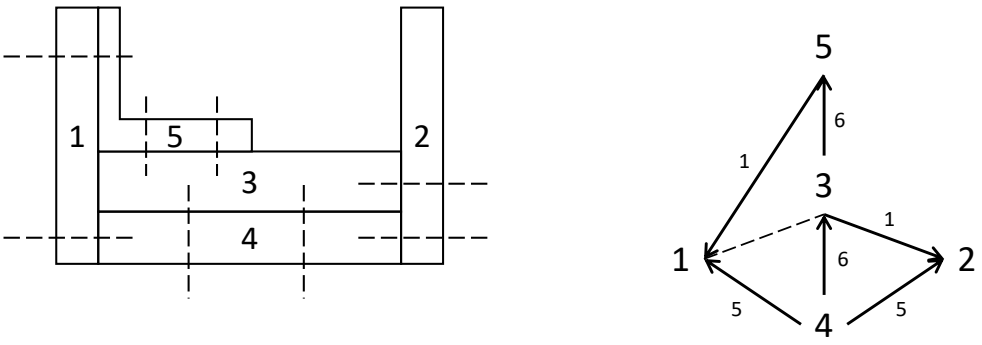


Figure 3.16: Example Assembly and its Associated DFC

erally a number on the arcs to signify the degrees of freedom locked in due to that mate. A part must have the six degrees of freedom defined in order for it to be fully located. The DFC's purpose is to provide a hierarchy on the importance of each part in an assembly sequence based on how well they locate other parts to help designers with tolerancing analysis. An example DFC and its associated assembly is presented in Figure 3.16.

3.1.2 Optimality Criteria

When optimizing for the ideal assembly sequence, the two primary factors that must be considered are the constraints that must be adhered to and the criteria with which the optimization is being performed. The former, whose representations were discussed in the previous section, consist of geometric and precedence constraints and are non-optional: an assembly sequence cannot be implemented if it is not feasible no matter how ideal it otherwise is. The latter, however, vary according to the different topics being studied and the assemblies being put together. Indeed, as stated by Goldwasser, there is no single "unanimous cost measure that all people will hope to optimize" [96]. It is thus useful to define the different optimality criteria that are often used.

The most commonly used criteria are the assembly direction change, assembly tool

change, and assembly type change [81, 97]. The former refers to how many times the components in an assembly are put together in different directions, such as inserting one component in in the +x direction and then inserting the next in in the -y direction. Better sequences are assumed to have fewer direction changes with the justification, according to Goldwasser, being that fewer direction changes allow an assembly robot to be less complex by having fewer degrees of freedom, thus costing less [96]. It is more efficient as well since the robot is able to move around less if there are fewer direction changes. Likewise, more efficient assembly sequences involve less tool changes because every tool change requires time to perform. Sequences with numerous tool changes thus use up unnecessarily large amounts of time. An assembly type change refers to the changing of assembly operations between parts, such as switching from riveting to screwing between steps. Similar to tool changes, type changes involve switching out different equipment and excessive type changes waste large amounts of assembly time. Therefore, better sequences will have fewer type changes.

Other important criteria include reorientations, stability, fixture complexity, and part visibility/manipulability. Reorientations, as described by Goldwasser, are similar to but distinct from direction changes in that the latter involves the robot assembly arm changing directions while the former refers to repositioning the entire assembly so that the robot arm does not have to change directions [96]. Lower numbers of reorientations are better. Assembly stability refers to the concept of the assembly or subassembly being put together having the ability to support itself during that process without the aid of fixtures or other supports. It is desirable because such fixtures or supports tend to be incredibly expensive and time consuming to create. As a result the desire with the stability metric is to maximize it. Related to stability is fixture complexity. Where stability focuses on the part being able to hold itself together, fixture complexity looks at the fixture needed to perform the holding. Complicated assemblies that cannot locate themselves will need equally complex fixtures to accommodate them. Airplanes in particular need especially complicated fixtures because

the majority of their parts can only be located relative to each other and must otherwise be set into their initial positions via a fixture. A less considered criterion is in regard to manipulability. As mentioned by Wolter, it is the ability to easily and ergonomically access and perform the desired assembly operations on the part [98]. In this way it is related to visibility, which impacts the laborer's ability to see the part as they operate on it. Higher visibility and manipulability will allow the laborer or assembly robot to be more comfortable with the assembly operations, increasing their speed and also preventing injury and wear and tear. It is likely that these metrics are not used often due to being quite subjective, with their more objective counterparts in tool accessibility being a combination of both a geometric and precedence constraint as opposed to an optimization criterion. Regardless, it is best to maximize manipulability and visibility.

Other metrics that are sometimes used as optimization criteria include minimizing assembly angle, minimizing part count, and minimizing energy consumption, with many more created for their particular author's specific topics and applications.

3.1.3 Graph-based Methods

There are currently numerous techniques to assist the designer in obtaining the optimal assembly sequence with several different ways to provide the information needed to start the process. Traditionally, these techniques consist of querying the user for information about the assembly to generate graphs of all feasible assembly sequences. These graphs are then traversed using formal graph search methods, by pruning prospective undesirable sequences, or a combination of both. Graph searching techniques were already relatively well developed when the first assembly sequence planning methods were made due to being used for assembly line balancing beforehand. Due to this, graph-based methods for assembly sequence planning generally focus on creating the graphs.

Bourjault is the author of one of the first methods to generate all possible feasible sequences [83]. His method consists of the user creating a liaison diagram for the assembly

and then asking the user a series of yes or no questions about the liaisons to create precedence relations. The two main questions asked for each liaison are shown below:

- Can liaison L_i can be done before the other liaisons $L_j ... L_k$?
- Can liaison L_i can be done after the other liaisons $L_j ... L_k$?

This sums up to a total of at least $2l^2$ questions that the user has to answer, with l being the number of liaisons present. The precedence relations established from these questions are then used to generate the list of feasible assembly sequences.

De Fazio and Whitney argued that the number of questions asked in Bourjault's method is excessive, with some of the questions becoming increasingly harder to answer as the user goes through them [85]. De Fazio and Whitney thus created their own method. Like Bourjault's, their method requires a liaison diagram to be made. However, the questions asked in their method are slightly different and presented as follows:

- What liaisons must be done before liaison L_i?
- What liaisons must be done after liaison L_i?

This slight difference in not repeating the same question over and over for the same liaison reduces the number of questions asked from at least $2l^2$ to just $2l$. The answers to these questions, similar to Bourjault's method, are used to establish precedence relations which De Fazio and Whitney encode into graphical representations of liaison sequences. From there, the optimal sequences can be found using graph search methods or utilizing the optimality criteria as filters.

De Mello and Sanderson later wanted to make sure that they had indeed generated all possible sequences and done so correctly and created the concept of the assembly cut-set [93]. In this procedure the assembly is decomposed into two subassemblies at a time, which are represented as cut-sets being made in the assembly's liaison diagram. Each set of subassemblies is checked for task-feasibility, making sure the subassemblies can be

feasibly joined to create the assembly, and for subassembly stability, making sure that the subassembly does not immediately fall apart if only one of its constituent parts is fixed in place. Each of the subassemblies are then decomposed further into smaller diassemblies and the process repeated for all possible subassemblies until only a single part remains in each subassembly. The final results are represented in an And/Or graph, which can be used by the designer to find the optimal assembly sequence.

De Mello and Sanderson's cut-set procedure was improved on and made more formal in the form of the assembly cut-set method by Baldwin et al. [99]. A liaison diagram is first created from the assembly and all possible subassemblies made by taking all possible part combinations. These are converted into liaison sub-graphs, which are verified to see if they are connected in the actual liaison diagram. Cut-sets are afterwards made by creating sets of two subassemblies, consisting of the parts of the first subassembly N_i, the parts of the second subassembly N_j, and the liaison connections between them. Precedence relations from this point are determined by asking the following question:

- Can the parts in the subassembly N_i be disassembled from the parts in the subassembly N_j?

The generation of precedence relations is made easier by taking advantage of two rules created by Baldwin et al.:

1. Superset rule: adding components to a set of two components or subassemblies that cannot be feasibly combined due to interference between them will not change the result.

2. Subset rule: if two subassemblies can be feasibly combined, removing components not related to the mechanism of their union will not affect the result.

Utilization of these rules allows for a single user answer to a question to also subsequently answer many other questions down the line, removing a lot of workload from the user. All of the precedence relations gathered are outputted in the form of graphical representations

of liaison sequences, which the designer can use to refine their ideal assembly sequence search.

Laperriere and ElMaraghy developed a combination graph-based and 3D CAD oriented method that differs from the previous methods by not exhaustively generating and searching through all possible sequences first [100]. This method, called the Generative Assembly Process Planner (GAPP), instead combines the steps of generating, evaluating, and eliminating of assembly sequences all into one process. It does this by first reading in a 3D CAD model of the assembly and generating the assembly's graph model, stability sub-graphs and freedom matrices, which are closely related to interference matrices, from the model. Cut-sets in a manner similar to De Mello's method are then created. However, these cut-sets are made taking into account geometric constraints from the freedom matrices, stability constraints from the sub-graphs, and accessibility constraints, as well as optimality criteria such as number of reorientations and stability. The cut-sets generated are thus essentially already evaluated. The all-inclusive integrated generate-evaluate-eliminate process involved in this method therefore makes it very efficient.

One of the more unique ASP works is Panhalkar et al.'s work [101]. Unlike most ASP works they optimized for assembly time using actual task times for assembling, which they obtained by using MOST. Theirs is one of, if not the only, works in the ASP literature to use such a technique or something similar to obtain and model assembly times for sequencing as opposed to building the product to get the assembly times or using a time analogue such as number of tool changes. Panhalkar et al. can thus actually get time estimates without having built a product yet. Their work focuses on the automotive manufacturing industry and has a use case revolving around a car door subassembly. They used a dynamic tree-based algorithm to systematically search through their precedence graphs to obtain sequences with the minimum assembly time.

Despite the importance of these graph-based techniques in laying down the groundwork for assembly sequence planning as a whole, one of their primary weaknesses is that the

graphs quickly become untenable in assemblies with large numbers of components. These methods also require asking the designer a large number of questions that can take many hours, causing many to turn towards more computer and CAD-focused methods. Additionally, there are only so many ways to generate the set of all feasible assembly sequences. Eventually, it becomes more attractive to look at methods to actually traverse through those sequences, at which point there is significant overlap with assembly line balancing and its related techniques. An example is with Lambert, who addressed the disassembly problem instead of the assembly one [89]. In his work he used a disassembly precedence graph and treated traversal through said graph as a Traveling Salesman Problem (TSP) and solved it by iteratively using binary linear programming. Similarly, another example is with De Tian and Wang, who focused on a modified assembly precedence graph where they attached weights to the directed edges based on optimality criteria such as reorientations, stability, and number of tool changes [102]. These weights are obtained using a fuzzy analytical hierarchy process method, and they search through the resulting graph with a minimum spanning tree-based method. Overall, modern graph-based methods for assembly sequencing are comparatively rare when set alongside other methods.

3.1.4 Connector-based Methods

The primary, base representation of an assembly for sequencing has generally been its liaison diagram, but it is not the only one that exists. A somewhat less used but still well known representation is the connector concept, made by Tseng and Li [92]. They argued that the fundamental building blocks of an assembly consist of not just its components but its connectors as well, connectors being the mechanical linkages that actually tie the components together. Additionally, Tseng and Li argued that, during conceptual design, the geometric modeling and information often used tend to be unavailable. Connectors can thus fill in that gap and serve as an excellent bridge between the functional and physical model. Due to their importance, they state that connectors should be included when generating

assembly sequences and so create an algorithm to generate their connector-based assembly sequence representation. These sequences are both single path and multi-path, essentially a precedence graph for connectors.

Despite the promise in this method, works that use connectors with graph-based representations are rare. This could be because the exact precedence relations between connectors are still not very well defined or because stability of the subassemblies is not considered much [103]. For one reason or another, the majority of works that use connector-based methods also combine it with other techniques. A notable class of methods that sometimes incorporates connectors are knowledge-based methods, which are covered in the next section.

3.1.5 Knowledge-based Methods

Knowledge-based methods, sometimes also known as Expert Systems (ES), are a class of methods focused on codifying knowledge and experience from experts into, typically, a computer system so that it can be reused in the future for similar problems. More broadly, according to Leo Kumar it deals with computer programs that "possess [their] own decision-making capability to solve a problem of interest" [104]. The field is thus incredibly broad due to the decades of manufacturing experience, the frequency with which fabrication and assembly methods are reused, and the sheer breadth of different fabrication and assembly operations available. Works in this literature can range from discussing functional model representations of assemblies, to how to select the correct tools and processes, to how to specifically design a part symmetrically or asymmetrically to be able to machine it. It is thus impractical for a comprehensive review of the different techniques available. Instead, only a brief overview of the methods pertinent to assembly sequencing will be covered.

The works related to assembly sequencing belong to the larger area of Computer-Aided Process Planning (CAPP). The area tends to be divided into two types of approaches, vari-

ant and generative [105]. Variant approaches involve analyzing the part being designed for similarity to a known, previously used part, after which a stored process plan based on that information is leveraged with the appropriate modifications. Generative approaches do not store process plans but rather information on rules regarding key information about part processes, geometries, and materials, which are then applied to the part being designed to synthesize and create a new process plan. The primary input to both approaches is geometry information, typically from CAD, causing a large amount of overlap between knowledge-based and 3D CAD feature-based methods [104]. The knowledge in the knowledge base itself is typically stored as "facts, rules, heuristics, [or] procedures" [104]. Sub-methods for knowledge-based systems therefore include rule-based systems, frame-based systems, object-oriented systems, and case-based reasoning [104]. Rule-based systems operate on an "if-then" basis where if a condition is meant, then a specific task is performed. Frame-based systems use the context of the situation to determine what actions should be performed. Object-oriented systems classify parts or operations as objects, with each class or type of object having various rules, properties, or attributes associated with it. Case-based reasoning tries to find the "case" that is most similar to the part being analyzed and leverages that case. All these different sub-methods therefore make the field of ES even wider.

Kusiak was one of the earliest users of knowledge-based methods for assembly sequence planning. In his work, he split process planning into eight phases: interpret part design data, select machining process, select machining tools and fixtures, machining optimization, decompose machinable volumes, select machinable volumes, generate precedence constraints, and generate sequence of machining volumes [105]. A knowledge base was employed in each step to solve the qualitative sub-problems while optimization routines solve the quantitative sub-problems. Example usages of the knowledge base include applying rules and pattern recognition to identify the part's features and then using those features to select the appropriate machining processes.

Dong and Hu created a framework using knowledge bases that is able to output more than one assembly sequence, arguing that there is no guarantee of optimality if a knowledge-based system only provides one output [106]. Their work looked specifically at machining operations. Part features are first identified and feasible machining operations for the identified feature assigned using the knowledge base. Afterwards, the knowledge base is used to assign physical equipment to the operations, which also have cost and tolerance relations tied to them. The designer-specified desired tolerance is combined with the cost and tolerance relations to obtain the optimal sequence.

Swaminathan and Barber proposed a case-based reasoning methodology to reuse previous process plans [107]. The methodology breaks down the inputted assembly into basic configurations there is previous knowledge of so that the knowledge base can build new assembly sequences. The parts are first used to match up with previous plans in the knowledge base. A plan is a blueprint on which parts form which assemblies, what order everything is assembled in, and what trajectories they all take during that process. Adjustments are made to the retrieved plans to match the requirements, after which they are combined. The resulting precedence graph can be searched through for the best sequence. This implementation allows for the creation of new plans that are not explicitly part of the knowledge base.

Kashkoush and ElMaraghy used a methodology similar to Swaminathan where a common underlying structure is combined with a knowledge base [108]. Their method differs in that they are generating the knowledge base to use for future assemblies as opposed to taking advantage of an already existing knowledge base. Kashkoush and ElMaraghy implemented a Mixed-Integer Programming model to create an assembly sequence tree from multiple other assembly sequences, which themselves come from different products in a product family. This resultant assembly sequence tree is essentially the "optimal consensus" of all the other trees in this product family and can be leveraged if a new product in the product family is being developed.

Zha et al. used a more holistic approach akin to the PLM techniques and opted for an assembly-oriented design method [109]. The method starts with a conceptual design functional decomposition, with the resulting functions and sub-functions matched in a knowledge base with pre-existing physical assembly structures that could be used to carry them out. The assembly structures are stored in an object-oriented knowledge base. A case-based reasoning knowledge base is then used to make the assembly structure more physical by providing information on how the different structures can be connected. Assembly sequences can then be generated, followed by leveraging the knowledge base again to generate a model with geometric features. At this point, the extra geometric features allow for the rest of the constraints to be applied and the assembly sequences and their assemblability evaluated. This knowledge base assembly-oriented design method is thus able to consider the entire design process of the product, though as implemented it is primarily for smaller assemblies like sets of gears. Additionally, it is unique in that it uses an object-oriented knowledge base, allowing for efficient storing, matching, and using of knowledge.

Yin et al. used a connector-based structure [110]. Inputs from the user are used to decompose the assembly into a set of primitives based on the concept of connectors. The knowledge base is used to obtain and reuse similar assembly sequence plans for each primitive, obtain stored plans similar to the primitives but that require changes, or generate new plans via geometric reasoning. The different plans for each connector-primitive take the form of assembly precedence graphs and are afterwards merged together. Although the method is efficient, the merging process only considers local optimality, i.e., only the merging between individual primitives is considered. There is no guarantee that the final product will be optimal according to the user's criteria.

Li et al. used a connector-knowledge-based method as well but focused more on a rule-based system instead of case-based reasoning [111]. In their method, the assembly is broken down into its constituent components and connectors and the rule-based knowledge base provides disassembly precedence constraints and relations for the connectors. These

relations are used along with geometric reasoning to build a disassembly constraint graph, which is a modified disassembly precedence graph, that can be traversed through. Li et al. were thus able to apply knowledge-based techniques to disassembly as opposed to assembly in the previous works.

Hsu et al. have a unique method that builds a knowledge base via neural networks and leverages the Taguchi method to optimize an assembly sequence [112]. The assembly's 3D CAD model is first used to create the assembly precedence diagram. The Taguchi method is used to perform a design of experiments where variables such as each part's weight, volume, and features are varied. The assemablability of the resulting assembly sequences is recorded and the subsequent data used to train a back propagation neural network that serves as the knowledge base, which can find the optimal sequence. In essence, the method revolves around creating a kind of surrogate model and optimizing the assembly sequence with it. However, the degree of difficulty in representing all the parts is non-trivial, and this method has only been able to be applied for very simple assemblies such as a toy motorbike and a brushless fan.

Mishra and Deb utilized genetic algorithms in their implementation of knowledge bases [113]. Information regarding the assembly such as its parts dimensions, weight, shape, and contacts are used by the rule-based knowledge base to select the proper tools and grippers for the task. A genetic algorithm then uses precedence information to obtain the optimal sequence in terms of reorientations and tool changes. Their method is overall simpler than the others and does not use knowledge bases to actually generate the assembly sequences but instead only to assist in that process.

Qiao et al. used an ontology-based method in order to generate and evaluate assembly sequences [114]. Ontologies are essentially object-oriented knowledge bases except that they use a common modeling language so that knowledge can be reused, shared, and analyzed. This language is typically the Ontology Web Language (OWL). Qiao et al. specifically enhanced their ontology to be able to read in and process geometric models for the

purpose of assembly sequencing, which had not been done before. They then further improve the process by also incorporating the Semantic Web Rule Language (SWRL), which is a rule-based reasoning engine that allows the ontology to make inferences, i.e., recognize implicit relations and constraints between different parts. This allows the ontology to create feasible assembly sequences on its own without user input. Zhong et al. expanded on this work by implementing an algorithm to allow the sequence generation to be done automatically [115]. This also enabled them to use optimization algorithms to search for the ideal assembly sequence, which they demonstrate by using an Ant Colony Optimization (ACO) to find the ideal sequence for multiple assemblies. Ontologies are therefore currently at the forefront of knowledge-based methods for assembly sequencing due to their ability to allow the domain knowledge to be shared.

Knowledge-based methods are undoubtedly useful due to being able to essentially act as subject matter experts without one necessarily needing to be present. However, because of this they also share a similar weakness, which is that the knowledge they provide is of questionable value if there is little to no past data, a particularly pertinent issue when new technologies and designs are being formulated. Additionally, all of the methodologies covered, and the majority of knowledge bases in the literature, are architecture agnostic. That is to say they can recognize that a particular assembly is an airplane wing, perhaps even be able to differentiate between a commercial versus a fighter airplane wing, but are not useful in sizing a specific commercial airplane wing. They are therefore generally not specialized enough to size a particular aircraft, or even any particular product in any architecture, hence why nearly all of them utilize geometric reasoning to some extent as a backup plan. The geometric reasoning information, nowadays, tends to be obtained from 3D CAD models of the assemblies. Because this step is so ubiquitous, the next subsection will cover sequencing techniques that, in particular, utilize feature extraction from CAD models.

3.1.6 Feature and CAD-based Methods

Feature-based methods revolve around extracting part and assembly features and other geometric information from CAD models and then using that information to generate and evaluate assembly sequences. A multitude of techniques can be used for the evaluation step, ranging from graph-based to connector-based to metaheuristic-based techniques. There is thus a large amount of overlap between those techniques and feature-based methods. However, not all metaheuristic or graph-based methods extract their data from CAD models, meaning this class of methods cannot be cleanly distributed amongst the others. Due to these reasons, this subsection will only cover those works that do not quite fit under the banner of the other mentioned methods or that are much more focused on the feature extraction.

As described by Neb, there are three general approaches to extracting feature information from a CAD model for assembly sequencing: external, internal, and ontology-based [116]. External approaches involve reading and extracting geometric information from the assembly's CAD model's input file. This approach tends to involve neutral file formats like STEP or IGES and, although convenient, is liable to have data losses or missing assembly features during the conversion of the original CAD model's data format to the neutral file format. Internal approaches involve directly interacting with the CAD program through its specific Application Programmable Interface (API). Although there is much less data loss, it is unique to each specific CAD software and is not generalizable. Ontology approaches store CAD model information and are easily extracted, but in order to acquire that information need the usage of one of the other two approaches and so are not really an independent approach. The majority of the following discussed works do not mention how their feature information is extracted unless they specifically focus on one of the approaches.

Lee and Ko imported CAD assembly information to the computer by creating a hierarchical tree of mating conditions between each component [117]. The hierarchy of different mates and components are used to generate the assembly sequences, with a swept-volume

approach being used to check sequence feasibility. Wolter created a tool with emphasis on robot path planning [98]. Here, the assembly's CAD model is used to create an assembly constraint graph, which provides information for possible insertion directions for the robot as well as precedence constraints and is traversed for assembly criteria like number of tool changes and direction changes. Dini and Santochi created a software system that generates sequences from a CAD model by automatically detecting possible sub-assemblies in the model [91]. It checks all the parts for interference to create an interference matrix, then uses a contact matrix and a connection matrix to decompose the interference matrix into contracted interference matrices. The contracted interference matrices represent the parts in a subassembly, which are checked for interference to determine the assembly sequence. Lin and Chang created a tool called 3D-MAPS that takes in a 3D model and automatically infers detailed geometric information from it, such as which parts are mated together, what surfaces and directions they are mated along, and along which directions will there be collisions with other parts [118]. This geometric information is combined with precedence relations to create hierarchy graphs that are used to generate the assembly sequences in a two-step process. Wilson and Latombe created the concept of Non-Directional Blocking Graphs (NDBGs), which are generated automatically from CAD models and utilize similar geometric information as those made from 3D-MAPS [119]. NDBGs, on top of providing geometric constraint relations, can also be used to provide possible disassembly and assembly sequences. Romney et al. further expanded the usage of NDBGs by creating a dedicated tool for it that generates assembly sequences, called the Stanford Assembly Analysis Tool, while also improving the concept of NDBGs [120]. They did this by turning it into a sphere instead of a circle for 3D assemblies and by only showing cones of translation directions where there is no local interference, unlike the original NDBGs.

Many feature and CAD-based methods, due to the availability of detailed geometry, often utilize extensive algorithms to check the interference relationships between the components in the models. Eng et al. have a method where the degrees of freedom of the parts'

features and its mates are used to generate precedence relations [121]. This method utilizes a bounding box algorithm to determine geometric feasibility when creating the sequences, which are then optimized via criteria like unidirectionality of the insertion direction and manipulability. Hu et al. similarly based their assembly representation on the degrees of freedom of the parts' features and its mates [122]. However, their method does not use a bounding box method to determine interference but instead uses a custom algorithm to perform a multi-step, multi-direction interference check for each part for local geometric feasibility. The global geometric feasibility is determined by projecting the part in the principle Cartesian coordinate directions, and the resulting feasible assembly sequences are optimized based on the assembly/disassembly time. Su created an approach called Geometric Constraint Analysis that, after it takes in the 3D CAD model, generates constraint directions for each part [123]. Constraint directions are directions along which the part cannot be translated and, if it extends to all possible directions for a single part, means that part is completely constrained and in a constraint assembly state. The constraint directions of each part and their relationships with each other are used to create assembly precedence relations that, by proxy, also include geometric feasibility constraints. Su et al. later expanded their approach to include more criteria such as stability, number of orientations, and insertion direction, which revolves around only inserting components from the top down [123]. These methods are all so focused on geometrical constraints that it is possible to mix and match their different algorithms and create a separate sub-discipline focused only on geometric reasoning.

An example of such a geometric reasoning focused approach is with Zhang et al. who, in an effort to improve the speed of automated assembly sequencing, devised a method with overlapping boundary boxes that can quickly determine the interference matrix, and therefore the geometric constraints, for CAD models meshed in very fine detail [124]. Many other such works focused on geometric reasoning follow the same route and further detail on these different algorithms will be provided in another section.

In a somewhat similar style to Zhang et al., Wan et al. used the meshed models of their assemblies as input instead of a typical boundary representation model [125]. However, their method also requires the meshed model of a robotic hand as input. This is because, instead of determining just interference between parts, the graspability and stability of the assembly is considered as well due to their effects on the assembly's handling. Their method uses these criteria along with what are essentially NDBGs, but without the precedence graphs embedded, to generate the proper constraints and select the best sequence.

Pan et al. did not look at using meshed model inputs due to their quality depending on the defined mesh but instead look at using STEP files, which are standardized inputs that can define an assembly in a CAD program [126]. They used a JAVA-based standard interface named JSDAI to first read in the assembly and determine the interference-free matrices by projecting each part onto a plane and seeing if any parts overlap. Although this method only works for simple geometric shapes such as prisms and axis-aligned cylinders, it is significantly more computationally efficient than the previous swept-volume and multiple interference detection algorithms. Pan et al. afterwards used this information to optimize for the sequence with the fewest reorientations [127].

Mantripragada and Whitney employed a unique variation of graph-based methods that also considers the 3D CAD model and features of the assembly [94]. Their method specifically looks at how tolerances are built up as parts in an assembly are put together. Thus, instead of looking at contacts between parts as inherently significant, they only consider mates and mating features between parts that also help locate them as important. This idea is expressed in their creation of the Datum Flow Chain (DFC), which is used to help them design sequences that are able to easily deliver Key Characteristics (KCs) for the assembly. KCs are significant geometric features that require very tight tolerances and are typically provided directly as customer requirements. The DFCs are first used to determine what parts belong in a subassembly, with subassemblies only being allowed to consist of parts that can all internally lock in each other's six degrees of freedom with exception for the

"base" part. The optimal sequence is obtained by organizing these subassemblies such that KCs can be delivered by as few subassemblies as possible to avoid stacking up too many tolerances. Due to the emphasis on degrees of freedom and constraining and locating parts, DFCs can be used to design fixtures as well. Zhanlei et al. further expanded on the usage of DFCs for assembly sequence generation by automating it through incorporation of a precedence matrix representation of a DFC [128]. The method is particularly useful for tolerancing purposes, but its main goal leans more towards helping designers create specific assembly features to aid in the tolerancing of an assembly as opposed to finding the optimal sequence.

CAD and feature-based methods allow for relatively easy and straightforward generation of geometric constraints. However, as shown in the covered works, beyond the generation of the constraints their optimization methods vary considerably. Additionally, the ease with which the geometric constraints creation can be automated has also allowed some of the analyzed assemblies to become very large. Such large assemblies are generally infeasible to tackle with the typical graph-based methods mentioned thus far and are also a challenge even for knowledge-based methods. The next section therefore focuses on soft computing methods, or methods that use meta-heuristics, that are currently used to tackle assemblies with a large number of parts.

3.1.7 Soft Computing Methods

A heuristic is a rule-of-thumb to be followed when attempting to solve a problem. A meta-heuristic is an overarching guideline that directs how heuristics are to be employed in solving the problem. Meta-heuristics in the context of optimization are algorithms that search for the solution to a problem that often has many constraints. They tend to not guarantee a perfectly optimal solution like more formal optimization techniques, graph-based techniques being an example, but trade this off by being able to arrive at a "good enough" solution in significantly less time. Meta-heuristics, also called soft computing methods,

have thus become the most-used method to optimize an assembly sequence because it was realized that the assemblies simply became too large to feasibly tackle with conventional methods like graph searching. Examples of soft computing methods are simulated annealing and genetic algorithms. A general flow chart of how soft computing methods work is shown in Figure 3.17. This section will cover the most significant and commonly used of these methods: neural networks, simulated annealing, genetic algorithms, ant colony optimization algorithms, and particle swarm optimization algorithms.

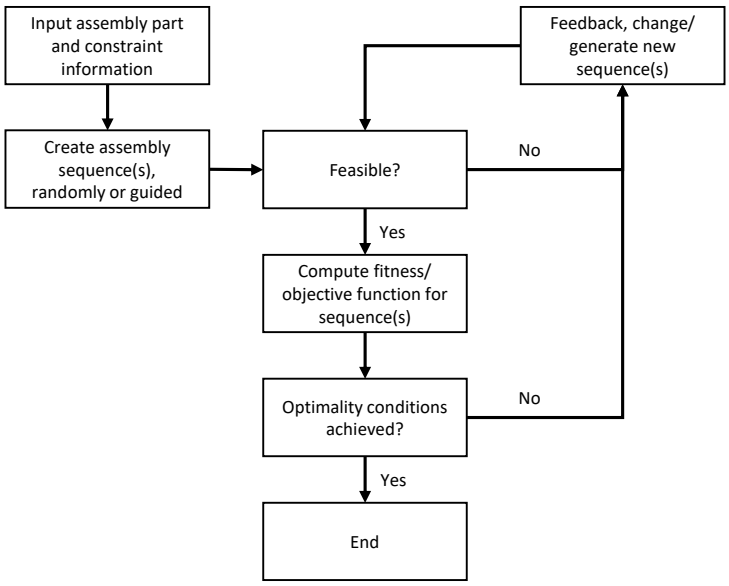


Figure 3.17: General Process Flow of Soft Computing Methods

3.1.7.1 Neural Networks

Neural networks were one of the first soft computing methods to be developed. They are based on the ability of many simple neurons in the human brain collaborating together and working in parallel to recognize complex patterns and be able to produce new, equally complex, similar patterns. Since many assemblies have similarities in their geometric and precedence constraints, a neural network can be trained to generate optimal and feasible sequences by recognizing those patterns on its own. In this way it is similar to knowledge-

based methods but often requires much less expert advice. Chen and Pao were one of the first to use neural networks on assembly sequences, which in their case is utilized to recognize design features provided by the designer and output similar assemblies and sequences to inspire the designer [129]. The types of features recognized by their neural network include component types, their associated tool types, and assembly direction, on top of more conventional features related to geometry and topology such as number of faces and volume and shape.

Hong and Cho more explicitly used their neural network to generate an optimal assembly sequence [130]. In their method, the assembly is inputted into an expert system, which outputs assembly constraints and costs into the neural network. The neural network calculates an energy function based on those outputs to arrive at an ideal sequence. The cost is a function of the assembly's degree of instability as well as the number of direction changes. One downside noted with using neural networks in this manner, however, is that the network's outputs sometimes do not constitute an actual assembly sequence due to improper calculation of the stability for the energy function.

Chen et al. used a back propagation neural network in their three step method [131]. The assembly's CAD model is used to generate a precedence diagram, a penalty matrix used to determine the difficulty of assembling the parts, and a relational model that indicates how the parts are connected. These, in turn, are fed as inputs into the neural network, which outputs the ideal assembly sequence based on what it was trained on.

Although neural networks are an important type of method often used for machine learning and artificial intelligence, they are rarely used in current assembly sequencing analysis because it is difficult to train them to recognize assembly constraints. In other words, a large number of their outputs are likely to have traits of the ideal sequence but, due to a mis-ordered task, be rendered infeasible. The order-centric nature of assembly sequence thus does not match well with neural networks.

3.1.7.2 Simulated Annealing

Simulated Annealing (SA) is another soft computing method developed relatively early on. It mostly behaves the same way as a local optimization algorithm in that, after the initial solution is presented, it perturbs the design variables to try to obtain a better solution and move downhill to the optimal one. It differs in that there is a chance for solutions that move uphill to be allowed to try to escape from local minima. This chance is dependent on the temperature, which is a variable that decreases over time as more and more solutions are evaluated. Higher temperatures provide a greater chance for a solution to move uphill, and similarly lower temperatures provide a lower chance. The amount of time spent at a certain temperature and the speed with which it decreases are design variables, with longer times and slower speeds resulting in better solutions but taking longer to carry out. In this way it is similar to the cooling or annealing of a solid, hence its name [132]. One of the first major works using simulated annealing for assembly sequence planning was by Milner et al., who additionally tackled the line balancing problem at the same time by considering the assignment of tasks to workstations and selecting the appropriate, least-cost equipment [132]. The candidate solutions are chosen by using the graphical representation of liaison sequences to sample through different assembly orderings. The fitness function used is the total cost of the sequence while accounting for equipment and tool costs. Despite the usefulness of their implementation, detailed tool, equipment, and part data is needed to calculate the costs for the fitness function.

Hong and Cho, similar to Milner et al., also combined assembly sequence analysis with line balancing. Their justification was that the ideal sequence does not necessarily correlate with the lowest number of workstations, and similarly that if a specific workstation time is required then some otherwise ideal task orderings could actually require more workstations and cost more [133]. To address this, assembly task times are used to allocate the tasks among stations such that each station does not exceed a specific cycle time. An idle time criteria is used to represent this, which is joined by the assembly cost, precedence,

and connectivity criteria to create an energy function utilized as the fitness function. The assembly cost is not actually explicitly calculated and is instead a function of the assembly sequence's stability and number of direction changes. Precedence is addressed via the energy function objective. In the opposite direction of Milner et al.'s method, which requires excessively detailed cost tables, Hong and Cho's method does not actually calculate cost at all but only an analogue for it.

Motavalli and Islam noted that many methods only consider a single criteria during their optimization and so proposed a multi-criteria optimization routine using simulated annealing [134]. The criteria they used are assembly time and number of reorientations, combined into one objective via weightings on each. Though their method is one of only several to consider assembly time, the assembly time itself must be explicitly provided for each combination of operations, leading to the necessity of a large data table. Additionally, the candidate solutions are generated by randomly creating assembly sequences until one that meets the precedence constraints is generated, which is an inefficient process.

Despite the utility demonstrated by the methods utilizing simulated annealing algorithms, they ultimately proved to not be very popular for assembly sequencing due to limitations in generating feasible sequences and so few other works using it have been published.

3.1.7.3 Genetic Algorithm

A very popular soft computing method that is currently one of the most-used for assembly sequencing is the Genetic Algorithm (GA). It is based on the theory of Darwinian evolution wherein animals in a population compete with one another in order to mate and spread their genes. This is done by generally only allowing candidate solutions that yield a better fitness function value to mate. During mating, also called crossover, two candidate solutions, the "parents," exchange their design variables, or "genes," to create new "children" candidate solutions. This is done to pass on successful design variables to the next set of

candidate solutions while also ensuring that new, similar, and possibly better solutions are also explored. The genes involved in this each have a chance to mutate to a completely different value, which helps prevent solutions from being stuck in a local minimum. The fitness of all the parents and children are evaluated, the least fit individuals removed, and the process repeated in a new "generation" until an adequately fit function is determined from a candidate solution. It has proven to be so popular due to its simplicity of concept and implementation as well as its ability to yield close-to-optimal results.

Bonneville was perhaps the first to apply genetic algorithms to assembly sequencing [135]. The sequences are represented as an assembly tree composed of different subassemblies put together in different orders and are encoded into binary strings to act as the genes, or chromosomes. The initial population of sequences are selected by an expert. During crossover, different partitions of subassemblies are swapped between the parents to create children. If a mutation occurs, a subassembly in the chromosome is swapped with a neighboring subassembly. The fitness of all individuals in the population are evaluated by an expert, and if an individual violates geometric or precedence constraints it is removed from the population.

Hong and Cho encoded sequences as a series of part numbers which represent the ordering of assembly tasks [136]. The fitness function is dependent on the energy of the assembly, which itself relies on the sequence's cost and whether it obeys feasibility and connectivity rules for precedence. The cost is a function of the sequence's stability and number of direction changes. Infeasible sequences are not eliminated but rather penalized via the energy function. For crossover, the part numbers are exchanged between the parents to create the children, indicating the swapping of assembly operations. Similarly, mutation is carried out by swapping the ordering of different operations in the same individual.

De Lit et al. used an Ordering Genetic Algorithm where the order of the individual genes in a candidate solution's chromosomes are important [137]. Sequences are encoded as a series of liaison graph connections or links. A mapping function is used to ensure the

sequences are feasible and meet all assembly constraints. Mutations in the chromosomes do not just consist of swapping genes but can also involve taking a gene and inserting it somewhere else. An individual's fitness is calculated using an outside program instead of using an aggregate cost function. A candidate solution's fitness, however, is still ultimately dependent on its sequence's stability, number of reorientations, number of operations that can possibly be done in parallel, and its ability to assemble cheaper components first and more expensive ones later. In this way this method explicitly focuses on the sequence of an assembly sequence.

Chen and Liu pointed out that static probabilities for crossover and mutation are not necessarily ideal: high mutation rates provide better chances of global optimality at the cost of decreased convergence ability [138]. They used an Adaptive Genetic Algorithm to address this by dynamically varying the probability of carrying out the genetic operators as generations go on. Chen and Liu also added a third operator, cut-and-paste, wherein multiple portions of chromosomes within a candidate solution are extracted and inserted somewhere else. To aid with this, they encoded the sequences as assembly trees similar to Bonneville. Infeasible sequences are altered until they are feasible and the criteria for the fitness function is based on minimum number of reorientations.

Smith and Smith proposed an enhanced genetic algorithm that maximizes design space exploration by not using a fitness function to eliminate unfit individuals [139]. Instead, the fittest feasible individuals until that point are periodically injected into the population or when there are no feasible individuals left in the population. All feasible individuals are otherwise allowed to keep mating and reproducing. The fitness of the fittest individuals are determined according to feasibility and number of reorientations.

Tseng et al. [140] expanded on their connector-based method from 1999 [92] and incorporated a genetic algorithm into it. Each sequence is encoded as a series of connectors. Each connector has several properties associated with it, such as the type of connector it is, its assembly direction, the type of tool it requires, and its precedence. Tseng et al.

used an object-oriented language to keep track of these properties during the optimization. The fitness function revolves around having minimal connector type changes, minimum assembly direction and tool changes, and adhering to precedence constraints. It was noted that this method does not fare well with larger assemblies that use a very large number of connectors.

Marian et al. greatly expanded the type of assembly sequences analyzed beyond the most simple cases [141]. This includes: non-linear sequences, where more than one part has to be assembled at a time for a feasible sequence; non-sequential sequences, where parts have to be assembled simultaneously for feasibility; non-monotonous sequences, where intermediate operations such as rotating a component need to be done before the next operation can be done; and non-coherent sequences, where a part does not necessarily touch any other parts upon being inserted. Due to the more complex types of sequences being considered, insuring the sequences are feasible and follow other precedence constraints is of high priority. As a result, they used a guided search routine to generate the initial population and perform the crossover operation, which consists of building up the chromosomes by traversing through directly-connected liaison graph nodes and edges. Typical mutations are not performed because they are liable to produce infeasible sequences. What is done instead is a pseudo-mutation where a partition of chromosomes is selected and replaced with another partition of chromosomes that results in a feasible sequence, as determined by the guided search. The method is quite robust, being able to handle large 25-part assemblies as well as many different types of assemblies. However, due to the extreme focus on feasibility, the diversity of the solutions could be adversely affected without making the pseudo-mutation rate excessively high.

Tseng et al. considered an integrated ASP and ALB problem using a Hybrid Evolutionary Multi-Objective Algorithm due to the significant effects each problem has on the other [142]. Because the two are considered together, each candidate solution has two sets of chromosomes, one to indicate the priority of which liaison graph edges to carry out first,

representing sequencing, and one to assign those operations to different workstations, representing line balancing. The decision to only indicate the priority of different operations as opposed to forcing them to be carried out in the order they appear on the chromosome allows the generated sequences to always follow assembly constraints and be feasible. Similarly, there are a set number of workstations to simplify the encoding for line balancing. The fitness function is an aggregate objective function focused on minimizing tool changes, number of direction changes, and cycle time. The weights for these factors is randomized, allowing the best solutions for each set of weights to be used to create a Pareto frontier, hence being multi-objective. Though this method is able to combine both ASP and ALB, it makes significant simplifications with ALB such as keeping the number of work stations constant.

Choi et al. used a multi-criteria genetic algorithm specifically focusing on assembly time [143]. They assumed that the actual assembly operation time for each component all sum up to be the same for each sequence, but that the setup time is different for reach. The criteria they used for their fitness function is thus dependent on setup time and number of reorientations. The focus on setup time essentially provides a more concrete representation of the effects of tool changes between operations. This work stands out in that it addresses setup time, a result of tool changes, as opposed to tool changing itself, but due to this requires a detailed setup time table as input.

Tseng, Chen, and Huang looked at integrating both the areas of assembly sequence and manufacturing plant assignment with a genetic algorithm [77]. Different manufacturing plants are used to produce different components due to their ability to increase production capacity and capability and reduce costs. To do this, the manufacturing plant and assembly sequence information are both encoded into a single chromosome. The location of a manufacturing plant gene in the chromosome dictates that all the component genes that appeared before it belong to that plant. The initial population of sequences is set up to ensure the sequences are always feasible. If an invalid sequence/plant assignment is generated dur-

ing crossover or mutation, rules based on the assembly precedence information are used to repair it and make it feasible. The fitness function is based on the assembly operation cost, instability cost, accessibility cost, tool setup cost, weight cost of moving heavier components around, and general transportation cost of moving components from one plant to another. Tseng, Chen, and Huang noted that the complexity of the method could increase drastically if larger assemblies were analyzed.

As can be seen, due to the simplicity of implementation of genetic algorithms, numerous variants for it have been made, some of which have incorporated considerations for other areas of assembly on top of sequencing analysis. Despite this, it is also apparent that an issue that every genetic algorithm faces is how to deal with the highly order-centric geometric and feasibility constraints. Due to this reason, it is not the only highly popular soft computing method used for assembly sequencing. The next section will cover the other popular method, ant colony optimization.

3.1.7.4 Ant Colony Optimization

Ant Colony Optimization (ACO) is a soft computing method based on the ability of ants being able to self-organize and, as they forage for food away from their nest, find the optimal path to the food. This is due to the fact that, as the ants travel, they deposit slowly evaporating pheromones that other ants follow. This results in paths to the food closest to the nest being more well-traveled because it takes less time to get there, creating a positive feedback loop that ends in an optimized path to the food being found. In a more sequencing-related context the ants probabilistically travel from node to node in a graph. Their chances of moving to any one node are dependent on how much pheromone is present, whether a feasible sequence would result if the node is traveled to, and on other heuristic rules implemented by the method designer. Any pheromone currently present is evaporated by a small amount before new pheromone is added, a process called local updating. An ant keeps traveling from node to node until there are no feasible nodes left, at which point it has

completed its tour. Once all ants have completed their tours all the generated sequences, which are represented by the tours, are evaluated and the best ones further reinforced with pheromones, a process called global updating. The next iteration then starts. The local updating aids design exploration and helps prevent the ants from being stuck at a local minimum, while the global updating encourages ants to travel to better solutions. Although much more complex than genetic algorithms in concept, ACOs are more easily able to handle assembly constraints and can sometimes have better convergence as well.

Wang et al. were one of the first to use ACOs for assembly sequencing [144]. The graph structure they created for the ants to traverse uses an assembly-by-disassembly approach. In the structure, each component consists of six nodes representing its disassembly in the six principal Cartesian coordinates. All the nodes are connected to each other and the ants can only travel to each part once. A disassembly matrix, which is a pseudo-interference matrix that also encodes precedence constraints, is used to dictate nodes the ants are allowed to travel to based on what nodes they have already visited. A heuristic is used to make the ants less likely to travel to nodes that cause a disassembly operation direction change. The global updating is based on the number of reorientations of the best sequence in that iteration, with fewer reorientations eliciting more pheromone deposition to encourage higher quality sequences. In their use case of a transmission assembly, Wang et al. showed that their ACO performed better than a genetic algorithm due to not having to worry about unfeasible sequences, making the case for ACOs in general, though they noted that their implementation did not address tool changes or stability.

Lu et al. looked at the disassembly problem instead of the assembly-by-disassembly problem and used an ACO to find multi-objective solutions [145]. The graph the ants traverse through simply consists of nodes of all the assembly's parts. The nodes the ants are able to travel to during the optimization process is dictated by three interference matrices, one for each principal direction. To handle the multi-objective aspect Lu et al. assigned weights to the three objectives: number of reorientations, tool changes, and different oper-

ation types. They then integrated them as part of the heuristic that influences which nodes the ants move to. Additionally, the best non-dominated sequences in each iteration receive additional pheromones during the global updating process. Lu et al. were thus able to address the aspects missing in Wang et al.'s [144] work.

Zhang, Sun, and He created a technique to try to speed up convergence [146]. They altered the local updating procedure so that the ants have a chance of simply traveling to the next best node in terms of pheromones and heuristics on top of moving stochastically to the next node like in regular ACOs. Additionally, during the global updating process all sequences receive additional pheromone corresponding to their quality. This quality and their heuristics are both dependent on the assembly time and number of reorientations.

Yu and Wang created a Max-Min Ant System [147]. Their method utilizes their developed turning and extended interference matrices instead of regular interference matrices. Their ants also have a chance to directly travel to the next best node like in Zhang, Sun, and He's work [146]. The amount of pheromone deposited during the local updating process is dependent on the length of the liaison graph edge the ant is on. The global updating process alternates between updating the iteration best sequence and the global best sequence so that adequate diversity is maintained for better convergence. Maximum and minimum limits on how much pheromone there is on the edges are set so that the ants are not too biased and converge too early. If a starting node position consistently yields unfavorable solutions over many generations, it is subsequently no longer considered when choosing initial node positions. Yu and Wang used stability, continuity, which is whether two parts are contacting when assembled, auxiliary stroke, which is how close two nodes are to each other on a liaison graph, number of reorientations, and assembly operation types metrics for heuristic information.

Wang, Rong, and Xiang wanted to map out the solution space and so used an assembly-by-disassembly approach to create a Disassembly Feasibility Information Graph (DFIG) to store information about all feasible disassembly sequences [148]. Their intent was to

have a two-stage approach where the solution space is first mapped out before an ACO finds the optimal sequence in that space. They acknowledged that trying to generate all feasible sequences is itself infeasible for a large assembly and so devised a method where the creation of the DFIG is guided by the ACO algorithm. Essentially, as the ants travel, nearby nodes are also checked for feasibility and the information added to the DFIG. If an ant runs out of possible nodes to travel to before it finishes its tour, the ant "dies" and the sequence of nodes it traveled through is marked as infeasible. How the ACO algorithm itself works is that, after an initial DFIG is created, the ants use the DFIG to query nodes they are allowed to travel to as well as heuristic information to influence which node they will select. The heuristic is based on assembly cost and composed of assembly time, number of reorientations, and number of direction changes. They summarized by saying their method scales more linearly than exponentially as compared to other methods when assembly size increases.

Yang and Lu used an ACO algorithm to integrate ASP and ALB because the problems mutually affect each other: an assembly operation assigned as the last step in a workstation may not have to deal with issues like reorientations, tool changes, or assembly direction changes. Similarly, consideration of those metrics changes the assembly time of each task, affecting how tasks are distributed among workstations [149]. Yang and Lu specifically wanted to minimize the number of workstations given a constant cycle time but, due to sequencing considerations causing total assembly time to vary, effectively had to deal with minimizing line efficiency. The fitness function is a function of the line efficiency and smoothness index, which are dependent on the total assembly time, cycle time, and number of stations. Because actual time must be used instead of an analogue, Yang and Lu converted direction changes and tool changes into a metric with time units, which is not normally done. The nodes the ants can move to are dictated by disassembly precedence graphs. Augmented liaison graphs that indicate the tool required for each edge are used to define tool changes. The heuristic information leveraged is based on the number of tool

and direction changes. Assembly sequences are built up during the ant's tour like in regular ACO methods, while workstations are assigned by creating a station every time the assembly time of a series of operations exceeds the cycle time. Yang and Lu used the same assembly as Tseng et al. [142] to compare their integrated methods and demonstrated that their method had slighter worse line efficiency and slightly better direction and tool changes, meaning the ACO implementation is at least equivalent to the GA implementation.

ACO algorithms, while complex in concept, overall fit the requirements of assembly sequencing analysis very well due to being able to accommodate for the different constraints automatically as opposed to having to repair sequences like with many GAs. Similarly, their performance has been shown to be roughly equivalent and sometimes better than their GA counterparts, which explains why ACO algorithms have grown to be so popular.

3.1.7.5 Particle Swarm Optimization

The last class of soft computing methods to be covered are Particle Swarm Optimization (PSO) algorithms, which have recently started seeing more usage. PSO algorithms are a population-based evolutionary technique centered around the concept of swarms of fish and flocks of birds being able to self organize and move together in formation towards food or away from danger. In a more mathematical sense, each candidate solution, or particle, moves around the design space with a certain velocity searching for a food source, the optimal solution. As the particle does so, it stores in memory the location of the best solution it has come across, called the personal best. It is also aware of the position of the particles around it and stores in memory the location of the best solution of all particles up until that point, called the global best. Every iteration, the velocity of each particle is adjusted based on the local and global best solutions so that it can get closer to them. The relative influence of the local and global bests are determined via weightings. PSO algorithms are steadily growing more popular due to being simpler to implement than SA

and ACO algorithms in terms of formula complexity [150]. Additionally, their population does not change over the course of the algorithm and is smaller than the population of SA and GA algorithms, making them easier to initialize [150]. However, they are still not as widespread as ACOs and GAs because the position and velocity concepts inherent to PSO are continuous in nature whereas ASP revolves around discrete variables, meaning a workaround must always be made [97].

Liu and Wang authored one of the first works using PSO with assembly sequences [150]. In their work, each solution is a vector of numbers indicating the sequence of components to be assembled. To accommodate for the discrete nature of sequences, the velocities are sets of vectors that indicate which operations in the solution are to be swapped. The fitness function is dependent on the assembly's constraints, which include precedence, geometric feasibility, number of direction changes, number of tool changes, stability, and number of connection changes. Connection changes here refer to changes in connector types, such as rivets, screws, and bolts. Liu and Wang found that the quality of their PSO results was highly dependent on the initial population quality and the number of particles used and required more computation time than the SA algorithm they were comparing against.

Tseng, Chen, and Huang [78] looked at the integrated ASP and multi-plant assignment problem using a PSO algorithm, similar to how they implemented it with a GA [77]. An assembly precedence graph, precedence matrix, and interference matrix are used to determine sequence constraints, a plant capability table is used to determine which operations can be performed at which plant, and an assembly capability table is used to determine the operations' properties, such as cost and tool usage, in that plant [78]. The fitness function consists of assembly operation cost, tool change cost, setup change cost, and general transportation cost. To encode the sequence of assembly operations and plant assignments while also making PSO's continuous nature more discrete, Tseng, Chen, and Huang expressed their solutions/particles as matrices. Due to this representation, the velocity and its

calculation and operations are allowed to stay continuous instead of being used to perform swapping operations like in previous works. Tseng, Chen, and Huang noted that, similar to their GA implementation, larger assemblies could render their method infeasible.

Lv and Lu implemented a discrete PSO algorithm where solutions/particles are composed of three vectors, one for their location/assembly sequence, one for their velocity, and one for the personal best solution they have stored in memory [151]. The location is a list of discrete numbers denoting the order of the components to be assembled, for example, if the number 4 is the third component, it indicates that component 4 is the third component to be assembled. Velocity and particle location operations are redefined such that their addition, subtraction, and multiplication yields a result in the discrete integer space as opposed to the usual real continuous space. The fitness function is based on number of parts, assembly orientation changes, tool changes, and operation type changes. If the sequence is infeasible, it suffers a large penalty to the fitness function. Lv and Lu tested this on a 22-part assembly use case and compared the results with a multi-objective GA, showing the results are comparable and their method viable for ASP.

Wang and Liu expanded on their previous work [150] and added a chaotic operator to their original PSO algorithm because typical PSO algorithms can become trapped in a local minimum once the particles have traveled to a stored global best solution [87]. The chaotic nature of their new algorithm helps alleviate this by scrambling the personal best solution of every particle on every iteration. Wang and Liu used the same fitness function, solution representation, and style of velocity operations as in their previous work [150]. Due to their improvements, during their use case on the same 15-part generator they found that the chaotic PSO algorithm converged significantly faster than their previous PSO algorithm as well as the SA algorithm used as the baseline [87]. Additionally, the quality of the initial population did not have as much impact on the chaotic algorithm as the regular one and did not need as large an initial population either.

Mukred et al. made a binary PSO where the particles and velocities are stored in binary

instead of traditional numbers to have a more discrete ASP PSO implementation [152]. The binary values in the velocities are used in a sigmoid function to determine the probability of flipping the binary values in the particles from 0 to 1 or 1 to 0. They used a GA-influenced method to ensure feasibility and diversity of solutions, wherein infeasible sequences have the order of random operations swapped as part of a "mutation" until they are feasible. Similarly, these mutations occur if the particles are converging on a solution to make sure they are not stuck in a local minimum. Assembly constraints are stored in a precedence matrix and the fitness function is dependent on assembly operation and setup time. Mukred et al. compared their method with Choi et al. [143] and demonstrated that their algorithm is slightly computationally faster than the referenced SA and GA techniques [152].

Wang, Kang, et al. used a modified PSO algorithm to determine both the assembly operation types to create an assembly's features as well as their assembly sequence [153]. They encoded this information in a two-row matrix for each particle. The values in each column represent a specific assembly feature. The numbers in the first row are used to encode the operation information for the features in their respective columns. The information consists of machine type, tool used, and tool approach directions. The numbers in the second row indicate the priority of the features to be operated on. Infeasible sequences are rearranged until they are feasible using a precedence matrix built using operation type and geometric interference information. The velocities are matrices of the same size as the particles' solutions. Due to all the numbers in the particles' matrices being used to represent priority or encode other information, the issue of position-velocity interactions needing to be discrete is avoided. To prevent early convergence, four particles are changed every iteration in four separate ways, with two ways altering the operation type and the other two altering the sequence. The fitness function is dependent on the machine cost, cutting tool cost, machine changing cost, cutting tool changing cost, and setup cost. Their work is relatively unique in focusing on both fabrication and assembly for sequencing.

Rashid et al. combined many past PSO works to make a multi-objective discrete PSO

to tackle the integrated ASP and ALB problem [154]. A matrix implementation of a precedence graph is used to represent both the ASP and ALB problems and a sorting procedure leveraged to assign tasks into a sequence one by one such that the sequences are always feasible. Tasks in a sequence belong to the same workstation until their total times exceed the cycle time, at which point a new station is opened to assign further tasks to. The objective functions are based on the number of direction and tool changes on the ASP side and cycle time, number of work stations, and work variation on the ALB side. The addition, subtraction, and multiplication of particle velocities and locations is done in a discrete way similar to Lv and Lu's work [151]. Rashid et al. tested their algorithm with numerous problem sets and compared their results with many other multi-objective algorithms, demonstrating that theirs generally performed better but had one of the highest computational times.

PSO algorithms have come a long way ever since they were first used for assembly sequencing and have been shown to sometimes be better than GA and ACO techniques. However, an issue that still plagues them is the fact that they are continuous in nature whereas the sequencing problem is discrete. This means every technique must have a workaround for that problem. Additionally, perhaps even more-so than with GAs, they are prone to early convergence unless they adapt some form of genetic operators taken from GAs or use some other method to add diversity to their solutions.

As PSO algorithms are the last of the soft computing methods to be examined, and with soft computing methods being the current state of the art of tackling ASP problems, this concludes the overview of ASP.

3.2 Assembly Line Balancing

3.2.1 Line Balancing Overview

A broad review of ASP was carried out due to the desire to add ASP as a whole into aircraft design systems-thinking methods, which currently lack that analysis capability. The review of ALB is guided by asking the following question:

Guiding Question 2: How is assembly line balancing carried out and what is the current state-of-the-art on problem types related to the material factors that can be used to better integrate aircraft design and assembly?

This makes the ALB review more targeted on the identified material factors from section 2.3 because baseline ALB analyses have already been incorporated into aircraft design systems-thinking methods. Those baseline capabilities just need to be further enhanced with the identified factors. These material factors are integral fabrication, variable manufacturing processes, spatial constraints, and variable throughput and cost. An overview of what ALB entails is first provided before delving into the problem variants centered on those factors.

Line balancing consists of assigning assembly work operations, also called tasks, to different workstations according to their precedence relations [155]. The precedence relations typically combine both manufacturing precedence constraints and any geometric or mechanical feasibility constraints since assembly sequence planning is not always present to do the latter separately. The representation of the precedence relations are considerably simpler than with ASP and are almost always represented as just precedence graphs of tasks and their accompanying task times. The workstations are somewhat flexible in their definition and can mean an arbitrary collection of ordered tasks that follow assembly precedence constraints or a group of related tasks that, while still following precedence constraints in their ordering, are operated on by a single, particular set of tools and equipment. The assignment of tasks to work stations is done with the major constraint being that the sum of the individual task times in a station, called the station time, cannot exceed the cycle time. Cycle time, according to Becker and Scholl, is the "maximum or average time available for each workcycle" [155]. Taking the inverse of the cycle time yields the production rate. This assignment of tasks is done with respect to various objectives. The most common is to either minimize the cycle time, effectively maximizing production rate, or minimize the number of workstations, an analogue to minimizing cost by having as little equipment and

therefore as little equipment cost as possible. Line balancing is called what it is because it involves optimally balancing the tasks in the assembly line with respect to the objectives while ordering them according to the precedence constraints. The problem of making these balancing decisions is therefore known as the Assembly Line Balancing Problem (ALBP) [155].

There are a fair few solution methods for ALBPs, but they are not as varied as in the ASP literature. The solution methods are split into three primary groups: exact methods, heuristic methods, and meta-heuristic methods. The most popular exact methods are dynamic programming and branch and bound methods. Dynamic programming is focused on breaking up the problem into a series of sub-problems and then optimizing these sub-problems. The solutions to these sub-problems are saved and used to recursively solve other sub-problems in either a bottom-up or top-down approach until an optimized solution to the original problem is found. It is faster than trying to solve the original problem in its totality but requires a large amount of memory to store the sub-problem solutions. Branch and bound methods are somewhat similar to dynamic programming in that they first also break down the original problem into smaller sub-problems. From here, the upper and lower bounds of where the optimal solution must be is determined before a "branch" in between these bounds is explored. These branches are recursively solved to try to establish new upper and lower bounds until the bounds are of a sufficiently small size that the optimal solution is found. Branch and bound methods can be efficient in that they do not need to explore all possible solutions, but can still take a considerable amount of time for large and complex problems.

Heuristic and meta-heuristic methods for ALB have the same definition as described above for ASP, with heuristics being a rule-of-thumb used to solve a problem and meta-heuristics being a guideline that directs how heuristics are used to solve a problem. As described by Capacho et al., heuristic methods can include ones that use greedy approaches, where the best solution is picked from a sampling of solutions, and ones that use con-

structive approaches, where priority rules are used to give ranks to tasks. The tasks are subsequently assigned to workstations based on their ranks. The priority rules can be based on a task's task time, its number of successors or predecessors, as well as the earliest and latest stations it can be placed in [73]. Meta-heuristic techniques are basically the same for ALB as for ASP. The most common meta-heuristic techniques used for ALB are GAs followed by ACO algorithms and PSO algorithms, based on Oesterle et al.'s sampling [76]. Meta-heuristic methods are nowadays the most used solutions methods due to the complexity and size of the ALBPs encountered and formulated. The next few sections go over the most pertinent ones to this work.

3.2.2 Simple Assembly Line Balancing Problems

The simplest, most well-studied, and most common ALBP is the suitably named Simple Assembly Line Balancing Problem (SALBP). According to Boysen et al. it is the quintessential line balancing problem and, though perhaps too simplistic for most real problems, is the basis of almost all the other, more general forms of the ALBP [156]. In the SALBP, an assembly line consists of $k = 1, ...m$ workstations. Fabrication and assembly operations are consecutively performed on the product in the workstations as it moves its way down the assembly line. The operations are composed of "indivisible units of work" called tasks $V = 1, ...n$, and the time allotted for the product entering a workstation, having all the tasks at that station carried out, and then leaving that workstation is the cycle time c [156]. Each task has a task time t_j and is assigned to a work station in the order dictated by its precedence relations, with the set of all tasks at a workstation being its station load S_k. The sum of all the task times at a workstation, called the station time $t(S_k) = \sum_{j \in S_k} t_j$, is not allowed to exceed the cycle time c in the SALBP. Assembly lines exhibiting such a property are also called paced lines because every workstation can start its tasks simultaneously and hand them off to the next station at the same paced rate. Although the station time can never exceed the cycle time in a paced line, it does not necessarily have to equal it, with

the unproductive time being called idle time $c - t(S_k)$. Because tasks are considered indivisible, the cycle time c is normally limited to being no smaller than the largest task time t_j, effectively capping the production rate to being the inverse of the cycle time/longest task time [155, 156].

The SALBP is considered simple due to its line balancing using the following assumptions, which are adapted from Boysen et al.[156]:

1. Only one homogeneous product is being produced.

2. Only one production process is being used.

3. The assembly line is a paced line.

4. The assembly line has no feeder or parallel lines.

5. The ordering of the tasks must follow precedence constraints.

6. The task times are static and deterministic.

7. The only task assignment restrictions are those regarding the precedence constraints.

8. Tasks cannot be divided among more than one station.

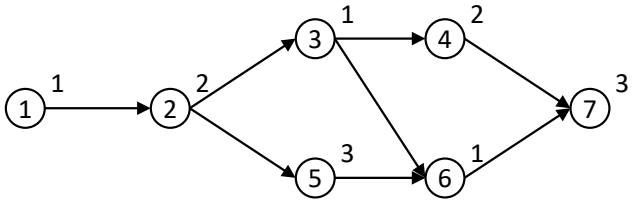
9. All stations have the same resources such as workers and equipment and tools.

There are several types of SALBP. If the objective is to minimize the number of stations, m, while keeping the cycle time c constant, then it is a Type 1 SALBP, or SALBP-1. SALBP-1 is used primarily to reduce the number of stations and therefore the costs of opening up a new station. If the objective is to minimize the cycle time c while keeping the number of stations m constant, it is a Type 2 SALBP, or SALBP-2. This type of problem is used to optimize cycle time and production rate when the number of stations is known. If both the cycle time and number of stations can be varied, then the objective is to maximize the line efficiency E, defined as $E = t_{sum}/(m * c)$ where t_{sum} is the sum of all the task

times. Because $t_{sum} - (m * c)$ is the total idle time among all the stations, maximizing E amounts to minimizing the total idle time. This is the Type E SALBP, or SALBP-E. If both the cycle time and number of stations are constant, then the objective is to simply verify that the resulting line balances are feasible, which is a Type F SALBP, or SALBP-F. A summary of this, adapted from Becker and Scholl, is shown in Table 3.1 [155]. An example precedence diagram for line balancing and the resulting SALBP-1 line balance is shown in Figure 3.18.

Table 3.1: Types of SALBP, Adapted from Becker and Scholl [155]

	Cycle Time	
	Constant	Optimize
Number of Stations		
Constant	SALBP-F	SALBP-2
Optimize	SALBP-1	SALBP-E



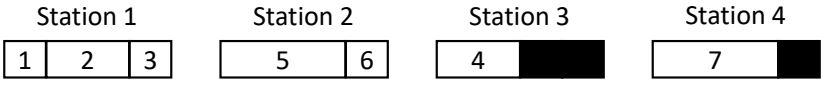


Figure 3.18: Precedence Graph Showing Task Ordering and Task Times on Top, SALBP-1 Example with Cycle Time of 4 on Bottom

Due to all of their assumptions, SALBPs are sometimes too restrictive to accurately capture real life interactions when balancing an assembly line. The loosening of such

restrictions to allow for more general problems to be addressed is aptly called the General Assembly Line Balancing Problem (GALBP). In GALBPs various assumptions from the SALBP are removed to model real-world characteristics of interest. This increased realism carries with it the tradeoff of the problem also being more difficult to solve. The GALBPs most pertinent to the material factors that need to be addressed and how they are solved are covered in the next sections.

3.2.3 Assembly Subgraphs and Equipment Selection Problems

In the context of ALB, the integral fabrication and manufacturing process selection material factors have some similarity. Integral fabrication is predicated on there being an alternative for a part to be assembled one way via fabrication and another way via more traditional assembling after it has been fabricated. In the assembly context, manufacturing processes differ from one another in that they use different sets of equipment, have alternative sets of tasks, and can have different task times if they share the same sets of tasks. The latter two aspects of manufacturing process selection and all of integral fabrication thus have the concept of "processing alternatives" in common.

This concept in the ALBP context was first introduced by Pinto et al. [157] and breaks the SALBP's assumptions 2 and 9. It is represented in their paper as alternative tasks being present with their corresponding alternative task times in a precedence graph. When assigning tasks to stations, either the original task or the alternative tasks can be used, but not both. Both sets of tasks must abide by zoning constraints, which are discussed later. Alternative tasks and processes have higher facility costs with the tradeoff of potentially lower labor costs. By accounting for higher costs if alternative processes are used, they essentially implemented a preliminary version of the equipment selection problem, which will also be discussed later. Pinto et al. focused on minimizing the sum of the labor and facility costs given a cycle time. They formulated their manufacturing alternatives problem as an integer programming problem, and solved it with the branch and bound method. Pinto

et al. additionally noted two very important details. First, alternative processes may have total labor time and cost savings, but those are only realized if they can reduce cycle time or the number of stations. And second, alternative processes with seemingly small total labor time differences can have large labor cost savings by reducing the cycle time enough to reduce the number of stations, increasing efficiency.

Capacho et al. made this more formal with what they called the assembly subgraphs ALBP [73]. It was developed to address the fact that alternative subassemblies can be made during a product's assembly and that, for the two important reasons pointed out by Pinto et al., must be actively considered while line balancing: pre-selecting an alternative that initially appears inferior could actually lead to a worse solution. These assembly alternatives appear as separate tasks with their own task times in the precedence graphs, similar to Pinto et al.'s work, and can be mutually exclusive. This causes the precedence relations and task times to be dependent on the alternatives being evaluated and so are not fixed. These alternative subassemblies in the precedence graph are called subgraphs, hence the name of the problem type. Capacho et al. solved this using their own customized heuristic methods. Though their work focused on subassemblies, it tackles the same fundamental issues as Pinto et al.'s processing alternatives and so can be applied to alternatives manufacturing processes.

A special implementation of the assembly subgraphs ALBP was made by Scholl et al. called the sequence-dependent ALBP [158]. In the sequence-dependent ALBP, the task times are dependent on the order or sequence that the tasks are traversed through in the precedence graph. This is due to some orderings of tasks reducing accessibility for future tasks, essentially changing their setup times. The formulation is akin to the assembly subgraphs ALBP because the different orderings through the precedence graph are akin to choosing different subgraphs, though none of the tasks in the sequence-dependent ALBP can be mutually exclusive. Scholl et al. created many types of sequence-dependent ALBP in a similar way to there being many types of SALBP and found they were most efficiently

solved with branch and bound methods. From all this it can be seen that the sequence-dependent ALBP is therefore not just a special version of the assembly subgraphs problems but also a basic implementation of integrating the ASP and ALB problems with the addition of dynamic task times and precedence graphs.

Scholl et al. continued their work on less specialized assembly subgraphs and further formalized the concept by introducing the concept of OR-nodes into the precedence graphs [159]. OR-nodes act as start and end nodes that enclose the subgraphs and make it easier to understand what alternatives are possible in the precedence graphs. To make it easier to connect from the main precedence graph to the subgraphs, unique start and end OR-nodes are present for every set of alternatives and essentially act as dummy nodes. Scholl et al. then improved on the binary problem formulation of the assembly subgraphs problem and used a branch and bound-based method to solve it. They did note, however, that the original subgraphs problem formulation was solvable only up to 50 tasks and that their own implementation went up to about 297 nodes.

These assembly subgraph problems are distinct from the integrated ASP and ALB problems reviewed in section 3.1 because the former can have mutually exclusive process alternatives while the latter do not. This difference is likely why the former is much more rarely addressed than the latter and why relatively few works on it exist.

The assembly subgraphs ALBP was covered as a way to implement the integral fabrication and part of the manufacturing process selection material factors. The other part of the manufacturing process selection material factor is in regard to the equipment selection, which is covered next.

Choosing the correct set of equipment is important since equipment have a large impact on multiple factors: they determine the processes that can be used, which in turn decides what materials the assembly's components can be made out of, and also greatly affect the cost and throughput of the entire assembly line. Since assumption 9 in the SALBP prevents equipment and tool information from being obtained from a workstation and therefore pre-

vents estimation of equipment costs, it can be changed such that each station has its own set of equipment and tools. This results in the equipment selection problem which, according to Bukchin and Tzur, is characterized by different sets of equipment that are all capable of carrying out the tasks to produce the product but vary in their flexibility, production speed, and cost [74]. Flexibility in this context refers to the ability of the equipment to perform a large number of tasks, as compared to their definition of an inflexible set of equipment that can only accommodate one task. As stated by Bukchin and Tzur, if one set of equipment does not completely dominate another in all three categories, than either can conceivably be part of the optimal line balance. Their work focused on minimizing cost given a set cycle time. The task to equipment assignation is carried out by each set of equipment having a bank of tasks that it can operate on. They implemented a branch and bound algorithm to solve their equipment selection problem.

Bukchin and Rubinovitz expanded on that work by combining the concept of station paralleling with equipment selection [160]. Station paralleling involves the duplication of a workstation and everything in it, thereby doubling the duplicated stations' cycle times. This has three main benefits: the cycle time change allows more tasks to be placed in a workstation, enabling greater line efficiency with station times being closer to cycle times; it makes it possible for high production rates and low cycle times should a single task have a task time higher than the cycle time; and it provides redundancy and, therefore, reliability should a workstation fail. Station paralleling thus allows for tradeoffs between the increased equipment and labor costs of the duplicated station versus the increased throughput, efficiency, and reliability that it brings. The concept will be further examined in subsection 3.2.5. Bukchin and Rubinovitz used station paralleling to further augment the trades that could be made with equipment selection. In their formulation their objective was to minimize line balancing cost. Stations are duplicated based on whether the financial benefit of improved efficiency outweighs the cost of the duplication. The financial benefit is significantly increased if the cost of not duplicating is violating cycle time constraints. As

each task can have its own equipment set, the question of whether or not to duplicate a station can potentially be applied for each individual task. Bukchin and Rubinovitz solved their formulation with a branch and bound procedure.

Milner et al.'s work was already mentioned earlier for ASP but is one of the works that also incorporates line balancing and, specifically, looks at the equipment selection problem and solves it via simulated annealing [132]. Tasks have three equipment types to choose from: manual labor, fixed equipment, and flexible equipment. Flexible equipment can handle many tasks, fixed equipment and manual labor can handle only one task, and equipment types other than manual labor have lower labor costs but higher equipment costs. The matching of tasks to their allowed equipment types, as well as the determination of each equipment type's cost, is done via one of their software's task-to-resource allocation matrices. One of their more remarkable findings was that, for the product they were analyzing, the assembly sequences did not vary or matter as much at very high production rates due to the high rate necessitating every task having its own workstation. More variability was noted at lower production rates.

A commonality with all these works is that their formulations are roughly the same, where different equipment sets change the task times and cost different amounts. They are particularly common as an add-on aspect where they are not the main sole focus, as could be seen already with both Bukchin and Rubinotz's and Milner et al.'s works. Due to their abundance no further ALB focused solely on equipment selection will be looked at in detail.

A closely related concept to the equipment selection problem's flexibility aspect is the zoning constraint. Zoning constraints, according to Vilarinho and Simaria, "force the assignment of certain tasks to a specific workstation" [161]. They are typically used to make sure that tasks requiring a particular equipment type are only assigned to a station equipped with that equipment, allowing the task to be performed [162]. There are two types of zoning constraints. Positive zoning constraints force tasks to belong to the same workstation,

generally if they use the same equipment for more efficient equipment utilization, while negative zoning constraints prevent tasks from being assigned together in specific workstations, whether due to equipment incompatibility, safety reasons, or a specific task needing some buffer time [161, 162].

Vilarinho and Simaria tackled a mixed-model problem where the assembly line can assemble slightly different models of the same product [161]. They minimized the number of stations while keeping cycle time constant in their work. To aid with that, the parallel stations concept is also used if a task's task time is higher than the cycle time. Both negative and positive zoning constraints are used to ensure tasks are assigned to the appropriate stations. This is because while similar product models share many tasks, some are mutually exclusive and cannot be paired with one another, hence the zoning constraints. The max number of station duplications is the max task time in a station divided by the cycle time, rounded up. Tasks that must be together under positive zoning constraints are combined to form one task in their stations while tasks under negative constraints cannot be assigned to a station if another negative constraint task is present. Vilarinho and Simaria solved their problem formulation with an ACO algorithm.

Akpinar and Bayhan also looked at a mixed-model problem and, similar to Vilarinho and Simaria, used parallel stations and zoning constraints [162]. However, Akpinar and Bayhan used a hybrid GA and so had to make sure that, when decoding chromosomes to assign tasks to workstations, the zoning constraints were never violated. They did this by opening up a new workstation whenever a violation was liable to happen. The zoning constraints themselves are implemented as a bank of tasks that have to be assigned to the same stations for positive constraints and as a bank of tasks that can never be in the same station for negative constraints. Stations are only duplicated if one of their constituent tasks' task times sufficiently exceeds the cycle time. Akpinar and Bayhan went on to further explore the capabilities of their hybrid GA.

Li et al. combined many different concepts together in their work including parallel

stations, equipment selection, zoning constraints, and even ASP [163]. Their goal was to minimize cost while keeping to a certain throughput. ASP is used due to the large variety of possible sequences, configurations, and subassemblies without the foreknowledge of which one would be the best one. The work consists of an outer loop and an inner loop, with the assembly sequence determined in the outer loop and the line balancing solved with a GA in the inner loop. Equipment selection is important in their problem because they determine whether tasks can be performed in series or in parallel. Tasks can be relatively freely assigned to almost any equipment sets desired, but they must keep positive and zoning constraints in mind during said assignation. In their example they demonstrated that, for their product, assembly lines with mixed equipment types performed better than lines with only one type of machine.

The assembly subgraphs and equipment selection problems, combined with the concept of zoning constraints, are used throughout the ALBP literature to model the tradeoffs between different processes. The assembly subgraphs especially allows for different subassemblies to be analyzed with variable task times and so could be useful for modeling of integral fabrication. There thus exist points of reference within ALB analysis that can be used as stepping stones to integrate the integral fabrication and manufacturing process selection material factors into aircraft design systems-thinking methods.

3.2.4 Space-Related Problems

The next material factor to be examined concerns the spatial constraints. Most line balancing problems do not consider spatial constraints due to them typically being handled later on in the design process with analysis like facility layout design. However, several works by Bautista, Chica, et al., incorporate spatial constraints in what they define as the Time and Space Assembly Line Balancing Problem (TSALBP) [164, 165, 75].

Bautista and Pereira created this new type of line balancing problem because accounting for space can reduce the physical dimensions of a workstation in a manufacturing plant and

enable it to fit better, allow workers to travel less, and make grabbing tools within the workstation easier [164]. It is more realistic and so can indirectly reduce cost and increase throughput. In the TSALBP, the amount of space required for each task is accounted for in the same way each task's cycle time and precedence relations are tracked. Specifically, the TSALBP adds a space constraint to each workstation such that the area taken up by the individual tasks in that station cannot exceed the station's allotted area, A, and essentially acts like the cycle time constraint. The list of variables that can be optimized for or kept constant thus includes the amount of space occupied by each workstation on top of the more traditional number of stations and cycle time. The amount of space needed by each task is represented by a value next to each task's task time in the precedence graph. Table 3.2, adapted from Bautista and Pereira's work, lists out all the different types of TSALBPs.

Table 3.2: Types of TSALBP, Adapted from Bautista and Pereira [164]. The "1," "2," and "3" Refer to the Optimization of Number of Stations, Cycle Time, and Station Area, Respectively, "F" Stands for Feasibility

Type	Number of Stations	Cycle Time	Station Area
TSALBP-F	Constant	Constant	Constant
TSALBP-1	Optimize	Constant	Constant
TSALBP-2	Constant	Optimize	Constant
TSALBP-3	Constant	Constant	Optimize
TSALBP-1/2	Optimize	Optimize	Constant
TSALBP-1/3	Optimize	Constant	Optimize
TSALBP-2/3	Constant	Optimize	Optimize
TSALBP-1/2/3	Optimize	Optimize	Optimize

Bautista and Pereira, and the majority of the TSALBP literature, focused on the automotive industry in particular. In their initial work they primarily looked at solving the TSALBP-1 with an ACO algorithm after defining the TSALBPs [164]. Of note is that they showed that their ACO algorithm yields solutions of comparable quality to branch and bound methods. Bautista and Pereira later expanded on their initial work and implemented a bounded dynamic programming method to solve the problem instead, which performs

better than their original ACO algorithm [165].

Chica et al. followed up on this by tackling the TSALBP-1/3 variant [166]. That variant of the TSALBP keeps the cycle time constant while minimizing the station area and number of stations as the objective. This particular variant was chosen for several reasons. Production rates are often fixed by the company's market objectives, which locks in the cycle time. Given a locked-in cycle time, the minimum number of work stations yields the lowest number of workers and therefore the lowest cost. And minimizing the area allows workers to more easily move around and access their tools, increasing labor and line efficiency. Chica et al. worked on developing a new multi-objective ACO algorithm to solve this problem, noting that constructive methods like ACO are more appropriate for the TSALBP. They utilized highly specialized and complicated heuristic information to guide the ants in their ACO algorithm but found that they obtained better information without it. They also implemented a GA implementation of the problem, the Non-dominated Sorting Genetic Algorithm 2 (NSGA-II) to be exact, to serve as a point of comparison and found that the solutions there were of minimal diversity compared to their multi-objective ACO algorithm. They concluded by testing their formulation on a real world Nissan Pathfinder engine use case.

Chica et al. afterwards explored this abnormality with their GA implementation performing so poorly as compared to the ACO one [75]. Their previous GA implementation was typical for line balancing, where the chromosome length is equal to the number of tasks and stations filled up by reading tasks in the chromosome from left to right until the cycle time is about to be exceeded. At that point a new station is opened. The crossover operation involves two cut points in each parent chromosome, with one offspring copying the start of one parent's chromosome, stopping at the first cut point, and starting again at the second cut point and going to the end. The portion in between the cut points uses the relative order of tasks from the other parent's chromosome, with this relativity ensuring feasibility. The second offspring carries out the same operation for crossover but with the

first and second parent's roles reversed. This was found to be far too limiting. Instead, Chica et al. made a new GA formulation where the chromosomes have dummy genes inserted to indicate when to start and end a new station. The number of such dummy genes is inherited from the parent chromosome or randomly generated. And for the crossover operation, roughly the same procedure happens except that the genes between the two cut points appear in the same order as in the second parent. Infeasibility in the form of exceeded cycle times or empty stations are fixed with a repair operator. Mutation schemes changes to induce further diversity are also implemented. Chica et al. demonstrated that their new GA implementation performs better than both their previous GA formulation as well as their previously developed ACO algorithm.

Zhou and Wu tackled the TSALBP-1/3 variant but looked at industrial robots specifically instead of the automotive industry [167]. This change in focus results in several major differences. The first is that each station has only one industrial robot that performs all the tasks in that station, which entails there being time allocated for the robot to change tools between tasks if the tasks require different tools. The more tasks requiring different tools are assigned to a station, the more station time is thus spent on just changing tools as opposed to useful work. Second, the area taken up by the tools in each station are accounted for on top of the area taken up by the materials required for each task. This is distinct from previous works where only the material area was considered and means that more tool changes in a station result in additional area required as well as tool change times. And third, all the robots can perform all the tasks, but each robot has a different processing rate and tool change time for each task, meaning the evaluation of the best robot for a station must be done carefully. Zhou and Wu solved this problem with a multi-objective immune clonal selection algorithm, a type of meta-heuristic method. Their work is unique in that the contents of a workstation beyond just the station time, in this case the tool area, actively and dynamically change depending on not just the tasks assigned to it, but the types of tasks too. This is seldom explored in the ALBP literature. Additionally, the availabil-

ity of robots and their differences in processing speeds adds an element of the equipment selection problem to this formulation, just without costs in this case.

Problems considering spatial constraints are among the rarer problem types in the ALBP literature but are nevertheless present. Due to this a starting point exists to incorporate ALB analyses that include spatial constraints into aircraft design systems-thinking methods so that they can have more assembly analysis capabilities. As spatial constraints are more representative of reality, especially the reality of limited space for large aircraft parts, their incorporation also allows the results produced using such methods to be more realistic and implementable.

3.2.5 Variable Throughput and Cost Problems

The last material factor to be considered is in regard to variable throughput and costs. The ALBP literature has a very large body of work in regard to this since nearly all the problem formulations involve production rate and cost in some way. Due to this only the ones most relevant to aircraft design systems-thinking methods will be looked at.

Aircraft costs by itself are looked at first, specifically manufacturing costs because those are the most affected by varying production rates without having to look at new technology development. New technology development can indeed drastically increase production rates but would require an in-depth work focused only on new such technologies. This, along with the inherent uncertainties surrounding brand new technologies, causes them to currently be outside the scope of this work, which will be limited to mature or mostly mature technologies. At a high level, aircraft manufacturing costs consist of labor, equipment, tooling, space, and material costs. Based on Hazir et al.'s survey paper, the primary manufacturing costs covered in the ALBP literature include: equipment costs, labor costs, inventory costs, setup costs, incompletion/failure costs, and reconfiguration costs [168]. Despite some nomenclature differences, this means what is available in the ALBP literature maps relatively well with actual aircraft manufacturing costs.

Labor and equipment costs in aircraft manufacturing directly map to their ALBP counterparts. In aircraft manufacturing, setup and incompletion/failure costs are part of the labor costs. Labor is needed to set up the parts to be worked on for each task and labor is also needed to finish unfinished parts or repair/rework damaged and failed parts. Tooling costs, based on how Milner et al. treats them [132], are essentially dealt with as equipment costs under another name but still separate from them. Material costs are included under inventory costs though, as Hazir et al. noted, those are generally rarely covered [168]. Space costs are generally considered an afterthought in the literature due to it requiring spatial considerations, which are rare enough as they are. Reconfiguration costs will not be considered because changing the line balances to dynamically adapt to changing demand is beyond the scope of this work. This thus makes the constituents of aircraft manufacturing costs reasonably well-studied with regards to the ALBP.

The labor costs warrant further attention as they are a cost factor that can be almost directly addressed by the SALBP-E, which is seldom explored in the literature as compared to the SALBP-1 and SALBP-2. This is despite the SALBP-E not explicitly mentioning that labor cost is a variable. As discussed by Kazemi and Sedighi, labor cost is the product of the time a worker spends at their work station and their wage rate. The time a worker spends working at their workstation is equal to the cycle time, regardless of the actual station time [169]. This is all dependent on the assumption a worker does not leave to another workstation to carry out other tasks during the idle time in their own workstation. This is not an unrealistic assumption because workers do not constantly clock in and clock out every few minutes or handful of hours in real life save for purposes such as taking breaks. They get paid even during idle time, hence it is in the line designer's utmost interest to balance the lines efficiently so as much useful work can be obtained for the same amount of wages paid. Total idle time should thus be minimized, meaning total line efficiency maximized, which directly corresponds to the SALBP-E. Another way to look at it is that the more stations there are, the more workers there are and the higher the labor costs. Similarly, a longer

cycle time also means every worker gets paid for a longer period of time, increasing labor costs. Minimizing the product of cycle time and number of stations should thus equate to the lowest number of worker sets working the fewest hours, minimizing labor costs. The SALBP-E can thus be used to address labor costs, at the very least indirectly, and so will be reviewed further.

One of the earlier SALBP-E works is Wei and Chao's [170]. They handled the SALBP-E by iteratively solving SALBP-2s for different numbers of stations. The number of stations is bounded by the theoretical minimum and maximum number of stations, themselves determined by the minimum and maximum cycle times. The minimum cycle time is greater than or equal to the largest task time, while the maximum cycle time is less than or equal to the sum of all the task times. During these iterations the line balances yielding the best line efficiency are found. Wei and Chao used Excel VBA to solve this and showed that one of the main ways to solve the SALBP-E is to iterate through simpler SALBPs.

Zacharia and Nearchou implemented a fuzzy SALBP-E where the task times are stochastic [171]. The fuzzy concept was used because there is variability in the time taken by workers to perform tasks. They importantly noted that there are no direct solution methods for the SALBP-E and that the works thus far have had to iterate through some combination of various SALBPs. Zacharia and Nearchou used a GA to solve their fuzzy SALBP-E. Due to SALBP-Es requiring that the number of stations must be dynamic, and thus not amenable to the typical method of their number being determined via filling up stations to a certain cycle time, Zacharia and Nearchou had a similar idea to Chica et al. [75] where specific genes in their chromosomes decide the number of stations. Their chromosome buildup itself is also unique in that each gene only represents the assignation priority of its respective task. If it is not feasible to assign a task yet, its priority is irrelevant. This guarantees that all solutions follow precedence constraints and are feasible. Their GA involves a two-part cycle where the first cycle's optimal solutions are used to seed the initial population for the second cycle. The novelty of fuzzy SALBP-Es therefore led Zacharia

and Nearchou to have a relatively novel GA solution for it, though they did note that the two cycles caused their implementation to be somewhat computationally expensive.

Jusop and Rashid followed this up with a multi-objective implementation considering resources using NSGA-II, a well known GA [172]. Their objective functions are based on minimizing the number of stations, cycle time, and number of resources. Every task requires a certain type of resource and distributing tasks with different resources over too many different stations results in a large number of resources being required for a line balance. As these resources are expensive and limited in industry, this is undesirable. Jusop and Rashid compared the NSGA-II with several other GAs and demonstrated that the NSGA-II was overall superior for this particular problem. Notably, they did not use the iterative-SALBP approach typically used and instead just opted for a multi-objective optimization of both the cycle time and number of stations.

It would initially appear that the SALBP-E could also be used to do more than just indirectly tackle labor cost and directly address the variable cost and throughput factor; it varies both cycle time, synonymous with the production rate, and the number of stations, which many works closely correlate with cost. This turns out to not be the case. From all the works reviewed it can be seen that the number of stations, by itself, can at most only address equipment costs. However, the SALBP-E is still a SALBP and, due to Assumption 9 in the SALBP formulation, does not actually say anything about equipment selection or its cost. Minimizing the number of stations without eliminating Assumption 9 could possibly minimize equipment cost if the equipment sets' capabilities and individual costs were similar. Despite this, it could also do the opposite if very cheap equipment exists that can only work on one task at a time, thus requiring many stations. No equipment information is actually present in the SALBP-E and nothing can be said regarding equipment costs as a result using just that formulation. Additionally, the SALBP-E, while varying both the number of stations and throughput, does not actually minimize both, which is what is desired. It minimizes their product and maximizes the line efficiency instead, as discussed above.

Given that minimum cost and maximum throughput are likely to stand in stark opposition to each other, with the best in one tending to be the worst in the other, it is extremely unlikely an ideal SALBP-E has either the best cost or production rate. The SALBP-E therefore does not directly address the cost and throughput factors being variable at the same time.

Further SALBP-E formulations can be found in Jusop and Rashid's survey paper though, as can be seen in that work, the SALBP-E literature is very small [173].

The production rate factor is the next to be examined. Although many of the previous ALBPs examined have discussed cycle time and production rate to an extent, most share the same restriction as the SALBPs, which is that the cycle time can be no shorter than the task time of the single longest duration task. This would normally put an upper limit on the production rate; however, if more than one instance of the task were being performed, or more than one copy of a workstation used, then the limit is no longer applicable. This forms the basis for the station paralleling problem. The station paralleling problem revolves around duplicating stations, or tasks in some cases, to reduce the number of unique stations and improve the overall efficiency of the line balance as well as improve production rate [174]. It improves production rate because a station's station time or a task's task time is effectively decreased by a factor equal to its number of duplicates. This removes the SALBP's lower limit on cycle time and therefore the upper limit on production rate. To prevent an infinite production rate with no drawbacks, which is unrealistic, the increase in production rate is always checked by a constraining factor: cost. The cost in this case comes from the cost of duplicating the equipment as well as the cost of opening up another workstation and having the additional operators work in it, with particular emphasis on the equipment [174, 175, 160]. It means the station paralleling problem must essentially always require consideration of equipment as well.

Bard authored one of the earlier works regarding parallel stations [174]. He noted that, without station duplication, there is an implicit assumption that facility costs are not tied to specific tasks and as a result tied to just the number of stations. This is why minimizing the

number of stations is often tied to minimizing costs. With station duplication, such costs must be accounted for accurately. He especially noted that if the total idle time is equal to or greater than the cycle time and the number of stations is greater than the theoretical minimum of total task time divided by cycle time, then paralleling stations could have benefits. Bard duplicated both tasks and stations whenever paralleling is desired as opposed to duplicating just tasks, like some of his predecessors. This necessitates the duplicating of both labor and facility costs. Despite this, he only allowed one duplicate of each station. Bard solved his formulation with a dynamic programming algorithm.

Ege et al. created a parallel station formulation without paralleling limits [175]. In order for tasks to be assigned at a workstation the requisite equipment must also be present at said station. As a result duplicating stations also duplicates their costs. The cost of opening up a new workstation on top of the duplicated equipment costs is also considered, with the objective function being to minimize equipment and station opening costs. They tackled this problem with a branch and bound method.

Leiber et al. have one of the more comprehensive formulations of the works reviewed and looked at incorporating assembly subgraphs, parallel stations, and resource selection, which is akin to equipment selection [176]. The objective function is to minimize total cost, which is comprised of resource costs and station costs associated with the number of stations. Their own version of an assembly subgraph is used to provide precedence information and allows them to select alternative processes with different task times and resource requirements. Duplicating stations is only allowed if the highest task time is greater than the cycle time. Leiber et al. explicitly stated that, as duplicating stations doubles a station's cost, it is necessary to always have a cost-oriented objective and focus. They also noted that a very simple alternative to scenarios where the max task time is greater than the cycle time is to just double the entire line, which would double the total costs as well. Station paralleling stands in contrast to this as being much more precise because it only targets bottleneck tasks and stations, improving the line without doubling

the cost, though if every task and station needed duplicating, then creating another line is simpler. They used a GA as their solution approach and are one of the few to do so for assembly subgraphs.

Station paralleling is an important factor not only in increasing throughput but also improving line efficiency. It was already mentioned several times in previous ALB sections when discussing other types of ALB problems and so is a necessity when looking at production rates on top of varying the cycle time.

The final material factor required is in considering the cost and throughput factors simultaneously. It was already mentioned that the SALBP-E is unable to account for this. Additionally, as demonstrated by the station paralleling problem, cost and throughput have an inverse relationship: increasing one tends to decrease the other and vice versa. The only option available to optimize both simultaneously in this situation is for the optimization to be multi-objective. That those two objectives have an inverse relationship means there will be solutions that are superior in one metric and inferior in the other, causing them to be incomparable. Essentially, it is impossible to optimize both at the same time and have a definite, single "correct" and "best" solution; tradeoffs must be made and a multi-objective approach is the only choice. Many multi-objective works have already been reviewed. A few more notable ones will also be discussed.

McMullen and Frazier authored one of the very few multi-objective works where production rate, in the form of cycle time, and cost are the primary objectives [177]. Their formulation is not only multi-objective but incorporates parallel stations, stochastic task times, and multiple products. The stochastic task times is implemented by using a random number generator to create the task times, with a 75% chance that the task times are picked from one uniformly distributed interval and a 25% chance from another. Stations are duplicated whenever the station time is greater than the cycle time. The cost of each station, which includes labor cost and equipment cost, is duplicated whenever the station is duplicated as well. Each task requires its own equipment. McMullen and Frazier used simulated

annealing to find solutions to their formulation and compared it to solutions obtained via numerous other objective heuristics. They found that simulated annealing performed well in obtaining the best throughput solutions and average in finding the best cost ones. Particularly noteworthy is that, since their work is one of only a few looking at both throughput and cost, they were able to demonstrate that optimum throughput came at the cost of not having the optimum cost. They could not consistently find solutions that performed well in both objectives and observed that the best throughput solutions required more laborers and equipment from station duplication, which drove up cost. Their simulated annealing approach performed better, on average, than the other approaches when both objectives were important. Despite multi-objectively considering throughput, the cycle time is not varied dynamically but pre-selected.

Oesterle et al. made one of the more complete multi-objective formulations in the ALB literature [76]. They incorporated assembly subgraphs with alternative processes, variable task times, equipment/resource selection, area considerations, and more than one objective in the form of unit cost and idle time. The only major relevant factor missing is station paralleling. The calculation of the cost objective is more holistic than in most works and includes the labor, overhead, resource, tool, energy, facility, and material costs. This total cost is divided by the product's gross volume to obtain unit cost, with gross volume being the total number of products made accounting for rejection rates. Oesterle et al.'s work is also very extensive and compares an enormous number of different multi-objective metaheuristic algorithms against each other on many different problem sets. GA and ACO-related methods on average had the best performance.

Takai's work is unique in that it combines ALB with not just ASP but product design as well [37]. An augmented liaison diagram is first used to establish precedence relations among all the tasks. This augmented liaison diagram is used to find alternative product designs, which actually end up being very similar to subassemblies or alternatives in an assembly subgraph. All feasible assembly sequences for all identified product designs

are then generated. Afterwards, line balancing is performed for every feasible sequence. Upper and lower cycle time limits are determined based on the sum of all task times and the highest individual task time, respectively. The cycle time is incremented from the lower to the upper limits, with line balances obtained at each increment. Tasks are placed into stations until the station time is about to exceed the cycle time, at which point a new station is opened. Up to 4 parallel lines are able to be opened if desired. After all this is done, the best solutions in terms of profit is identified, represented as the difference between the revenue and the sum of the material cost, labor cost, subassembly cost, mold cost, and line cost. Takai's work is thus multi-objective in the sense that all costs for nigh all possible throughputs are obtained. He used Microsoft Excel to implement and solve his formulation. This is appropriate for his use case that only has 8 distinct tasks but would not scale at all with larger problems, on top of the fact that a line balance is performed for every single feasible sequence. Nevertheless, his work is one of the few that tries to address the needs of IPPD, despite him not calling it that, by incorporating product design with ALB and ASP.

Multi-objective formulations overall abound in the ALB literature. They generally involve calculation of all sets of objectives, effectively allowing them to integrate multiple different problem types. They also involve the usage of weightings for the objectives, metaheuristics that search for non-dominated solutions, or both, to determine the set of the best solutions. Said objectives that tend to be optimized include cycle time, number of stations, cost, smoothness index, and efficiency. Many ALB multi-objective works were reviewed not just in this section devoted to them but also in previous ALB sections concerning the other ALB problem types. Indeed, almost all of the integrated ASP ALB works in section 3.1 were also multi-objective. As so many were already covered, no more will be examined.

The ALBP literature is extremely large, perhaps even larger than the ASP one if Battaia and Dolgui's survey paper is any indication [178]; there are many more ways to combine problems and subproblems and real world phenomena since ALB is further along in the

assembly design process and so closer to reality. This is why the review of ALB was much more targeted than the one done for ASP. This is on top of the fact that specific material factors are desired for the former while general capabilities are desired for the latter. With that done, the general flow of how ASP works has now been identified and can be used to connect geometry considerations from aircraft design directly with assembly analyses. There is now a bridge to see how geometry being changed on the aircraft design side to optimize aircraft performance can affect the order, feasibility, and accessibility of how parts are put together on the assembly side. Similarly, ALB problem types have all been identified for the material factors of integral fabrication, manufacturing process selection, spatial constraints, and variable throughput and cost. These can now be used to allow further material system choices to improve aircraft design structures and performance to be propagated to the assembly side to observe larger and more traceable impacts on production rates and cost than before. All these formulations need to be combined and integrated into one framework. The literature just reviewed has shown that this is possible with its many integrated ASP and ALB works. There is even one work that tries to integrate product design considerations with both ASP and ALB, though it does not go into as much detail regarding product design as desired. The path forward is clear and, at the minimum, feasible: ASP and ALB problem formulations and different works' approaches on how to solve them are to be integrated with aircraft design systems-thinking methods to augment them with additional feedback loops between the aircraft design and assembly disciplines in order to obtain higher production rates cost effectively.

3.3 Implementation Issues with Assembly Sequence Planning

The reviewed ASP and ALB literature has shown it is possible to combine those analyses with aircraft design systems-thinking methods based on their ability to be combined with other analyses, but it does not specify how. And in the implementation of a framework integrating all those things together to address the overarching research question, that lack

of specification leads to issues in trying to incorporate both sets of assembly analyses. This section is thus specific for ASP-related implementation issues and the next for ALB-related ones.

The most straightforward way to connect ASP with aircraft design is to take geometric data from the latter and use it as input when carrying out the former. There are currently three different ways for the geometric information from aircraft design to be communicated with assembly sequencing in such a way that the latter is able to understand the inputs. They are question-and-answer methods similar to what is done with Bourjault and De Fazio's works [85], CAD-based methods where a detailed CAD model is used to extract geometric details and features, and knowledge-based methods where data on similar geometries is already stored and can be utilized. However, herein there is an issue. Knowledge-based methods are only effective for standard geometries; if a non-standard geometry or a non-standard situation appears, usage of the other two methods is needed to fill in the gaps. Creation of the knowledge base itself also requires usage of the other two methods, as a knowledge base cannot self-create, making it a nonviable standalone method. This leaves only question-and-answer and CAD-based methods.

Question-and-answer type methods require very detailed answers from a subject matter expert who knows the intricacies of the assembly being designed. However, aircraft configuration changes can only be made before the detailed design phase. Before then, the subject matter expert is unlikely to know said intricacies to answer the detailed questions at that point in design, and it is a possibility that there may not even be a subject matter expert that early on. Another drawback is that the detailed questions asked for these methods are of a similar nature to the ones asked in the traditional DFA methods, such as the Boothroyd-Dewhurst method. Question-and-answer methods are therefore just as susceptible to having their responses being subjective and highly dependent on the expert being consulted, with different experts possibly giving different responses for the same product [179]. With as many configurations, geometry, and other factors being considered, it is

undesirable to also include subjectivity. Additionally, as mentioned in many works, the question-and-answer method of providing detail takes an inordinate amount of time, up to multiple hours per design, which is undesirable in a design phase where designs should ideally be quickly sifted through and evaluated. Question-and-answer methods are thus also not a viable geometric information input method.

This leaves just CAD-based methods. These methods are the most attractive not only because they are more objective than question-and-answer type methods, but because they can easily generate interference matrices which, as noted by Zhang et al., are simple to perform calculations with and widely used throughout the literature [124] Despite this, nigh-all CAD-based methods and works in the current literature focus only on CAD models generated during the detailed design phase. These methods thus expect there to be enough geometric detail in the model to accurately extract the features and mating information for the geometric reasoning. However, available CAD information during the early aircraft design phases is highly variable, depending on exactly which phase the information is being extracted from. More detailed CAD information is present the closer to detailed design the phase is, while the opposite is also true. Indeed, in some early aircraft design phases, there are no CAD models at all. If there is too little information or none at all, ASP cannot be done. This is made more complicated by the fact that the ability to make major aircraft design trades is desired, which is only available closer to the earliest phases where there are no CAD models. There is thus ambiguity on where the geometric information is to come from on the aircraft design side: too close to detail design and no major aircraft trades can be made, and too close to the earliest design phases and ASP cannot be performed. No works currently address this issue.

Similarly, on the ASP side, there is ambiguity on the appropriate techniques to process said geometric information through geometric reasoning. As can be seen in subsection 3.1.6, there are a large variety of geometric reasoning techniques, each of which require a specific level of detail and effort. The issue of the level of detail was already presented,

but it turns out the level of effort can also cause problems. Some of the geometric reasoning techniques are complex and computationally intensive, which is not desired because a large number of designs must be iterated through for aircraft design. Techniques that take too much time will severely stifle the aircraft design space exploration. As a result, not only is there a question of what level of detail the geometric information coming from aircraft design should have, there is also a question of what techniques should be used afterwards to process that information such that it is still feasible to iterate through a large number of designs. Since there are currently no methods that detail how to address these tradeoffs, assembly sequencing and aircraft design are unable to be immediately integrated. This leads to Implementation Gap 1, named so because it is a gap in capability encountered in the attempted implementation of the framework proposed by the overarching hypothesis:

Implementation Gap 1: No guidance exists on the appropriate level of detail with which to perform assembly sequence planning on aircraft assemblies with considerations for early aircraft design.

3.4 Implementation Issues with Assembly Line Balancing

The ALB literature has problem formulations and solution approaches for all of the desired material factors to be integrated with aircraft design systems-thinking methods: the assembly subgraphs problem for integral fabrication, the equipment selection problem with zoning constraints for variable manufacturing processes, the TSALBP for spatial constraints, and the station paralleling, multi-objective, and SALBP-E problems for variable throughput and cost. On top of this, integral fabrication dealing with what are essentially subassemblies means that ASP can be used to represent them as well, in addition to the assembly subgraphs problem. And as reviewed in section 3.1, there are ALB problems that simultaneously incorporate ASP too to handle even that. All the ways to tackle the individual material factors are available as an ALB problem type.

However, what is needed is for all of these material factors to be present at the same time, and no works or formulations in the ALB literature are able to do this and consider all the various desired problem types simultaneously. The ALB is also used to provide resource considerations for the ASP to allow throughputs to be outputted while examining different assembly sequences. And on the flip side, ASP is used to provide the precedence constraints for ALB with every different sequence resulting in different precedence constraints, making them intricately linked. If ASP and ALB exist in the same formulation, they must be analyzed together. This means all the various ALB problem types need to be integrated together with ASP too. Even in the relatively comprehensive survey papers done by Battaia and Dolgui and Hazir et al., no ALB works can be found that are able to include all of this in one formulation [178, 168]. The most complete work that includes almost all of the necessary problems and factors is Oesterle et al.'s, which still lacks that station paralleling problem and does not incorporate ASP [76]. The assembly subgraphs problem inclusion in their work is able to serve as somewhat of a proxy for ASP, since the subgraphs are essentially a stand-in for different assembly sequences, but subgraphs cannot handle all the actual sequences considered in a typical ASP problem. This means their formulation and solution approach is close but ultimately insufficient for this work's purpose in terms of incorporating all the needed problem types.

There are several other reasons that Oesterle et al.'s work, and all other works in the ALB literature, prove to be insufficient. One of them is concerning spatial constraints. The current formulations and solution approaches for TSALBPs only examine the space requirements for a single workstation, checking to make sure that the space needed for all the tasks in a station do not exceed the station's available space. For aircraft factories workstation space is not nearly as important as factory space. The limit is not on whether the tasks in a station can fit in said station, but on whether all the tasks in all the stations can fit in the available space in a defined factory. This has not generally been an issue in the ALB literature because it only comes up when station paralleling is combined with spatial

requirements, which has not been done in any works yet, or when the product architecture is being actively changed while performing ALB, which has also not been explored in any works yet. However, it is pertinent to the problem at hand because station duplication is being combined with space requirements while the product geometry is being extensively changed too. To the author's knowledge, this form of ALBP with spatial constraints for the entire factory as opposed to just a workstation has never been addressed yet.

The second reason is that there are no works with considerations on how to arrive at a physically feasible upper limit on throughput or what its implications are. Because the goal of the multi-objective optimization is to provide the designer with as comprehensive a picture of the design space as possible, two important data points of interest are the design with the minimum cost and the design with the maximum throughput. They are important because they allow the designer to see the full capabilities of their factory in terms of both cost and throughput. They provide the designer with insight as to the sort of aircraft performance and manufacturing tradeoffs to expect should they absolutely prioritize one objective over the other. From there, the designer can decide for themselves which objective they should focus on. The minimum cost design is currently easier to obtain because it has built-in physical restrictions from there being a minimum of at least one workstation and one task for every design and line balance. There is currently no guidance on how to obtain the maximum throughput design because maximum throughput estimation has no built-in restrictions when combined with station paralleling. Cost itself is usually used as a constraint, but in this case it is an objective and so using it as a constraint introduces a large amount of arbitrariness and subjectivity into the problem, which is not desirable. All other works in the literature currently have the throughput be set or provided as an input, even in multi-objective works where throughput is an objective like in McMullen and Frazier's [177]. This means how to obtain maximum physically feasible throughputs in a throughput and cost multi-objective problem with station duplication is currently not described in literature and no resultant insights can be made from them.

Another reason current ALB works are insufficient is they do not appropriately consider dynamic task times. With both the assembly sequence and the assembly size and geometry constantly changing, the task times will never stay static and be constantly changing themselves, a topic seldom covered in ALBP works. The assembly subgraphs problem attempts to address this by having process alternatives that have alternative task times, but this implementation is impractical to scale up to include hundreds or thousands of possible alternative designs and geometries each with hundreds or thousands of possible sequences, all of which have tasks with unique task times. On top of this, these dynamic task times are not known a priori, as assumed in the assembly subgraphs problem. There are only a few works in both the ASP and ALB that address this by using various techniques to obtain scalable implementations of dynamic task times, such as in Panhalkar et al. and Takai's works, but they are incredibly rare and do not have a formal formulation yet [101, 37]. This sums up to mean that current works in the ALBP literature insufficiently address dynamic task times.

The fourth reason that current ALB works do not provide sufficient capability is that they are unable to appropriately implement integral fabrication. The assembly subgraphs problem attempts to do this with alternative subgraphs in a precedence graph. However, as previously discussed, this already breaks down when incorporating just regular assembly sequences. It breaks down even further when integral fabrication is combined with assembly sequence planning and essentially adding consideration of subassemblies into the mix. This creates far too many combinations for a precedence graph with alternative subgraphs to ever be able to feasibly represent. Another problem is with the integral fabrication concept itself: the tasks involved with fabricating integral parts are generally mixed in with other fabrication tasks. Assembly line balancing is primarily focused only on assembly tasks. When performing line balancing, then, it is unclear where to draw the line on what tasks should be included such that line balances with and without integral fabrication can be compared on a one-to-one basis. As an example, a line balance without integral fabrica-

tion only needs to include tasks that occur after all fabrication has finished. When normally separate parts such as a spar and its stiffeners are integrally fabricated instead, not including the integral fabrication tasks would artificially reduce the amount of resources needing to be modeled, giving that line balance an unfair advantage; but that fabrication task has many other fabrication tasks that are dependent on it, such that including it and not including its dependent tasks could break the flow of the fabrication factory. And on top of that, there are many different sets of parts and combinations that could end up integrally fabricated, so a one-off exception will not suffice. This is not addressed in the current literature.

The last reason is related to the fourth reason and is that ALBPs typically do not include fabrication tasks in their modeling. It is desired to compare different materials and manufacturing processes to see which yield the most benefits in terms of cost, throughput, and performance. This can be partially captured by modeling the differences in each process's assembly tasks and using the equipment selection problem. However, a large portion of what differentiates manufacturing processes and their material handling is located in the fabrication tasks. Without including those, the comparisons of different processes may not be necessarily on a one-to-one basis. A process with expensive assembly activities could have cheap enough fabrication activities that it is still worth using. By not modeling fabrication tasks, as is currently the case for nigh-all ALBP formulations, this is impossible to account for.

All these insufficiencies combined together yield Implementation Gap 2:

Implementation Gap 2: No works in the assembly line balancing literature sufficiently and simultaneously integrate problems and characteristics related to assembly sequencing, integral fabrication, variable manufacturing processes, and variable throughput and cost for the purpose of performing aircraft design and cost-effectively increasing aircraft production rates.

3.5 Implementation Issues Regarding Both ASP and ALB

Both ASP and ALB are NP-Hard problems whose simulation times grow ever-larger the bigger the problem becomes. Using an aircraft wingbox as an example for ASP, an extremely simple wingbox composed of only two spars, two skin panels, two ribs, and just two stringers per skin panel yields over 3.5 million possible assembly sequences to evaluate. Attempting to solve just one NP-Hard problem by itself is a tall order and can take a substantial amount of time. Solving two at a time in the form of integrating ASP and ALB can take even more time and is even more liable to a combinatorial explosion if not done with sufficient care. In this work it is desired to not just solve an integrated ASP and ALB problem but to do so along with performing aircraft design, which itself requires a large number of iterations. The integrated ASP and ALB problem may need to be solved hundreds or even thousands of times during the process of converging on an ideal set of aircraft design and geometry parameters. Complex implementations of said integrated problems can take up to half an hour to multi-objectively optimize per set of categorical variables, according to Rashid et al.'s work, of which aircraft design has quite a few [180]. This means that the desired assembly analyses taking too long is a major issue. If the assembly analyses take a prohibitive amount of time, it becomes infeasible to explore the aircraft design space for the best designs. Currently, no works have addressed or provided guidance on how to implement the assembly analyses incorporating considerations for the time limitations involved with needing to perform them a very large number of times. None have needed to in the past due to not explicitly performing architecture design while performing assembly sequence planning and line balancing, but it is a capability that is needed now. This leads to Implementation Gap 3:

Implementation Gap 3: Current methods in literature insufficiently describe how to efficiently handle the integrated assembly sequencing and line balancing problems in the context of performing them many times for the aircraft design process.

3.6 Chapter Summary

The guiding questions were used to perform a deep dive into the ASP and ALB literature. An extensive review of how ASP is carried out and its many representations and its current state-of-the-art was performed to see what kind of capabilities it is able to provide. Afterwards, an extensive review of the relevant ALB problem types was also performed to see what its capabilities and state-of-the-art are. In the process of doing this, the problems and gaps with the implementation that need to be addressed to answer the overarching research question within the scope of this work were identified. This is all summarized and shown in Figure 3.19.

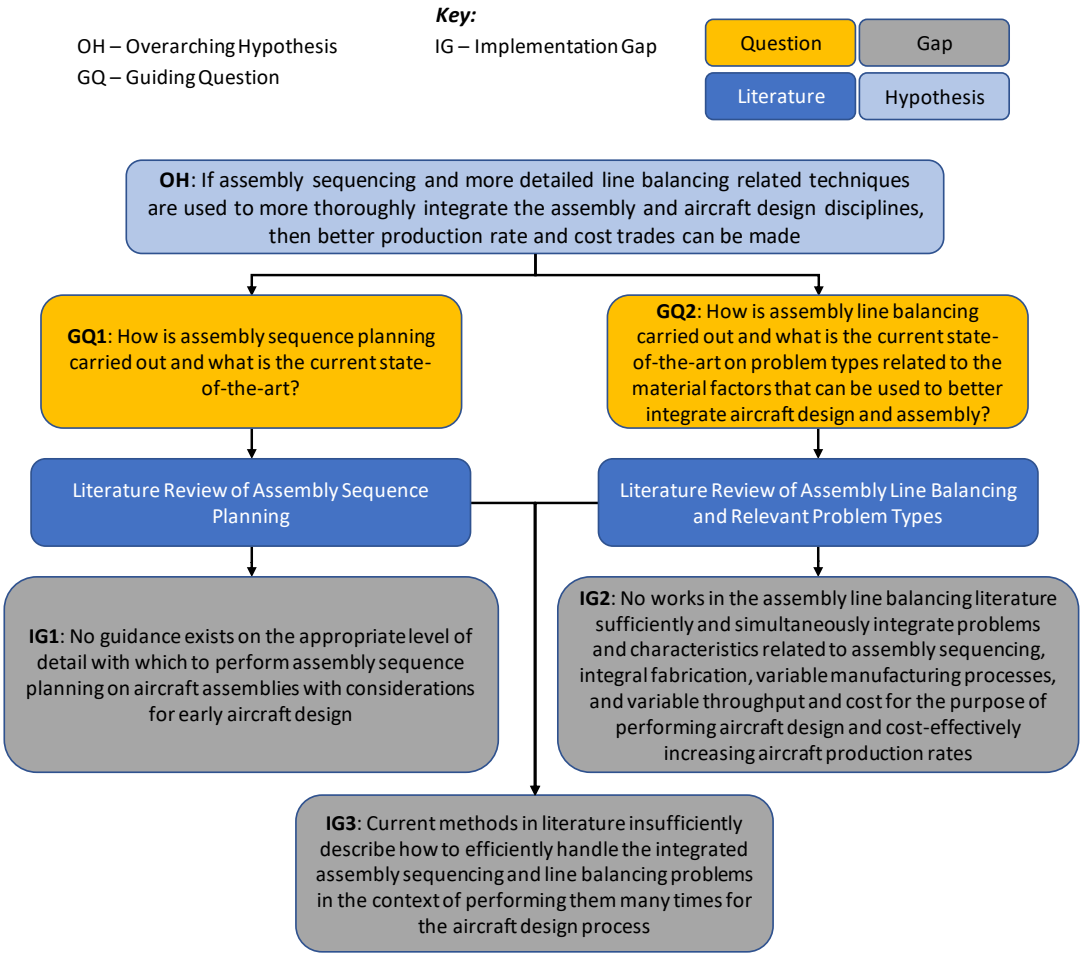


Figure 3.19: Summary Logic Diagram of Literature Review and Implementation Gaps

In order to create the framework that can answer the overarching research question, implementation gaps one through three must be addressed. Filling the first one allows geometric information from the early aircraft design discipline to be properly used for assembly sequence planning. Filling the second one allows the material factors to further enhance the tradeoff capabilities of assembly analyses that operate on variables common to both aircraft design and assembly. And filling the third one allows this process to happen quickly enough to be able to actually explore the design space in a reasonable amount of time. The next few chapters go over how this is all individually addressed before ending with the finalized realization of the proposed framework in Chapter 7.

CHAPTER 4
INTEGRATING AIRCRAFT DESIGN CONSIDERATIONS WITH ASSEMBLY SEQUENCE PLANNING

4.1 Research Question 1: Data and Technique Fidelity for Integration

To combine aircraft design with ASP, geometric information from the former is used as input when performing the latter. CAD-based methods have been identified as the most favorable method to perform this data transfer, being able to take geometric information that describes the vehicle in aircraft design and translating it into assembly information through the process of geometric reasoning. To reiterate, geometric reasoning is the process of using geometric information to infer or create new relationships and solve problems. According to Wilson and Latombe, and specific to the assembly discipline, geometric reasoning involves using said geometric information to determine the order to assemble or disassemble an assembly, the degrees of freedom the assembly and its parts have, and what parts should be removed to create a particular subassembly, among its many other uses [119]. The primary impediment to integrating aircraft design with ASP at the moment is there is little to no guidance on what fidelity the geometric data and reasoning techniques should have. Higher fidelity data ensures that ASP will be able to be performed but is only obtainable later in the aircraft design process when trades have already been locked in. And higher fidelity techniques allow for more grounded geometric reasoning, but are more computationally intensive and can make aircraft design space exploration in early design infeasible. The appropriate fidelity levels must be identified before the integration can occur.

To further define the problem, the concept of fidelity must first be better established. As summarized by Cox in his thesis, fidelity is most simply defined as how much a model represents the real system. The fidelity of a model—and the disciplines of interest are indeed

forms of modeling, since they do not involve physical products being realized every step of the way and for every option explored—is argued to be composed of its three different aspects: scope, abstraction, and resolution. Scope is the "amount of the overall system" that is included in the model, abstraction is the degree to which the actual system's behavior has been simplified to allow it to be modeled, and resolution is the level of detail of the model or how much it "resembles the actual system" [181]. In terms of aircraft design and ASP, the scope refers to how large and what type the assemblies are, determining which systems are analyzed. The level of abstraction refers to the logic of the problem such as what assumptions are made in the handling of the assemblies, determining how realistically the assemblies behave and how complex the solution procedure is. And the level of resolution refers to how finely the details in the assemblies can be resolved, determining how small the smallest parts are as well as the type of geometric and material information encoded in them.

An insufficiently large scope would prevent any early aircraft design trades from being made but allow for very focused assembly sequencing analysis, while an overly large scope would allow for many early aircraft design trades but possibly overwhelm the ASP if the abstraction is too low and resolution too high. Too little abstraction yields very realistic and precise geometric reasoning but is very computationally intensive, while too much abstraction could be too unrealistic to be useful. High resolutions allow for the largest number of ASP tradeoffs but are only possible in the last aircraft design phases, while very low resolutions allow the largest number of aircraft design tradeoffs but can make ASP unusable. The goal is to thus find a "sweetspot" wherein the fidelity described by these three aspects is high enough to carry out ASP while retaining a sufficient amount of computational speed and occurring early enough to explore the design space in early aircraft design. Because the integration problem of interest has never been tackled before, the levels of fidelity that allow this are unknown. This leads to the first research question:

Research Question 1 *What fidelity of data and techniques should be used in the geometric reasoning to enable assembly sequence planning while retaining early aircraft design tradeoff capabilities?*

4.2 Research Question 1.1: Input Data Fidelity

Perhaps the easiest aspect of fidelity to tackle for the integration problem is the scope. Traditionally, the assemblies addressed by ASP have been ones that only exist in the detailed design phase of a product. This includes gear assemblies [103], subsystems like the electronics and pipes in an aircraft nosecone [23], transmission assemblies [85], electronic controller assemblies [127], and motor drive assemblies [113], among many others. Although these assemblies play an important role in the final physical realization of the product, they are of a different scale than the primary parts and assemblies typically addressed during early aircraft design. Most distinctly, these small assemblies, if they were all actually part of an aircraft, would generally have little to no effect on its overall performance, which is the focus of early aircraft design. The scope therefore needs to include assemblies with a more pronounced effect on aircraft performance.

For an aircraft, this means the scope must include major assemblies such as its wingbox, wings, fuselage, and tail surfaces. These assemblies, during the early aircraft design, determine critical parameters like the aircraft's lift-to-drag ratio, range, endurance, center of gravity, stability, and the majority of its weight. Performing assembly sequencing on these assemblies would thus yield both manufacturing and aircraft design changes. As an example, if, during the sequencing analysis, it was determined that a particular spar or wing skin shape caused the whole wingbox assembly to be infeasible, the redesign of the shape of those components would affect not only the cost of the wingbox but also its aerodynamic and structural performance due to those parameters being sensitive to those components' shapes. To date, some of the only works that have incorporated the overall desired scope

into their assembly sequence planning methods by looking at an aircraft wingbox are those by Mantripragada and Whitney, Wang et al., and Zhanlei et al. [94, 182, 128]. Mantripragada and Whitney and Zhanlei et al.'s works were described in subsection 3.1.6 and revolve around determining the optimal sequence to assemble a wingbox's components so that its tolerances could be as tight as possible. Meanwhile, Wang et al.'s work performs assembly sequencing on a wingbox to determine the best way to assemble it such that it could be located properly with fixtures. These works' analysis of a wingbox to determine its optimal assembly sequence is thus a step in right direction in terms of scoping.

Of note is that although the tolerancing and fixture design considerations in the three discussed works are not necessarily out of scope in terms of fidelity for combining aircraft design and ASP, they are out of the scope of this work. This is because, as discussed in section 2.3, they require more information than is available in early aircraft design and so are not pertinent to this work. Regardless, they do make clear that the best scope to consider involves determining the feasibility of major performance-defining assemblies, which includes the wingbox. From here on, this will be considered the correct scope, that is, the scope includes the geometric reasoning of major aircraft assemblies like the wing, wingbox, fuselage, and tail surfaces.

With the scope settled, the abstraction and resolution of the geometric reasoning techniques and data are addressed next. The fidelity of geometric reasoning techniques is based almost entirely on their level of abstraction due to their resolution being essentially synonymous with their abstraction. This is a result of them only being concerned with how parts are moved around to be disassembled or assembled and, therefore, how realistic and how many assumptions there are regarding those movements. Increasing the resolution merely increases the level of real life interactions captured, which is the same as decreasing the abstraction. Deciding on the abstraction of geometric reasoning techniques therefore also effectively decides its resolution.

Similarly, the fidelity of the data inputted into ASP is based almost entirely on its res-

olution. Primarily geometric data is being transmitted from aircraft design to ASP for geometric reasoning and it does not make sense to assign levels of "physics" or abstractions to it; there is no way to make a shape like a triangle more realistic. This results in the fidelity of geometric data being most and best described by its resolution.

In terms of which to tackle first, geometric reasoning techniques operate completely independently from the resolution of the parts in the assemblies themselves; a part with almost zero resolution can still be moved around and feasibility checked for using the exact same techniques that also apply for a part with very high resolution. The quality of the geometric reasoning's results thus ultimately depends on the resolution of the parts and the assemblies being analyzed. Even the highest abstraction geometric reasoning techniques can obtain usable results given an assembly with a high resolution, while no results are possible even with the lowest abstraction techniques if the inputted assembly has zero resolution and basically no parts to analyze. This makes it critical to have the correct resolution as it functions as the primary bottleneck.

This brings up the question of what the resolution of the major assemblies outputted by the aircraft design discipline should be. Outputs from the detailed design phase have normally been used. However, it has already been established that the detailed design phase is not appropriate because design changes can no longer be made in that phase. Early aircraft design has been proposed as the answer, but there are two design phases that could be considered as such, conceptual and preliminary design. Each has its own benefits and drawbacks in regard to outputting CAD models for the CAD-based geometric reasoning. Conceptual design allows for more aircraft designs to be explored due to the large amount of design freedom and ability to change variables. However, the risks include seemingly optimal sequences becoming infeasible later when more detail is added due to the low initial resolution leaving important shape information undefined and no sequences being able to be generated at all. Meanwhile, preliminary design can provide models with parts that have resolutions nearly as high as detailed design, meaning the possibility that

sequences will turn infeasible when more resolution is added is minimized. Despite this, preliminary design can, similar to detail design, lack the desired design freedom and take nearly as long to evaluate each design. The decision thus comes to a speed versus capability tradeoff.

"Capability" is preferred over "accuracy" because, given the nature of the different design phases, accuracy can always be increased; detail design always has higher accuracy than the other phases, whether in the form of additional embedded geometric information or increased number of parts that are not able to be present earlier on. This does not mean the earlier phases are useless for ASP because this is true even in aircraft design, where some of the variables determined in the earlier phases also need to be refined in the later ones, and the earlier phases are not useless there. The purpose of the earlier design phases is not to provide unwavering, unchanging accuracy but to give as representative a concept of what comes later based on the data available while exploring all possible options. In aircraft design, for example, the wing's shape determined early on is representative of its shape at the end of design but not exactly the same. Small changes such as slightly increased area to account for high lift systems, landing gear, and winglets are made to accommodate other requirements encountered in the later design phases. That these changes occurred does not invalidate the earlier design phases. The same can be done for ASP. Using early aircraft design phases as inspiration, for the earlier phases to be similarly representative for ASP, the parts that exist in whichever phase is chosen should have sufficient resolution to minimize the chances of their sequences becoming infeasible as compared to sequences of the exact same parts in later design phases. In other words, if a feasible sequence of five parts were made in conceptual design then, to be representative, enough geometry data would need to be present to minimize the possibility that a sequence of the same parts but with data from detail design turns out to actually be infeasible. The result from this is that capability is preferred over accuracy and is measured in terms of the chances of sequences being correctly marked feasible or infeasible as compared to the same design and parts in

the detailed design phase and the ability to make sequences in the first place.

This all leads to Research Question 1.1:

Research Question 1.1 *What resolution should the geometric reasoning inputs have to capably perform geometric reasoning while still being able to quickly explore the aircraft design space?*

4.2.1 Aircraft Conceptual Design

The desired resolution is one that can accommodate major configuration changes that still provides the stated necessary capabilities: lower odds of becoming infeasible later and generation of sequences in general being possible. In order to select the design phase with the best resolution to use for geometric reasoning inputs, the design phases themselves must be reviewed and their relevant characteristic traits examined to see which is most suitable for the purpose of integration of the disciplines.

As defined in section 2.1, aircraft design is split into the conceptual, preliminary, and detailed design phases. In the conceptual phase, requirements from the customer and regulatory agencies are used by the designer to determine the aircraft's configuration, set its size and weight, and evaluate its performance [30]. It is in the conceptual phase that the aircraft's overall geometry and equipment are determined. In the preliminary design phase, the aircraft's configuration can no longer be changed so that specialists in disciplines like structures, controls, and aerodynamics can perform more in-depth studies and further define the aircraft. The goal of the preliminary phase is to have enough information about the aircraft to decide whether it should proceed into the next design phase and the overall design committed to [30]. The last phase is the detailed design phase, where the fine details of the aircraft are ironed out. This includes analysis of all the minor pieces and components that were not looked at during the preliminary phase, prototyping and testing of the aircraft's systems, as well as analysis on how to produce every single part in the aircraft

[30]. The goal of the detailed design phase is to prepare the aircraft for production, and as such it is generally during this phase where fabrication and assembly considerations are taken into account and where methods, including the conventional and conceptual Design for Assembly methods, are applied.

The earliest possible design phase in the aircraft design process, and design processes in general, is the conceptual design phase. In this phase, the overarching requirements from stakeholders are translated into performance metrics that are then used to define what the airplane will look like and shape its form. According to Anderson, this amounts to setting the plane's shape and weight and determining the aircraft's major configuration details. This includes selecting the wing's shape, sweep, taper, and location, the fuselage's size and shape, and the tail surfaces' location, size, and shape [183]. This phase therefore has the largest amount of design freedom possible due to its goal of selecting the best possible aircraft configuration from an almost limitless number of combinations. The emphasis on defining the aircraft's shape in particular makes this phase an attractive option to integrate with assembly sequencing. This is because, on the design side, the geometric feasibility information from assembly allows the designer to filter out designs and shapes that may be impossible to manufacture and allows them to refine their search and only look at more feasible designs. Meanwhile, on the assembly side, it prevents the sequencing analysis from being too short-sighted and end up selecting the optimal sequence for a sub-par design. Both disciplines benefit. The main draw of the conceptual phase is thus its ability to explore all designs possible and reducing the possibility of ending up analyzing a design that is only, in optimization terms, a local minimum instead of a global one.

However, the tradeoff with conceptual design is that the resolution of the analyses performed in this phase is quite low and able to go to almost zero. This is done so that as many designs can be iterated through as quickly as possible. According to Anderson, the primary calculations performed in this phase are related to "aerodynamics, propulsion, and flight performance," and even then only their relatively basic, non-computational versions

are used. Structural analysis, the area most applicable to assembly sequencing due to its determining and sizing of the individual parts and their shapes, is "not dealt with in any detail" [183]. Though the overall shape of the aircraft such as its wing and tail configuration, sweep, taper ratio, and thickness-to-chord ratio are determined in this phase, higher resolution geometric information like the shape of the spars and ribs and stringers, or even their very existence, are generally absent from conceptual design. This means that for the majority of conceptual design there are little to no parts to even perform assembly sequencing with.

At most, according to a picture description from Raymer, the only structural definitions provided in this phase are extremely basic outlines and locations of the major parts such as the spars and wingskins [30]. The lack of any significant structural sizing causes the basic outlines of the major parts to generally be represented by simple flat plates, belying the fact that their geometries are still flexible and likely to not stay flat plates. The subsequent lack of geometric detail associated with those structures makes them liable to become infeasible when more geometric detail is inevitably added later on. This is because the parts are sufficiently underdefined that there is a high chance that the sequences made with the parts represented by the flat plates will interfere with one another when they are no longer just flat plates. This is not even considering the parts only present in the later phases. More geometric definition is needed such as number of spars, part spacing, part shape, etc., which is not available in conceptual design.

Overall, while the conceptual design phase is able to traverse through a myriad of aircraft designs, it lacks the necessary resolution to provide any meaningful input for the assembly sequencing. It offers the absolute lowest resolution possible which, it turns out, is too low due to the lack of capabilities. This makes the conceptual design phase an inappropriate phase to provide geometric input for assembly sequencing, despite its many other benefits.

4.2.2 Aircraft Preliminary Design

Sequentially, the next design phase after conceptual design is preliminary design. It is in this design phase that the majority of the structural sizing and analysis is performed. As a result, there are a variety of different resolutions in this phase that can be considered depending on how detailed the structural sizing and optimization analysis is. Starting in this phase is typically when Finite Element Analysis, or FEA, is done for more detailed sizing of the structures. The many different ways to model a structure with FEA is why this phase has such variable resolution. A wingbox can be represented as being composed of a handful or even only a single spar, which itself is represented by a one-dimensional beam or bar element; or it can be represented as two wingskins and several spars, each represented by two-dimensional shell elements that may or may not have the skin or spar's cross-sectional shape information embedded inside; or as a two wingskins, several spars, a large number of ribs, and a large number of stringers attached to each, all of them represented as three-dimensional solid elements with an explicit shape and thickness; or any of the numerous combinations of the options mentioned and many others unmentioned, each with their own resolutions and accompanying calculation speed tradeoff.

Conventional preliminary design is what most designers in commercial aircraft companies think of when preliminary design is mentioned. As described by Raymer, it "begins when major changes are over" and most of the configuration is locked in [30]. This allows structural analysts to size the aircraft's primary structures in greater detail without fear that a design change will cause their work to be for naught. As a result, primary assemblies like the wings, fuselage, and tail surfaces are much better defined and their individual parts given proper geometry. Another key characteristic of preliminary design is that it involves lofting, defined by Raymer as the modeling of the aircraft's outside skin to have its actual thickness and done so with enough accuracy that it can be assured the other parts fit together. Additionally, the end of conventional preliminary design, as claimed by Raymer, is marked by the stoppage of even minor design changes and the freezing of the entire con-

figuration [30]. These two factors, combined, mean that, by the end of the conventional preliminary design phase, all components such as stringers, ribs, spars, longerons, skins, frames, etc., are essentially fully sized and defined three-dimensionally. They can thus be modeled in FEA as solid elements and therefore able to be directly imported into a CAD program, where they will have as much geometric resolution and information as the parts normally analyzed in detailed design using conventional assembly sequencing methods. This allows parts from conventional preliminary design to be used as inputs into assembly sequencing.

Raymer states that some of the only structural details lacking in the conventional preliminary design phase that are present in the detailed design phase are items such as access holes, fuel seals, and fittings [30]. Otherwise, essentially the entire load bearing assembly is present by the end of preliminary design. This is supported by the fact that the majority of detailed aircraft cross sections in literature, particularly those of the wingbox, consist of the previously mentioned parts that make up the semi-monocoque structures: skins, spars, stringers, frames, etc. Examples can be found in Raymer's book, in the Navy's Nonresident Training Course book, in Boeing's 707 Workshop Manual, and various other manuals and picture dictionaries [30, 184, 185]. This is significant because the bulk of the aircraft assembly process is spent putting together its structural components. If basically all the structural components are present in the conventional preliminary design phase and the geometric models of the parts and assemblies made from this phase carry all the needed data for assembly sequencing, then assemblies from the conventional preliminary phase are essentially just as adequate as assemblies from the detailed phase in regard to capability and resolution. In other words, the resolution of geometric models of assemblies obtained from the conventional preliminary design phase are sufficient for performing assembly sequencing on an aircraft's major assemblies.

The tradeoff given all this capability is that this design phase lacks the ability to explore the design space as thoroughly as in the conceptual design phase. Due to the major con-

figuration changes, and all configuration changes by the end of this phase, being locked in, other designs are not examined at all. While the required resolution to perform geometric reasoning and assembly sequencing is attained, the desired ability to explore more than just a single set of static geometries is still sacrificed. Different configurations like large wing sweep angles versus low wing sweep angles, large taper ratios versus small ratios, large rib spacing versus small rib spacing, and I-spars versus C-spars all have large effects on the aircraft's performance and can yield noticeable sequencing differences but cannot be explored if this design phase is the one used to provide geometric information. In essence, the conventional preliminary design phase offers almost the same benefits and drawbacks as the detailed design phase; highly detailed geometric representations of parts for the assemblies are provided, but at the cost of only being able to analyze those specific shapes. Though the conventional preliminary design phase is indeed earlier in the design process than detailed design, it does not actually provide a large number of additional benefits. It has the necessary capability required for the desired resolution but is itself too high of a resolution. For this reason, the conventional preliminary design phase as described by Raymer is also inadequate for geometric reasoning for the purpose of integrating aircraft design and assembly sequencing.

4.2.3 Aircraft Early Preliminary Design

An alternative to the conventional preliminary design phase is the early preliminary design phase. It is characterized by its ability to make structural trades and perform structural sizing like in conventional preliminary design while retaining the ability to make configuration changes like in conceptual design. This phase has only relatively recently been implemented due to computing and technique advances in FEA solvers. As described by Collier et al., this phase is made possible by performing FEA using 2D shell elements that have membrane information embedded within them [186]. This allows the semi-monocoque structure of aircraft to be modeled using shell elements while also retaining information on

their attached stiffeners without having to discretely mesh or model them and their shapes. Discretely meshing the stiffeners would normally require additional shell or solid elements, making the entire structural model very large and causing the run time to scale exponentially with the model's size [186]. Embedding the stiffener information in the shell elements makes this relationship linear instead of exponential and thus makes design exploration incorporating structural analysis more tractable. This all enables structural analysis to be performed quickly enough that it is feasible to change structural configurations as well.

Though Collier et al. in their work did not explicitly explore different configurations, Corman et al. and Sarojini et al. did in their works by leveraging the capabilities that come with focusing on early preliminary design. Corman et al. were able to look at many different aircraft types, ranging from conventional tube-and-wing aircraft to hybrid wing body and truss-braced wing aircraft, while still being able to perform relatively detailed structural analysis in sizing the aircraft's wingbox [187]. Similarly, Sarojini et al. were able to do the same while analyzing several different aircraft configurations incorporating different wing sweeps, taper ratios, and aspect ratios [188]. The early preliminary design phase thus is able analyze different aircraft configurations and explore the design space like in the conceptual design phase while also expanding on it by being able to perform relatively high resolution structural analysis.

The early preliminary design phase's geometric resolution is suitable to geometric reasoning and assembly sequencing as well. While the stiffeners and stringers for the aircraft's primary components are not discretely modeled like they are in conventional preliminary design, they are nevertheless still considered by virtue of being embedded within the shell elements representing said primary components. It means that the number of parts present in early preliminary design is effectively the same as in conventional preliminary design. This is important because it prevents situations where a feasible sequence between two parts is made infeasible due to an interference-causing third part not being initially considered. The shapes of the components are similarly preserved within the shell elements, so

that geometric information is available as well. Finally, the embedded stiffener, stringer, and thickness information within the shell elements can all be leveraged to loft the 2D shell structure and make them 3D in an outside CAD program. Making the structures 3D in this manner in a CAD program allows conventional geometric reasoning methods to be able to interact with the sized parts from early preliminary design as though they were from conventional preliminary or detailed design. In this manner, geometric inputs from the early preliminary design phase are able to meet the resolution requirements for geometric reasoning, similar to conventional preliminary design, while also being able to analyze different aircraft configurations and explore the design space, similar to conceptual design. This leads to the following conjecture:

Conjecture 1.1 *Geometric reasoning inputs with similar levels of resolution to geometric data present in the early preliminary aircraft design phase can be used to capably perform geometric reasoning while still enabling rapid design space exploration.*

4.3 Research Question 1.2: Geometric Reasoning Techniques Fidelity

With the resolution of the inputs for geometric reasoning determined, attention must now be directed at carrying out the geometric reasoning itself. Geometric reasoning, to be completely inclusive, actually operates on two different sets of objects. The first is the most obvious, being the parts that make up the assemblies. Sequencing feasibility of these objects is known as geometric feasibility. The second is less obvious and involves the tools that get the parts to where they need to be in order to be assembled or that perform the assembly operations themselves. Sequencing feasibility of these objects is known as mechanical feasibility, which itself is further split into two different aspects, feasibility and accessibility. The feasibility aspect of mechanical feasibility is in regard to: whether there is enough space for the tool to be used at the location it is intended to be used; whether a path exists that allows the tool to reach the location where it is intended to be used; and

whether there is enough space for the tool to fit along that path. The accessibility aspect of mechanical feasibility is in regard to how easily the tool can reach its desired location and can relate to how long the path is, how much the path curves, how much extra space there is along that path, etc. To take full advantage of all the geometric information available and extract as many inferences as possible, geometric reasoning for both geometric and mechanical feasibility must therefore be considered.

The primary goal in performing geometric reasoning is to provide the assembly sequencing analysis with geometric information and determine if a particular sequence is feasible or not. This is what defines the "feasible" aspect of geometric and mechanical feasibility. The secondary goal is to extract information relating to how the parts move or are able to be moved during the assembly process to help determine which sequence is optimal. This secondary goal is often optional, depending on the metrics used to optimize the sequences. In regard to both feasibility analyses, the secondary goal is often carried out while accomplishing the primary one, thus allowing the two goals to be accomplished using the same methods.

The implementation of both the primary and secondary geometric reasoning goals has similar capability requirements to the selection of the ideal design phase and resolution for the geometric reasoning inputs: the results should be such that infeasible sequences are not categorized as feasible and vice-versa and, should the secondary goal be pursued, the results should be such that more optimal sequences are properly identified as better than less optimal sequences. The additional constraint is that the preferred resolution to use was identified to be similar to geometric data obtained from the early preliminary aircraft design phase. This has two implications. First, the calculations involved should be able to be done quickly enough that many different aircraft configurations can be iterated through. And second, due to this all occurring in a relatively early aircraft design phase, detailed and granular manufacturing data is unlikely to be available, meaning techniques and abstractions that rely on the existence of that data are less desirable. Identification of

the geometric reasoning techniques and levels of abstraction that meet all these conditions will allow for the ideal integration of aircraft design and assembly sequencing. There are currently no works in the literature that provide guidance on the appropriate abstraction levels or techniques to use for this, however. This leads into the next research question:

Research Question 1.2 *Given the constraints of early preliminary aircraft design, what abstractions should be used for the geometric and mechanical feasibility analyses so that the design space can be rapidly explored while minimizing the chances of mislabeling the feasibility and quality of the sequences?*

4.3.1 Geometric Reasoning for Geometric Feasibility

There are numerous levels of abstraction for geometric feasibility, which can be summarized as a few overarching levels and many sub-levels to each overarching level. The levels will be covered first. There are roughly two distinct, overarching levels of abstraction for the individual components in an assembly, based on information from Wilson and Ghandi et al [119, 88]. The lower level of abstraction involves the fewest assumptions about the components and can include considerations for how various physical factors and effects such as gravity, friction, deformability, and tolerancing affect the component and its assembly operations. Subsequently, it can also include considerations for how the object is handled and involves factors such as graspability, fixtures, and gripper usage. The addition of the extra factors allows for more realistic results but also complicates the geometric reasoning because calculations for aspects other than part geometry must be considered. As an example, for friction considerations, the compressive force exerted by each component on the assembly and the resultant assembly stability is needed [189]; for tolerancing, the edges of each part are varied by a tolerance factor and the resultant sequences must be feasible and robust with respect to said tolerance factor [190]; and for graspability, the added calculations cause the geometric and mechanical feasibility areas to blend due to consider-

ation of the size of the grasping tool being used and whether it is able to clear the obstacles around the target part as well as manipulate it [125, 191]. Although an even lower abstraction level is possible by combining all these factors together into one work, no work exists that attempts to do so, likely because of the extreme computational expense. As such, this abstraction level has a very wide breadth due to all the miscellaneous non-geometric factors that can be calculated separately from each other.

The other, higher level of abstraction is to just assume that the components are "free-flying geometric objects" during geometric reasoning [119]. This abstraction level thus removes all the previously discussed factors and focuses only on geometric calculations regarding the parts themselves. Despite being less grounded in reality by assuming the other physical factors have no effect, this level of abstraction allows the analysis to be carried out in a much simpler and faster fashion. Additionally, as stated by Wilson and Latombe, it can be done early on in the design process when the geometry of the components and the assemblies are only just being developed [119]. Another benefit is that, without physical factors from the lower level of abstraction such as gravity, friction, deformability, etc., the components can be treated as rigid bodies and assembly sequences can be generated by first determining the disassembly sequence and then inverting it. Disassembly sequences are often easier to work with because they start with everything being assembled already, significantly decreasing the degrees of freedom and ranges of motion the designer has to examine and keep track of [190]. Thus, despite perhaps being less realistic, this level of abstraction has seen far more use in literature due to its ease of use and general applicability.

With respect to geometric feasibility for aircraft major assemblies in the early preliminary design phase, the best overarching level of abstraction to use for design space exploration is the higher level of abstraction where the components are treated as free-flying. This is for a multitude of reasons. The higher level is more appropriate in terms of computational speed for the design space exploration due to its simplicity. The tolerancing aspect from the lower abstraction was explicitly stated to not be considered at the end of

section 2.3 and so is not an important factor. The components in major aircraft assemblies tend to be located and put together via fixtures, causing friction and gravity to not be large issues. That being said, the fixture aspect itself is impractical to tackle due to fixtures tending to be extremely specific to particular geometries and needing to be designed almost from scratch for each, which is too computationally expensive when numerous such geometries are being sifted through in early preliminary aircraft design. On top of this, more information is required to look at fixtures and perform fixture design than is typically available in the early preliminary aircraft design phase, making its inclusion difficult. The same can be said regarding graspability. Graspability analysis requires identification of specific grasping tools, which entails more manufacturing detail than is typically available with the desired design phase's resolution. Regarding deformation, aircraft components are ideally left undeformed during assembly lest they be sent in for rework, making the rigid geometry assumption in the higher level of abstraction attractive. For these reasons, the higher level of abstraction is the most appropriate for early design space exploration.

With the overarching abstraction level selected, the sub-levels and the actual techniques are now reviewed and selected. Because the techniques are essentially synonymous with the abstraction sub-levels, the selection of the method to use is essentially the selection of the preferred sub-level of abstraction.

There are many different methods that can be used to perform geometric reasoning on the components in an assembly. The most notable ones include swept volume, NDBGs, projections, path planning, and Geometric Constraint Analysis. Swept volume methods are essentially collision detection problems where the goal is to translate a part or several parts in specific directions and continuously checking to see if those parts overlap with each other or other parts. Swept volume is generic though can become computationally expensive, with the true shape of the parts being extended out and swept along a specified direction and the presence of interference checked by seeing if there are any other components inside of that swept volume [117]. Many different works use it or its basic concept of translating

parts a specific distance in order to check for local interference.

Similarly, many works use the concept of bounding boxes to represent their geometries in the geometric reasoning, so they will be covered here for reference as well. Bounding boxes simplify collision detection of parts by simplifying their representation and encasing them within bounding boxes. There are several types of bounding boxes, with a few being Axis-Aligned Bounding Boxes (AABB), Oriented Bounding Boxes (OBB), and Discrete Oriented Polytopes (DOP) [192]. AABBs have the bounding boxes being aligned with the principal axes and OBBs have the bounding boxes oriented arbitrarily. DOPs do not actually use bounding boxes but rather many-sided polygons or many-faced polyhedrons to approximate the shape of the part and are essentially the intermediate resolution between bounding boxes and the part's actual shape. Bounding boxes are often utilized with hierarchies where different parts are clustered together based on how close they are to each other. If a node representing a collection of parts on the hierarchy does not intersect or interfere with another node on the same level, then the parts under the first node will not interfere with parts under the second node. This prevents a large amount of unnecessary interference checking, making bounding box hierarchy a computationally efficient concept to take advantage of [193].

NDBGs are a special type of swept volume method in that the translation directions are not arbitrary and are instead the normal vectors of the components' mated faces. Additionally, the parts being translated, based on whether or not they interfere in the translation direction, can actually be subassemblies instead of just individual parts. Further details on NDBGs can be found in section 3.1. They can be computationally efficient methods for geometric reasoning, but only if the sequences are binary and monotone. Binary refers to there being no more than two subassemblies moving with respect to each other at any moment in time and monotone refers to the subassemblies not being able to be modified once assembled [120]. These conditions apply to the majority of assemblies and sequences considered.

Geometric Constraint Analysis (GCA) is similar to NDBGs. It utilizes a diagram showing all the constrained directions, or directions where parts cannot be disassembled or assembled, in a manner akin to the NDBGs showing all the feasible directions. GCA embeds sequencing information as well in that if all directions are constraint directions, then the assembly or subassembly currently being examined is fully constrained, akin to how NDBGs have smaller directional blocking graphs for each feasible direction. The only major difference between the two is that GCA requires user input or augmentation by some other method, like bounding boxes, in order to determine the constraint directions, making it somewhat inefficient [192].

Projection methods are perhaps the simplest of all the geometric reasoning methods. As described by Pan et al., it consists of taking a group of parts and projecting them along the principal axis directions onto a plane some distance away and seeing if any of the parts overlap. Overlapping parts are parts that interfere with each other if assembled or disassembled along that direction [127]. In this manner it is able to take 3D parts and geometries and collapse them into 2D for more efficient calculation. The method described by Pan et al. is limited to the principal axis directions, though it is still possible for the designer to specify a customized projection direction in advance. Most assemblies are more efficiently put together in the principal directions however [16]. Because the majority of geometric reasoning methods collapse down to a projection operation in some way, shape, or form, projections by themselves, combined with the two previously stated facts, are one of the most computationally efficient methods. They do have limitations in that they require sequential assemblies where only one part or subassembly is being removed or inserted at a time. Additionally, they can fare poorly with assemblies whose parts require more than one or two translation steps to be assembled or disassembled. This is because the projection plane tends to be placed relatively far away from the assembly so as to not accidentally bisect an existing part, which results in parts needing multiple steps to be assembled or disassembled seeming to not be able to do that at all due to all their projections indicating

interference.

Whereas projection methods are perhaps the simplest method, path planning methods are perhaps the most complex. According to Ghandi and Masehian's survey paper, the path planning problem consists of, given an assembly and its parts and geometries, finding a sequence of positions for each part, known as paths, such that there are no collisions and that the parts are able to finish their paths at their assembled location [88]. It is primarily reserved for complexly-shaped assemblies where parts have to essentially traverse a maze in order to reach their final assembled location. Other methods, such as projections and bounding box methods, are often used as intermediate steps while a separate algorithm determines which paths should be explored such as in Hu et al.'s work [122]. Because of this, these methods are quite complex and computationally expensive despite providing capabilities that no other class of methods can provide. This causes them to oftentimes be limited to assemblies with relatively few parts such as in Christiand and Yoon [194] and Masehian and Ghandi's [195] works. As a result, the body of literature of ASP works that include path planning for geometric reasoning only makes up a small part of the overall ASP literature, as noted by Masehian and Ghandi [195].

With some of the major methods covered, the characteristics of the aircraft assemblies themselves will now be discussed to match them with the appropriate method. The tail surface assemblies are very similar to the wing ones, with the only difference being they are slightly smaller in size. Tail surface assemblies will thus just be considered small wing assemblies. The wing and fuselage assemblies are, at the early preliminary design resolution, relatively simple. They are both effectively monotone, meaning parts do not need intermediate placement before being fully assembled. Parts that do need intermediate placement, such as stringers on wingskins or stiffeners on spars, only have one practical sequence to consider to generate their respective subassemblies. As such, these subassemblies are generally treated as one part each and must be formed first before they can interact with any other parts, causing the overall assembly to be monotone in nature.

The assemblies are also sequential because none of the components need to be moved at the same time as any other component to be feasibly assembled or disassembled. All of the subassemblies can be broken down into their own sequences where only one part is moved at a time. Multiples of each type of rib, stringer, or stiffener are treated as only a single part each. This means, at the base level, all feasible possible sequences can be presented as only moving one part at a time, making the assemblies sequential.

The two types of assemblies can, however, be non-linear due to parts having the ability to be fabricated integrally or be a part of a subassembly. Inserting or removing these thus technically counts as operating on more than one part, hence the non-linearity. Despite this, in practical applications said integrally fabricated components and subassemblies are fabricated or assembled before any other assembly activities are carried out. By the time of the usual assembly steps, these behave as a single part. On top of this, no parts are required to be part of a subassembly for their assembly or disassembly to be geometrically feasible, which is the main worry of whether an assembly is linear or not. As such, the discussed geometric feasibility methods are all still applicable and can be used on the two assemblies in question.

Because the assemblies covered are so simple, meaning overly complex scenarios need not be considered, all the methods are essentially equally capable in minimizing the odds of incorrectly evaluating sequencing quality and mislabeling feasible sequences as infeasible. This means only the design space exploration requirement needs to be met. Thus, the best method to use should only be dependent on simplicity and computational speed. For this reason, the projection method is the best candidate method to use. It is simple mathematically and in terms of implementation and the majority of the other methods eventually resort to projection as their last step anyways. One of the few issues is in regard to which directions the projections will be performed in. Based on historical trends, almost all of the components for the wing and wingbox assemblies are assembled along the principal axis directions and so are not a problem. For the fuselage assemblies the circular cross-sections

prevent projections in the principal axis directions from yielding useful results, which can be problematic. This can, however, be alleviated by replacing the $\pm$x and $\pm$y directions with a radial direction, akin to replacing a Cartesian coordinate system in a problem with polar coordinates. This is demonstrated in Figure 4.1, which depicts a 2D cross section of a generic fuselage. Here, the fuselage's stringers and longerons are free to move in the radial direction once either the fuselage skin or frame is removed. The projection location simply has to be anywhere outside of the fuselage or at locations A and B inside the fuselage. At locations A and B inside the fuselage, the stringers and longerons can be removed/inserted in the $\pm$z directions afterwards or in any other direction if the skin and frames are both not made as one component and assembled/disassembled already. Effectively, the stringers can be disassembled by getting to points A and B or assembled from those locations. This essentially renders all the aircraft's major assemblies monotone in nature, addressing the main weakness of projection-based methods and their inefficiency when dealing with non-monotone assemblies.

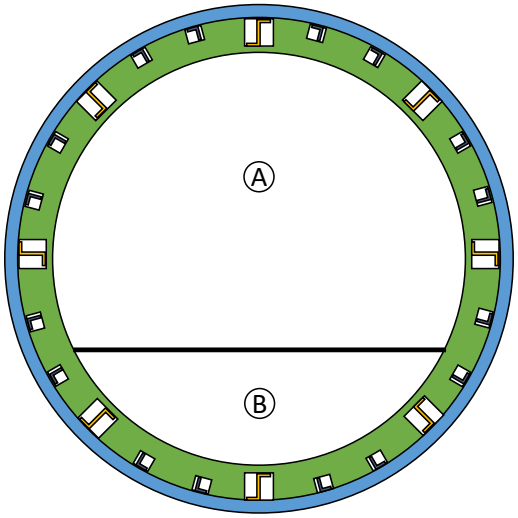


Figure 4.1: 2D Cross Section of Generic Fuselage Assembly with Projection Locations Inscribed in Circles, and Colors Indicating Structure Type: Blue = Fuselage Skin, Gold = Longeron, Green = Fuselage Frame, Gray = Stringers

Projection methods are thus the desired abstraction sub-level underneath the overar-

ching free-flying object abstraction level. They will be the primary method to perform geometric reasoning with for the assemblies, specifically geometric feasibility. Concurrent with this, they will be used to generate interference matrices, as the two tend to go hand in hand [124].

4.3.2 Geometric Reasoning for Mechanical Feasibility

With the geometric reasoning for the assemblies' actual parts in the form of geometric feasibility covered, attention is now directed towards geometric reasoning for the tools that allow the parts to be assembled in the first place in the form of mechanical feasibility. This particular topic, as noted by Bahubalendruni et al., is often not covered, whether because other works simplify assemblies enough to where they do not need tools, do not have sufficient information to analyze the tools as well as the components, or simply believe it is too computationally expensive [196]. Regardless, geometric reasoning for the tools is vitally important because if the tools to perform the assembly operations cannot reach the correct operation location, then components cannot be put together and properly assembled even if they do not interfere with one another on their way to their final position. Additionally, different assembly tool types can be analogous to different types of assembly equipment, and for aircraft manufacturing the equipment has a significant impact on the final cost and throughput of the entire design. It is thus important that tools and equipment be checked for feasibility and accessibility and geometries that make tool-access infeasible be altered or tools that cannot access the assembly area swapped out before large amounts of cost are invested in the wrong equipment.

Mechanical feasibility analysis has two differences as compared to geometric feasibility analysis. The first difference is that it is composed of two aspects, a feasibility one and an accessibility one. The feasibility aspect is concerned with whether the desired tool to be used is able to get to and be used at the assembly operation location. If the tool is unable to get to the operation location, whether due to lack of space or a feasible path, or

unable to be used at the operation location, often due to lack of space, then that tool and possibly that sequence is infeasible. The accessibility aspect is concerned with the quality of a mechanically feasible tool and sequence. The quality can be denoted by a number of characteristics, such as how well the tool fits or how much extra space is available or how long it takes the tool to get to where it needs to, with the common factor being that they are all related to the ease of using the tool, hence the term assemblability. To have a holistic coverage of mechanical feasibility, both aspects must be considered, though the feasibility aspect is the more critical of the two since ease of use is secondary to whether a tool or sequence is usable at all.

The other difference is that where geometric feasibility analysis has overarching levels of abstraction and sub-levels in each level, with the techniques located in the sub-levels, mechanical feasibility analysis has no sub-levels. The different levels directly define the methods available. The levels of abstraction can range from just checking if the tool can fit into the allotted space between parts and assuming it will automatically be able to get there to detailed path planning problems that see if there is a viable path for the tool to arrive at the designated location. On top of this, mechanical feasibility analysis must be performed in between every assembly operation. This is because the geometry of the assembly changes in between every operation, so it must be continuously rechecked. It means that if there are n parts in an assembly, $n-1$ mechanical feasibility analyses must be performed for each sequence. Since there are possibly $n!$ different sequences to evaluate, more computationally expensive techniques with lower levels of abstraction quickly become intractable. Despite this, methods too high in abstraction lose critical information such as whether the tools can get to and fit in the desired location as well as how easily they can get there. If it is difficult to get the tool to where it needs to be then, while the assembly sequence is still feasible, it will take longer and cost more than normal. The problem is thus to select a method that balances the need for computational speed with the need to obtain as much accessibility information as possible.

To be able to select the correct method and level of abstraction, the abstraction levels themselves must first be identified and discussed. Though it is not explicitly mentioned, in the mechanical feasibility literature there are approximately three different levels of abstraction. The higher levels tend to be less computationally expensive but also generally do not provide the full picture required to concretely determine if a tool can access the required area. The lower levels use fewer assumptions and are more realistic but also increase in computational time and complexity to the point where they themselves become a difficult problem to solve on top of the sequencing problem.

The highest level of abstraction is in regard to visibility. In the context of assembly sequencing, visibility refers to the ability of an assembly worker to see the joining area. Its importance is primarily related to ergonomics because a joining area more visible to the assembly operator is generally also easier to access, reducing labor time and cost [191]. The computation of visibility is straightforward, though different authors use different metrics. Ding et al. used the ratio of the projected visible area of a part to the projected visible area of the subassembly or other parts it is assembled to [191]. Frank et al. and Hoefer and Frank used the visibility cone. To calculate this visibility cone, both sets of authors take cross sectional slices of their parts/subassemblies and then determine the unobstructed range of angles from the features of interest to the tool/observer using the 2D cross sections [197, 198]. The computational speed for both these metrics is similarly relatively quick, with Frank et al. able to make over a thousand calculations per second in his work [197]. Despite these speed benefits, the visibility metric by itself does not actually offer an objective measure of mechanical feasibility; it simply just indicates how visible the joining area is, which has an effect on a tool's accessibility but is not its sole determinant. The only exception to this is in the case of the fabrication work covered by both Frank et al. and Hoefer and Frank, where the milling and machining work they discuss requires a direct line of sight in order to operate [197, 198]. The majority of aircraft components are no longer machined in such a way, however, so the visibility metric is only useful as an add-on to

other methods involving accessibility.

The next level of abstraction focuses on whether there is enough space around the joining area to be able to use the tool. Methods at this level of abstraction check to see if there is enough space to both place the tool in the correct location as well as for it to operate. This is because many tools, such as wrenches, require room in order to turn and operate on the nuts and bolts used to assemble parts together. Even if there is room to place a wrench at the joining site, it is pointless if there is no room to turn said wrench. In the assembly literature, there are two major classes of methods that address this problem at this level of abstraction, ones that compute and use the configuration space (C-space) and ones that do not. To define configuration space, a configuration must first be defined. A configuration is a collection of parameters used to describe the positioning of an object. A configuration space is the collection of all possible said configurations. It is used because it is significantly less expensive to describe an object's configuration in a path planning problem than to describe every single obstacle, detail, and possible path in its surrounding environment.

Wilson has the most well known C-space implementation in regard to assembly tool usage area [199]. In his work he combines the physical size of the tool along with the required space for it to be used and calls this the tool use volume. The available space in between all the components at the joining before, during, and after the assembly operation is continuously checked to see if the tool use volume is able to fit. The multiple available configurations of the tool use volume during all this is what defines the C-space, which is determined by subtracting the area occupied by obstacles from the available space [199]. The method used by Wilson is a more simplistic application of the Polygon Containment problem, which looks at translating a polygon until it is completely contained by another polygon [200]. Wilson's application only examines simpler shapes, reducing the complexity and computational expense of the problem he needs to solve. Though Wilson's method is able to map out the entire area surrounding the components for available space by calculating the C-space, the process itself, as defined by Wilson, must be redone for every tool

and assembly configuration, which makes it computationally expensive regardless [201].

Chung and Peng were one of the first to use the concept of Global Accessibility Cones (GACs) for assembly sequencing [201]. GACs were created in order to calculate tool usage volumes and place them in an assembly without interference in a manner similar to Wilson but without computing the C-space, which is an expensive process. This is done by first calculating a GAC, which is essentially a 3D sphere centered on the joining site and indicates which directions contain obstacles and which do not. Typical GACs are made by projecting unit vectors on shapes within a unit sphere, but Chung and Peng used a more efficient method involving projecting a triangle patch instead. To check the accessibility of the tools, they essentially take the tool and sweep it through the projected 2D areas needed to operate them and determine these swept angles. The swept angles are compared with the GAC and the tool usage deemed feasible if it does not intersect the sections of the GAC that contain obstacles. Although this method of determining feasible placement of a tool use volume is less expensive than calculating a C-space, the fact that it requires sweeping the tool through its range of motion drastically limits the types of tools that can be analyzed. In particular, this method is limited to fasteners and hand tools that require motion such as a wrench and is less applicable for tools like rivet guns or robot arms.

The lowest abstraction methods, similar to geometric feasibility, are path planning ones. Whereas the previous level of abstraction is focused on the tool being usable at the joining site, this level of abstraction is focused on getting the tool to that joining site. Assuming the tool is just a point object, the same extensive literature on these methods, as reviewed by Ghandi and Masehian, is thus also generally applicable to for determining mechanical feasibility [88]. However, since the previous abstraction already considers tool use volumes, path planning methods for mechanical feasibility should ideally include the physical volume the tool occupies as well. Seen in this light, only a few works in the literature discuss the path planning of tools or components while also considering their physical size. One of the few is from Hui et al. who, after determining the assembly sequence, essentially

simulated each step in each sequence by sweeping their components through 3D space and planned a different path for each component if it runs into obstacles. If the paths are poor in quality, such as requiring too many turns to avoid too many obstacles, the sequence is reevaluated and altered. Hui et al.'s work is more oriented towards components than tools, but their method is flexible enough to do both [202]. Another work is from Dakdouk and Xi, who combined the tool use volume and path planning techniques in a more piece-wise manner. The physical space and obstacles the tool has to traverse around are represented via a global accessibility area for quick 2D checks and a global accessibility volume for 3D interference checks. The tool use volume aspect is checked using the global accessibility areas and volumes, while the path planning is done via only the global accessibility volume. Dakdouk and Xi's method is thus able to more robustly find paths for the tools to traverse by performing the volume and path checks separately. Their work is focused on modeling robotic arms but can conceivably be expanded to other tools as well [203]. As is expected of such low abstraction methods, the computational expense is very high and only a single set of geometries are analyzed at a time.

The best method and abstraction to use is determined by examining what mechanical feasibility analysis provides, how that translates to the methods just reviewed, and whether the resulting observations align with the requirements for the framework being developed. Mechanical feasibility analysis ultimately revolves around two types of analysis, one related to checking if there is enough space to place the tool in a desired position and one to check if there is a path that allows the tool to move in a desired way. Tool use volumes and GACs pertain to the first, since they check if there is available space to use the tool at locations where joining operations occur. Meanwhile, visibility and path planning are related to the second, since visibility checks if a tool is able to move to the desired location in a straight line while path planning allows for a more varied route. The requirements for the framework being developed are that the analysis be computationally inexpensive enough to be rapidly iterated through and used during the aircraft early preliminary design process,

that detailed manufacturing information is not needed since it is generally not provided in said design phase, and that the sequencing feasibility and quality are evaluated properly. "Properly" in this case means that feasible and infeasible sequences are less likely to be mislabeled and that the possibility of objectively better sequences being mislabeled as worse than inferior sequences and vice versa is minimized. This must be done for both the feasibility and accessibility aspects of mechanical feasibility. As the feasibility aspect is more critical and not optional, it will be covered first.

The requirement for the analysis to be computationally inexpensive and not need detailed manufacturing data rules out the tool placement-type analysis and so the GAC and tool use volume abstractions for the feasibility aspect of mechanical feasibility. In the framework being developed, many manufacturing processes must be analyzed, implying many sets of equipment and a myriad of tools available for each set. This causes those stated abstractions to fail to meet the requirements in several ways. First, the tools are all varied and not necessarily limited to wrenches and screwdrivers that can leverage GACs. This rules out GACs, with the wide variety of tools and tool shapes effectively forcing tool use volumes to be used instead. Those involve calculating the expensive C-spaces and possibly more complex versions of the Polygon Containment problem as well, which become computationally intractable in early preliminary aircraft design [204, 205]. Second, trying to analyze all the different tools for every possible equipment type would result in a combinatorial explosion anyways even if GACs were somehow able to be used or C-spaces were made less computationally intensive. To try to get around this by analyzing only a select few tools instead is to default several important manufacturing variables, such as the manufacturing process and part size, that should be left un-defaulted to produce more optimal results. Defaulting variables in this manner when trying to explore the design space is undesirable. And third, even if both of the previous issues were resolved, they still need data on all the tools such as their size, type, ranges of applicability, and degrees of freedom to be available. During early preliminary aircraft design that data typically is not present

or obtainable, making inclusion of this type of analysis impractical or impossible. It is thus desired to remain as tool agnostic as possible during the early design phase.

Tool use volumes are still an important component of the feasibility aspect of mechanical feasibility, since a tool not fitting into its desired space renders that tool or sequence unusable. However, the strict constraints imposed by operating in the early preliminary aircraft design phase necessitate a simplifying assumption be made. The history of commercial aircraft manufacturing proves that there are generally a large number of tool sizes available for every type of equipment used, meaning if a particular location is too small another tool will simply be used. From this, the assumption is made that the proper sized tool will always exist for each sequence given they remain within the overarching design scope of involving an aircraft major assembly such as a wingbox or fuselage. A typical aircraft's size only further emphasizes this as even comparatively small stringers are still large enough to be assembled to skin panels or spars via drilling and riveting. Therefore, the assumption of the correctly-sized tool always existing is used to address the fact that tool usage volume is an important part of mechanical feasibility analysis because the only available abstractions are too computationally intensive to properly consider and require too much detailed data. This does not violate the requirement that feasible sequences not be mislabeled and vice versa because, as noted by Dakdouk and Xi, a possible option when tool use volumes are small is to just develop a new tool to accommodate them [203]. The geometric structures present in the early preliminary aircraft design phase are common, standard, and tested and so are very unlikely to be complex enough to warrant a redesign due to tool use volumes alone. This means that, at worst, the feasibility issue in regard to tool use volumes turns into a cost problem of how much it would take to develop a tool able to access the desired part, minimizing the chances of mislabeling a mechanically feasible or infeasible sequence.

The majority of path-related analyses, path planning in particular, are also ruled out due to all of them requiring even more detailed C-space mappings and calculations than with

the tool use volumes, making them too computationally expensive for early design. The tool must still be able to reach its assembly operation location, however, for the sequence and tool selection to be feasible. Given this dilemma, a compromise is to forego the detailed path planning, where the optimal path is found, and only check for path existence to ensure that the tool is able to reach its destination in the first place using any path possible. In this way, only a single path planning calculation needs to be done for each configuration to identify a single working path as opposed to the myriad number involved in identifying the optimal one.

The main determinant of the feasibility aspect of mechanical feasibility during the early design phase, if the issue of the tool being able to fit in the available space and determining a specific path is not pertinent, is whether there is a closed contour formed by the assembled parts around the joining site that blocks it off from any locations where a tool can physically originate from, before the joining operation can take place. A 2D cross section of a generic wingbox assembly and its structures is presented in Figure 4.2 to help demonstrate this.

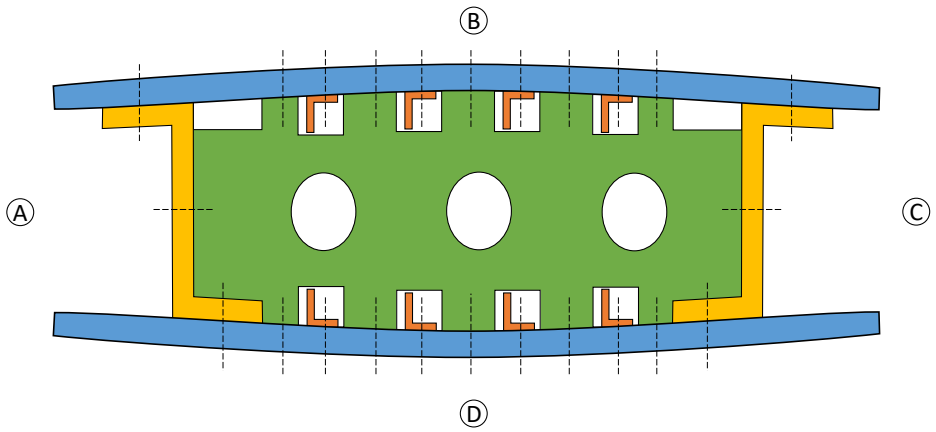


Figure 4.2: 2D Cross Section of Generic Wingbox Assembly with Dashed Lines Indicating Joining Locations, Accessibility Locations Inscribed in Circles, and Colors Indicating Structure Type: Blue = Wing Skin, Gold = Spar, Green = Rib, Orange = Stringers

Both sides of the joining locations indicated by the dashed lines must be accessible in order for the assembly operation at that particular joint to be carried out. This is because almost all current regulation-acceptable operations involving drilling and riveting, which are

often used even in co-cured structures due to said regulations, require access to both sides of the structure. Tool feasibility is thus determined by whether both sides can be accessed given a particular assembly sequence. In this scenario, accessibility refers to the ability to form a path from the joining site to any area outside of the wingbox contour because, for this wingbox, all of the accessibility locations are outside its contour. To check this, therefore, normally requires a calculation to see if the shapes surrounding the joining site form a larger, closed off shape and includes sequentially checking the intersections of each pair of neighboring parts. However, given the overarching design scope of conventional commercial aircraft, the fact that major assemblies for such aircraft primarily involve fuselages and wings and wingboxes, and the fact that such fuselages and wings and wingboxes are basically all built in similar styles, the closed-contour calculations can be drastically simplified once the accessibility locations are defined.

Instead of sequentially checking intersections between parts, the same results can be obtained by tracing a ray from the joining sites to the letters inscribed in circles in Figure 4.2, which will from now on be called accessibility locations. If a ray can be successfully traced from a joining site to any of the accessibility locations, then a tool can reach that site unimpeded, indicating tool feasibility. This works because of the fact that the directions in which an aircraft part is assembled for the prescribed major assemblies are nigh always a principal Cartesian direction or a straight radial direction. It is the same reason why projections are applicable for the geometric feasibility analysis. Given this, when the aircraft major assemblies are being assembled, if their contour is open, it will always be open in a principal direction relative to the major assembly, indicated by the accessibility locations in Figure 4.2. The tools essentially originate from these accessibility locations, meaning if an unimpeded ray can be established, the feasibility aspect of mechanical feasibility is ensured since verifying the existence of a tool path is enough for early design. The specifics of the actual path are not needed. For a fuselage assembly, the accessibility locations take different positions since fuselage assemblies are assembled more in the radial direction

than in principal Cartesian ones and because their interior is typically much larger. The accessibility locations are therefore located more similarly to where the inscribed letters are in Figure 4.1, representing the locus of where tools are used by workers working from the inside of the fuselage. The accessibility locations can also be located anywhere outside of the fuselage, representing the locus of where tools are used by workers working from outside the fuselage. The method just described almost exactly matches the description of the visibility abstraction level, making it the level that best suits the feasibility aspect of mechanical feasibility for aircraft early in design. This works out in terms of computational speed as well since the visibility level is the only remaining one not ruled out by the computational speed requirement.

There are two caveats to this, however, for the fuselage and wingbox assemblies. The first caveat is that there are no known often-used stringer shapes that, on their own, form a closed contour around their own joining locations. This means that stringers will never block off access to their own joining locations. Since they themselves are never actually the structures that seal off the wingbox or fuselage contours, as long as a line of sight can be made from anywhere on a stringer to an accessibility location, then the stringer's joining site is accessible.

The second caveat is, if Figure 4.2 is observed carefully, it becomes apparent that there are no sequences where both sides of the joining sites for all the parts always have line of sight with the accessibility locations. This does not mean that there are no feasible sequences because aircraft are still obviously being built despite this. It does mean, however, that until one-sided drilling and riveting techniques are developed enough to pass through regulations, maintenance access holes will always be required whenever line of sight with an accessibility location cannot be established. This does not make all sequences equal as those with fewer parts needing access holes to be assembled will still be superior; fewer parts needing access holes means that fewer operations are done via an operator working from said access hole, leading to less time and cost spent on assembly.

This directly leads into the accessibility aspect of the mechanical feasibility access, now that the feasibility aspect has essentially been entirely covered. Analysis involving the accessibility aspect is typically even more involved than the feasibility one since it not only checks whether there is enough space for the tool to move where it needs to, it additionally checks how much extra space there is around the tool's use volume and is much more reliant on the path length. The accessibility aspect's analysis essentially requires a complex integration of the Polygon Containment problem and the path planning one due to needing to evaluate how easily the tool can get to where it needs to. And to add to this, Wilson states that the most detailed of this analysis is liable to cause assembly plans to be over-constrained without additional calculation steps, causing it to be even more undesirable [206]. Due to the elimination of the majority of the tool use volume and path planning analyses for computational expense reasons, the conventional way of carrying out the accessibility aspect of the mechanical feasibility analysis is unable to be incorporated.

However, the second caveat of using visibility-based methods for the feasibility aspect provides an alternative path to carrying out the accessibility aspect and determining the quality of a mechanically feasible sequence. The existence of access holes and the assumption of a correctly-sized tool always being available essentially makes the feasibility aspect not a matter of whether a sequence is actually mechanically feasible, but a matter of how much that feasibility will cost. Without one-sided drilling and riveting, access holes are always required for a wingbox and so all sequences are always mechanically feasible, but some will have more locations that require those holes. Taking inspiration from Dakdouk and Xi's own accessibility percentage metric based on accessibility volumes, a ratio or number can be made indicating how many joining locations need usage of access holes for feasible tool usage, which can act as a measure of accessibility [203]. Said accessibility metric denotes a sequence's quality because sequences with more joining locations that need access holes will require workers to spend longer periods of time in those access holes, making the assembly and sequence less accessible and take more time and cost

more in terms of labor. It is also simple and quick to obtain, being a part of the visibility calculations for the feasibility aspect, making it appropriate for the early preliminary aircraft design phase. Usage of such a metric turns the question of feasibility into a matter of a sequence's accessibility, cost, and quality. Implementing a metric like that therefore fulfills the optional capability requirement of being able to properly indicate the mechanical quality of sequences on top of the mandatory one regarding minimizing the chances of mislabeling feasible sequences. For all these reasons, an accessibility metric based on how many joining locations require access holes to make the sequence feasible will be used to judge the mechanical feasibility quality of sequences for wingboxes.

Though most fuselages do not need access holes because their interiors are meant to encapsulate people and so are more spacious, the same concept as wingboxes can be used where tools originating from certain locations incur accessibility penalties. For fuselages, this is dependent on whether the accessibility locations are sufficiently enclosed in the fuselage's contour. Accessibility locations in a sufficiently open contour indicate that large equipment and tooling is easily placed at both the interior and exterior locations, allowing for greater accessibility. Conversely, even if the contour is still open, it can be closed enough that large equipment and tooling can no longer be easily placed in the interior accessibility locations, requiring more specialized equipment and tooling able to operate in the confines of the more closed contour and so impairing accessibility. This results in a similar metric to the wingbox where the accessibility is based on a ratio or number involving how many joining locations require an interior accessibility location after it is sufficiently closed off to start incurring penalties for its use. As the "closed enough" state is more dependent on the available equipment and tooling and fuselage size and even the manufacturing process used, since all those impact what can fit into the fuselage's interior, the fuselage accessibility metric is more difficult to calculate than the wingbox one. Regardless, it still meets the stated requirements in the same ways the wingbox one does. As such, a metric based on how many joining locations require penalty-incurring interior

accessibility locations will be used to judge the mechanical feasibility quality of sequences for fuselages.

Abstraction levels and techniques that are least likely to mislabel a sequence's feasibility and quality have thus been determined for both the geometric and mechanical feasibility aspects of geometric reasoning. These are deemed appropriate for the early preliminary aircraft design phase as they do not require detailed manufacturing information and can be quickly executed to not bog down the design space exploration. This leads to the following conjecture:

Conjecture 1.2 *The levels of abstraction represented by using projection techniques on parts depicted as free-flying objects for geometric feasibility calculations and visibility-based techniques for mechanical feasibility calculations allow for rapid and capable geometric reasoning in the early preliminary aircraft design context.*

No guidance was provided on the proper fidelity of geometric data and reasoning techniques to use to combine aircraft design with ASP and so infuse aircraft design systems-thinking methods with ASP capabilities. The term fidelity was then better defined and the appropriate levels of scope, abstraction, and resolution for the data and techniques were established by searching through the literature for design phases and methods that met the requirements laid out. There is now a precedent with which to perform aircraft design and ASP together, which can additionally be further improved on in the future if desired. This concludes the augmenting of aircraft design systems-thinking methods with geometry factors by including assembly sequencing analysis and is summarized in Figure 4.3.

Demonstrations of Conjectures 1.1 and 1.2 will be performed as part of the experiment in the next chapter as well as the case study in Chapter 7. This is because standalone demonstrations of either conjecture would not actually show very much beyond the obvious of being able to implement them; no further implications or inferences would be able to be drawn from such demos by themselves, especially not how any of the geometric informa-

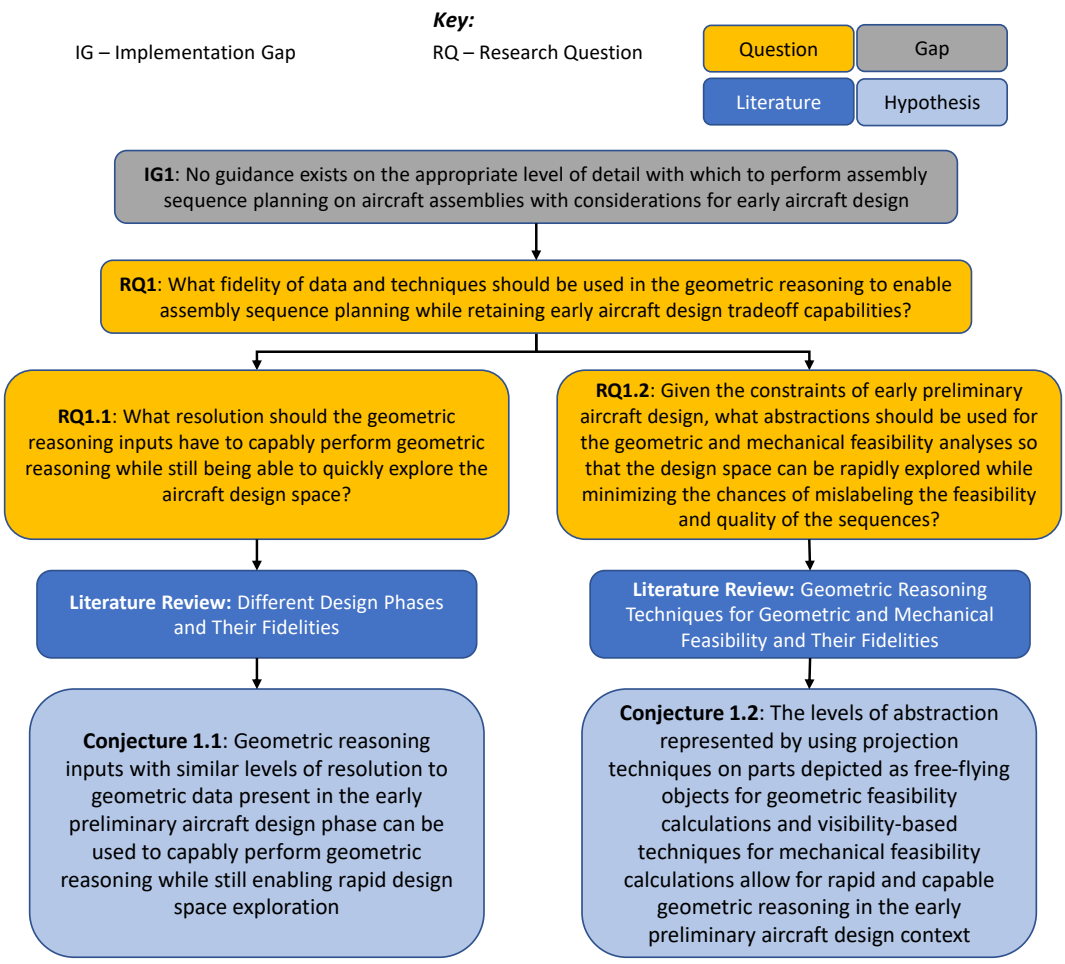


Figure 4.3: Summary Logic Diagram for Research Question 1

tion would affect cost or throughput. The demonstration in the next chapter's experiment will show the effects of using these conjectures on data with early preliminary aircraft design fidelity. The demonstration in the case study will show the effects of these conjectures when early preliminary aircraft design is carried out in full and numerous geometries and configurations looped through. Essentially, the experiment in the next chapter showcases how the assembly analysis should be performed such that it is able to incorporate early preliminary aircraft design, while the case study shows what happens when early preliminary aircraft design actually is incorporated and carried out with the assembly analysis.

The following chapter will thus discuss augmentations of the assembly analysis portions of aircraft design systems-thinking methods via including the material factors and incorporating ALB.

CHAPTER 5
INTEGRATING AIRCRAFT DESIGN CONSIDERATIONS WITH ASSEMBLY LINE BALANCING AND SUBSEQUENT IMPLEMENTATION

5.1 Problem Characteristics and Research Question 2: Integrated Assembly Sequence Planning and Line Balancing Problem

As has been mentioned several times, the goal of this work is to obtain improved throughput and cost by better integrating the aircraft design and assembly disciplines. It has been determined that this can be done by using geometry and material-related information as the common link between the aircraft and assembly analyses to create stronger feedback loops between them. Aircraft geometry and materials are already regularly traded and changed in in the aircraft design discipline. The main challenge has been linking those tradeoffs with the equivalent tradeoffs in the assembly discipline. The integration of aircraft design and assembly via addressing geometric factors was tackled in the previous chapter through ASP's geometric reasoning process. This chapter looks at doing the same by addressing material factors through incorporating ALB. Therefore, while the previous chapter was more focused on the aircraft design side of aircraft design systems-thinking methods, looking at aircraft design phase fidelities as well as the appropriate geometric reasoning fidelities to match them, this chapter is focused more on the assembly analysis side, looking at how to appropriately carry out the assembly analysis and obtain the desired tradeoffs based on the materials selected in aircraft design. This means incorporating material considerations regarding integral fabrication, spatial constraints, being able to look at multiple manufacturing processes, and being able to optimize for both throughput and cost. Integral fabrication is an important aspect and benefit of new and novel materials; comparing multiple manufacturing processes allows for the best one to be identified and is

also associated with comparing different material systems, since different materials almost always require different processes; spatial constraints allow for more realistic throughput and cost estimates when looking at different manufacturing processes and materials and can accentuate their benefits and drawbacks; and not prematurely locking in the production rate and cost prevents the design space from being hemmed in too early on. These are all important material factors that can improve production rate and cost on the assembly side and can link up with the material trades on the aircraft design side, better connecting the two.

The ALB literature has problem types that are related to all of these factors and so can be used to try to implement them. The assembly subgraphs problem exists for integral fabrication, the equipment selection problem with zoning constraints can be used for variable manufacturing processes, the TSALBP is available for spatial constraints, and the station paralleling, multi-objective, and SALBP-E problems are present for variable throughput and cost. These must additionally be combined with ASP both because integral fabrication can also be represented by ASP, because ASP needs the resource considerations of ALB to provide a throughput estimate, and because ASP creates the precedence constraints that ALB needs, intimately linking the two. There are integrated ASP and ALB problem types in the ALB literature that can handle this too.

This all combines into a complex integrated ASP and ALB problem. To slightly reduce the complexity, two simplifying assumptions are made in this initial posing of the problem. The first is that, despite the emphasis on variable manufacturing processes, only one process will be analyzed at a time. That is to say, a single line balance will only model a single manufacturing process and, because materials tend to be dependent on the process used, a single material. Manufacturing processes and materials will be evaluated by comparing their respective line balances. The second simplifying assumption is that there are no equipment alternatives, i.e., there is only one equipment option for each type of equipment. This is because allowing equipment alternatives for each equipment type or allowing the

mixing of materials and manufacturing processes for each assembly dramatically increases the complexity and size of the design space while not providing large-enough benefits. Allowing for the former enables comparisons of different equipment manufacturers when what is desired is comparison of different materials and their processes. Comparing equipment manufacturers does not have an analogue on the aircraft design side and so does not aid in integrating the aircraft design discipline with the assembly one. And allowing for the mixing of materials and processes for each part only provides similar insight to comparing manufacturing processes as a whole, with the cost of being significantly more combinatorially expensive. These two assumptions are thus made due to the high cost and low benefits of doing otherwise. These assumptions can be rolled back in future additions that seek to improve on this work.

Based on those assumptions, the resulting integrated ASP and ALB problem has the following characteristics:

- Only one homogeneous product is being produced, but it is composed of many smaller parts. The size and shape of the product and its parts is dependent on the design selected, which cannot be known in advance.
- Tasks must be processed according to precedence constraints.
- Precedence constraints are dynamic and can change depending on which parts are integrally fabricated, which sequences are being used, and what design is being evaluated. It cannot be known in advance as a result of this.
- Tasks cannot be divided among more than one station.
- Task times can be deterministic but are not static, changing depending on the assembly sequence being used, the parts being integrally fabricated, and the geometry and configuration of the design selected. It cannot be known in advance as a result of this.

- The assembly line is a paced line, meaning all station times must be lower than the cycle time. Subsequently, the production rate is the inverse of the cycle time.
- A station's station time can be larger than the cycle time only if enough parallel instances or duplications of that station exist such that its effective station time is lower than the cycle time.
- Each instance of a duplicated station has the same resources as the original, but these resources are not shared, e.g., if there are two duplications of a workstation that has Equipment A, Tooling B, and number of Workers C, then the factory has, including the original station, three total sets of that station with three sets of Equipment A, Tooling B, and number of workers C.
- The sum of the space requirements of all stations and their parallel instances cannot be larger than what is available in the factory.
- Stations can only process one task at a time.
- More than one manufacturing process will be compared, though only one will be used at a time.
- All stations have resource requirements in the form of needed equipment, tools, and number of workers. Tasks can only be assigned to stations with the requisite resources.
- Tasks with different resource requirements are incompatible with each other and so necessitate zoning constraints.

There exist no works in the ALB literature that are able to simultaneously incorporate all these factors and so there is no defined way to integrate them all together. Even worse, there are some characteristics that, by themselves, are unaddressed by any works in the ALB literature. No works currently handle spatial constraints in the context of the limited

space present in a factory, with the TSALBP solution approaches only being able to tackle the limited space in a work station. No works describe the proper way to estimate maximum upper limits for throughput when station duplication is allowed and cost is also an objective and cannot be used as a constraint. No works sufficiently handle the ASP-based dynamic task times, with the only ALB-related implementation in the form of the assembly subgraphs problem not being scalable. No works address how to handle the ambiguity involving which fabrication and assembly tasks should be included when considering integral fabrication and do so in a more scalable way than with the assembly subgraphs problem. And no works sufficiently include fabrication tasks to help further differentiate the usage of different materials and manufacturing processes. How to integrate all the material factors together via both ALB and ASP to obtain assembly throughput and cost tradeoffs, and through that make large improvements to the assembly analysis portion of current aircraft design systems-thinking methods, is thus an unanswered question. This leads to the next research question:

Research Question 2 *How should the integrated assembly sequencing and line balancing problem considering integral fabrication, alternative manufacturing processes, and spatial constraints be solved to obtain aircraft throughput and cost tradeoffs?*

5.2 Addressing the Integrated Assembly Sequencing and Line Balancing Problem's Capability Gaps

There are two main ways to answer the posed research question. The first is to use the most similar, appropriate, and current approach from the ALB literature. Such a work has been identified as the one made by Oesterle et al. [76]. Their work uses a combination of the assembly subgraphs problem, the TSALBP, and the equipment selection problem while being multi-objective in nature. It is the most comprehensive work available that includes the majority of the problems related to the desired material factors and is well-tested. Al-

though it is the most comprehensive work, it still has several gaps: it critically does not include the station paralleling problem and does not address the three issues not handled by current ALBPs. These are: the improper type of spatial constraints being used; the lack of scalability of assembly subgraphs for dynamic task time modeling; and the fabrication and assembly task ambiguity regarding integral fabrication. Going down this route amounts to using a tried-and-true approach that is not quite sufficient and could ultimately lead to subpar throughput and cost improvements when combined with early preliminary aircraft design.

The other way to answer the posed research question is to create a brand new solution framework that specifically addresses all the described capability gaps. This would ensure the maximum number of trades are made regarding material factors to optimize throughput and cost when the new framework is combined with early preliminary aircraft design. The potential pitfalls with this approach are that its complexity could cause it to optimize more slowly than current literature methods and that, due to it being untested, it could actually perform worse quality-wise than literature methods as well. Going down this route would require that the new framework be benchmarked against the most capable literature method available to ensure its benefits outweigh its drawbacks and warrant its development effort and use.

For this work, the second way is the preferred approach due to the promise of minimizing or completely addressing all of the existing capability gaps. Therefore, how to patch up these gaps will be investigated before a new framework is made and presented. This framework will then be tested against the best that is currently available in the ALB literature to prove it is worth using.

5.2.1 Integral Fabrication and Variable Processes

The first ALB gap to be examined is in regard to integral fabrication. This includes the issue of the ambiguity of tasks to be included when trying to represent it along with how to do so

in a scalable way. Focusing on the ambiguity issue first, integral fabrication is an activity that occurs during the fabrication steps, meaning it is part of fabrication, but its output is one part that used to be two parts, meaning it is also an assembly activity. This duality means that, especially for composite materials, assembly cannot be wholly separated from fabrication as they can become interconnected. Changing one can affect the other, and so trying to optimize the assembly implies also optimizing the fabrication to some extent. And herein lies the issue: ALB has traditionally only modeled assembly activities and ignored the fabrication ones, with the assumption that the fabrication activities are wholly unrelated to the assembly ones and done in a separate factory or off site or by another manufacturer entirely. This makes line balances that utilize less integrally-made parts and so require more assembly tasks look artificially worse than line balances that use more integrally-fabricated parts, which is not necessarily true due to the increased handling complexity of integrally fabricating composite parts. Essentially, due to the fabrication steps not being present at all, line balances with more assembly tasks will always seem more expensive.

Trying to fix this by including just the integral fabrication tasks brings with it additional issues: the fabrication tasks will have their own setups and line balances, and modeling and affecting a few through ALB will have a ripple effect on all the tasks that are dependent on it. And as there are many part combinations that can be integrally fabricated and end up as many different tasks, limiting the modeling of fabrication tasks to just the potential integral fabrication ones and its successors can actually result in the inclusion of almost all the fabrication tasks. The only solution to have an equal and unbiased one-to-one comparison, then, is to increase the scope of assembly line balancing to include balancing of all fabrication as well as assembly tasks. This is typically not done because most products are small and inexpensive enough that the fabrication and assembly activities can focus on their own throughput and cost needs without needing to interact with one another. However, with aircraft, the parts are large and expensive enough that they are fabricated for the express purpose of being assembled later, meaning it does make sense to line balance both

fabrication and assembly tasks.

In doing this, should two parts be made integrally, the task is represented as a fabrication task, whereas if they were made separately and afterwards assembled, two fabrication and one assembly task is modeled. This removes the ambiguity of how to represent such scenarios. It also conveniently allows for inclusion of the entirety of a manufacturing process and both its fabrication and assembly tasks. This enables a more one-to-one comparison when looking at different manufacturing processes, regardless of whether parts are integrally made or not, because it is no longer possible to unfairly weigh against processes with more fabrication improvements than assembly improvements. The incorporation of fabrication tasks thus allows modern materials to be properly considered by removing ambiguity for integral fabrication and allowing for more holistically differentiation of manufacturing processes.

An additional advantage of being able to line balance both fabrication and assembly tasks is that it saves a large amount of storage space. Aircraft parts are very large and occupy large amounts of space, meaning it is better to not have any in inventory and have them be assembled as soon as they are fabricated. This is the point of a line balance, that the moment a part is finished with one task it can immediately move to another, in this case going from a fabrication to an assembly task. Thus, being able to line balance both fabrication and assembly tasks for aircraft allows for the minimization of idle time a part is kept lying around and wasting precious space along with the number of parts kept in inventory, also wasting space. This makes line balancing both fabrication and assembly tasks critical for making sure the parts arrive "just in time."

With the issue of ambiguity of integral fabrication task representation and the issue of holistic manufacturing process comparisons solved by modeling both fabrication and assembly tasks, attention is now directed towards how to represent integral fabrication in a scalable way. Conventionally, precedence graphs are used to visually represent an assembly's precedence relations in ALBPs. They break down with integral fabrication allowing

two or more parts to be represented by an assembly task or a separate fabrication task, forming what is basically an alternative subassembly. This breakdown is due to there being ambiguity regarding which tasks should be displayed and how all the precedence relation for each task should be handled. The assembly subgraph problem's alternative subgraphs provides an answer to this by allowing for alternative processes in the precedence graphs. However, the alternative subgraphs themselves break down when there are too many alternatives because they become impossible to visualize as a graph. This turns out to be the case as there are many integral fabrication combinations. A more scalable structure to represent the precedence relations is needed.

Integral fabrication is related to ASP in the sense that it can be represented as a subassembly during the creation of an assembly sequence. It is also related in the sense that it is affected by precedence relation changes resulting from being a part of an assembly sequence. Both of these facts mean that ASP is related enough to integral fabrication for the former's representations to implement the latter and its precedence relations. Because ASP-based precedence relation representations can accommodate for an enormous number of assembly sequences, and hence alternatives, and because ASP must be incorporated with ALB anyways, such representations are ideal for creating a more scalable structure than assembly subgraphs. The only roadbump is the fact that integral fabrication's subassemblies create non-linear sequences, of which only a few representation structures in the ASP literature can handle in a generalizable way.

Marian et al. have one of the only works somewhat able to do this [141]. In their work they completely forego precedence graphs and instead opt to create a database of information describing the product and its parts that allows for and governs the creation of assembly sequences. This database is more implicit than precedence graphs in that it does not outright state the exact relationships between parts in one structure but essentially has rules that describe their relationship. It is slightly more difficult to use but is more flexible as a result, being able to account for non-linear, non-sequential, and even non-

monotone sequences. Based on this the database of information idea can be repurposed to model integral fabrication and its precedence relations. The database is made up of liaison graphs, a table of assembly constraints and precedence relation information, and Boolean precedence relation rules instead. The liaison graphs indicate which parts are in contact and were covered back in subsection 3.1.1. The table of assembly constraints is a close analogue to precedence graphs, encoding geometric and precedence constraints in a matrix form, and the Boolean relations capture any manufacturing precedence relations that cannot be expressed by the table. The liaison graphs can be used to determine which parts can be integrally fabricated, as only parts that make contact with each other are able to do that. Interference matrices generated from projection-based geometric reasoning techniques from the previous chapter can provide information on geometric feasibility and stand in for the table of assembly constraints. This is because there is no set part that must be assembled first in aircraft assemblies, which is not as easily captured in the table of assembly constraints where some order is already presupposed. It is easier to do so with a combination of the liaison graphs and interference matrices, meaning those two combined capture the geometric description of the product. The Boolean precedence relations are subsequently responsible for capturing precedence relations related to the manufacturing process. They will need to encode which types of parts can be integrally fabricated with one another and describe constraints regarding the order parts must be assembled in based on manufacturing process restrictions. Doing this would allow for assembly sequences, assembly alternatives essentially, with and without integrally fabricated parts to be created in a scalable way while bypassing precedence graphs altogether. It would result in a representation structure that on the ASP side is able to look at parts with and without integral fabrication and their resulting geometric and precedence constraints to generate sequences. However, integral fabrication requires a more flexible description mechanism than is able to be provided by just Boolean relations, meaning all these benefits cannot be fully realized just yet. A more flexible way is needed. If such a way were found and if the resulting sequences and con-

straints can subsequently be translated into precedence relations for ALB, the scalability issues encountered with assembly subgraphs would be solved.

It turns out that the ASP-ALB translation process becomes complicated when considering integral fabrication as well. The solution to this issue similarly happens to be flexible enough to also solve the problems with trying to use Boolean relations to capture integral fabrication restrictions on the ASP side. As the solution to these issues is more easily explained in terms of the ASP-ALB translation, that will be addressed first before circling back to replacing the Boolean relations. ASP precedence and geometric constraints are typically part-based in the sense that they dictate which ordering of parts is feasible and which is not. ALB precedence relations are typically task-based in the sense that they dictate the relative ordering of the tasks. ASP constraints being part-based means they cannot account for fabrication tasks while ALB constraints being task-based means they can. This is because parts being fabricated, by nature, do not at all interact with other parts being fabricated elsewhere, and ASP requires that the parts interact with one another. A spar being machined cannot physically interact with a wingskin being machined on the other side of the factory, hence ASP is unable to include regular fabrication tasks or parts. But fabrication tasks can interact with each other such as by competing for resources, hence why ALB is able to include them. This difference causes problems when translating constraints from one analysis type to the other while including integral fabrication. ASP precedence and geometric constraints are conventionally converted into ALB precedence relations on a part-to-task basis by assuming that every part-to-part relation can be transformed into a task-to-task relation. But with integral fabrication, ASP is able to transfer information on the integrally fabricated part while being unable to transfer information on all the fabrication tasks succeeding it and how their constraints changed as a result of the integral fabrication. The altered precedence relations for these fabrication tasks will not be transferable to the ALB side because they were not present on the ASP side to transfer over.

As an example, assume a wingbox assembly sequence of upper skin–integrally fabri-

cated spars and ribs–lower skin for a stitched resin infusion process. This means the spars and ribs must be integrally fabricated first before they are assembled, first with the upper skin and then the lower skin. That ordering can easily be transferred from the ASP side to the ALB side. What is not transferred is the fact that, normally, the spars and ribs all have their own cure, trim, and inspection tasks that are now condensed into just one cure, trim and inspection task due to the integral fabrication. The information on the elimination of several tasks is not present on the ASP side and so is never explicitly transferred to the ALB side as a precedence relation. This makes translation of ASP constraints to ALB relations much more difficult because a small assembly sequence change involving integral fabrication can result in large ALB precedence relation changes without those changes being explicit. And this cannot be fixed by explicitly including fabrication tasks on the ASP side because it does not make sense to include regular fabrication tasks in ASP, regularly fabricated parts do not interact.

The answer to this is found in the work of Topaloglu et al. [207]. In their work, precedence relations for line balancing are expressed as If-Then rules to perform rule-based modeling of assembly constraints as opposed to directly using precedence graphs. There are only as many rules as there are tasks, making this approach very scalable. Additionally, according to them "a rule-based model can include precedence relations involving complex constraints without the need for several precedence graphs," meaning this approach can accommodate more complexity and is more flexible than using just precedence graphs. The flexibility of a rule-based approach means that, on the ALB side, rules for precedence relations for each task can be made that are dependent on the information and constraints from the ASP side. This allows for implicit fabrication task precedence relations to be explicitly obtained for ALB based on which parts are or are not integrally fabricated on the ASP side. And as these rules are always present, precedence graphs or matrices can always be derived from them for line balancing regardless of what information is or isn't explicitly expressed in ASP. Bringing back the upper skin–integrally fabricated spars and

ribs–lower skin sequence example, a rule can exist stating that some cure, trim, and inspection tasks only exist or have alternative precedence relations if the parts related to them are not integrally fabricated. This enables transferring of information from ASP to ALB on those tasks regardless of whether integral fabrication is used or not because rules will exist for both situations. Using rule-based modeling for ASP-to-ALB translation thus allows for sequences and other information and constraints from ASP to always be convertible into precedence relations for ALB.

Circling back to Boolean relations from Marian et al.'s work, their deficiency regarding integral fabrication is they are unable to account for a part's state of whether or not it is integrally fabricated. Topaloglu et al.'s If-Then rules are able to keep track of these states and so has a much easier time expressing both manufacturing process and integral fabrication-related precedence relations. This means, on the ASP side, a database of information using liaison graphs and interference matrices for the geometric constraints and If-Then rules for the precedence constraints would be able to sufficiently describe the product and the desired process to allow for ASP with and without integrally fabricated parts in a scalable way. This is all able to be done without precedence graphs. It would result in a representation structure that on the ASP side is able to look at parts with and without integral fabrication and their resulting geometric and precedence constraints to generate sequences. This representation structure derived from Marian et al.'s work to store the information and constraints on the ASP side, combined with Topaloglu et al.'s rule-based modeling for ASP-ALB precedence relation translation, results in a scalable precedence-relations representation structure able to incorporate integral fabrication that can be used both for assembly line balancing and sequencing.

This concludes the filling in of ALB gaps related to the integral fabrication and variable manufacturing process factors. The next section focuses on doing the same for the capability gaps surrounding incorporating variable task times.

5.2.2 Variable Task Times

In the ALB literature, one of the few ways to incorporate variable task times is by using assembly subgraphs from the assembly subgraphs problem. These allow for alternative tasks that can have alternative task times. The problems with this are that the alternative task structure from the assembly subgraphs cannot be scaled up for large numbers of alternative designs or different assembly sequences and that the task times are known a priori. In reality, the task times will vary according to the designs and assembly sequences chosen as well. To fill in the capability gap regarding variable task times, then, both these problems must be dealt with.

The problem with the assembly subgraphs not being scalable was addressed in the previous section where a representation structure was developed to accommodate precedence relations for both ASP and ALB. Though it was made specifically to cater to the needs of representing integral fabrication precedence relations, it can also serve dual purposes and provide a scheme to account for large numbers of designs and assembly sequences. The ASP portion of this structure can capture the geometric and precedence constraints of practically any design through the use of the liaison graphs, interference matrices, and precedence rules, making it amenable to handling large numbers of designs. Similarly, because this portion is based on assembly sequencing, it by default is able to handle the representation of large numbers of sequences as well. The rule-based ASP-ALB translation portion of this structure is then able to turn all of this information into precedence relations for every task that can be used to perform the line balancing. The scalability problem has thus already been resolved.

This leaves the problem of the task times being known a priori. The task times cannot be known in advance because they are dependent on the design of the product as well as the assembly sequence chosen. Since those two are dynamic, the task times themselves must be dynamic. Given a set manufacturing process where the available fabrication and assembly operations do not change, different designs can change task times primarily by

changing the geometry of the part or assembly being worked on in a task. And given a set manufacturing process, different assembly sequences can change task times by altering the parts being worked on in a task, which itself boils down to changing the geometry of what is being worked on. With the manufacturing process set, then, the variability in task times is dependent on the part and product geometry. When looking at different manufacturing processes, the task times change because different fabrication and assembly operations are being used. Once the different operations are accounted for, the variability again simplifies down to being a function of essentially geometry. Task time variability is thus dependent on part and product geometry and on the manufacturing operations being modeled, meaning whatever method is used to capture task time variability must be able to account for different processes and different geometries.

This boils down to being able to estimate task times and do so given different processes and geometries. Some of the most prominent ways to do so in the assembly literature include Methods-Time Measurement (MTM), MOST, and even Boothroyd and Dewhurst's DFMA system. MTM is the basis of essentially all these techniques and revolves around breaking down manual operations into their constituent physical motions [208]. Each of these motions has a standard time attached to them describing how long they take to perform depending on the part's characteristics such as required tolerance, size, and complexity. The total time it takes to carry out a task is obtained by adding up the standard times of all the individual motions. Multiple simplified versions such as MTM-2 and MTM-3 have since been made. Pertinent works using MTM include those by Krause and Halfmann, Bahubalendruni, and Ortegon et al. [24, 103, 46].

MOST is an even further simplified and modernized form of MTM and has three versions, each focusing on different levels of detail: BasicMOST, MiniMOST, and MaxiMOST [209]. MiniMOST is used for repetitive operations with very short cycle times of less than one minute, BasicMOST is for general purpose operations and models operations up to ten minutes in cycle time, and MaxiMOST is for non-repetitive operations with large

cycle times. According to Jin et al., BasicMOST is up to eight times faster than MTM-2 without sacrificing much accuracy [65]. Along with its comparative ease of use, this has made MOST one of the most popular MTM-based techniques for estimating assembly times. Pertinent works that use MOST or any of its forms include those by Sirirojvisuth, Jin et al., and Panhalkar et al. [58, 65, 101].

DFMA was discussed in Chapters 1 and 2 when reviewing Boothroyd and Dewhurst's assembly work. Whereas MTM and MOST focus on breaking down operations into individual body motions, DFMA instead has a database of how much time it takes to perform simple tasks like grasping, orienting, and inserting parts [16]. Those times can be altered based on the difficulty of performing those tasks, which can depend on the complexity required of those motions, the accessibility of the part, how well the part is secured, and the size and orientation of the part. Because it is built on the concept of DFA, DFMA is also able to offer suggestions on how to minimize part count to improve assemblability as well as provide a theoretical minimum task time if the part or product were fully optimized. It is not quite as flexible for time estimation as compared to the MTM-based systems and hence fewer works use it explicitly for time estimation.

Other notable MTM-based systems include ProKon [210] and Modular Arrangement of Predetermined Time Standards (MODAPTS) [211], which will not be covered.

Another, slightly less direct way to estimating task times than using MTM methods is to use cost estimating methods. Labor costs constitute a large portion of overall costs and are almost always calculated as the product of labor time and labor rate. As a result, many cost estimating methods are also able to estimate labor time as a byproduct of estimating labor cost and total cost. Additionally, as this work is focused on both throughput and cost, cost estimating methods must eventually be examined anyways. As such, a brief review of cost estimating methods will be carried out in the search for methods to estimate dynamic task times.

According to Layer et al., quantitative cost estimation primarily consists of three ap-

proaches: statistical modeling, analogous modeling, and generative-analytical modeling [212]. Statistical modeling relies on historic data and uses techniques such as regression analysis to correlate past products' characteristics to its cost so that the cost of new products with similar characteristics can be captured. Analogous modeling estimates cost by comparing a product with a known cost to one with an unknown cost and analyzing their functional, geometric, and cost structure commonalities; if these commonalities exist, then the cost of the unknown-cost product is similar to that of known one. Generative-analytical models use information on "relevant processes of product creation" to determine a product's cost drivers and how much it should cost. It thus requires a large amount of input data from the user. The names of these approaches are not universal, with different authors calling them different things, but their overall concepts stay consistent. As an example, Curran et al. [213] call statistical modeling as parametric estimation and generative-analytical modeling as bottom-up estimation, while what Sirirojvisuth [58] and Bao and Samareh [214] call process-based modeling and activity-based modeling are part of statistical and generative-analytical modeling, respectively.

Statistical modeling methods, or parametric modeling as it will be called from now on due to its increased usage in aircraft design, have many different levels of detail. They range from low detail and simple overarching costs for an entire aircraft based on its weight to high detail cost estimating relationships for individual tasks based on parameters related to a part's geometry, what type of task is being performed, and how the task is carried out. The lower detail modeling is often seen in older aircraft design methodologies while the higher detail ones can be seen in parametric tools like Galorath's SEER or PRICE Systems's PRICE software or in works like Bao and Samareh's [214] or Northrop's Advanced Composite Cost Estimating Manual [215]. At this more detailed level parametric modeling is able to estimate labor times as well and more resembles bottom-up modeling, as can be seen in the aforementioned tools and works. It is thus possible for there to be overlap between the two.

Although all three cost modeling methods rely on historical data, analogous ones perhaps rely on that data the most due to the need for there to be a similar baseline product to ensure an accurate estimate on top of requiring that data for the modeling in the first place. Additionally, because analogous approaches derive estimates from similarity to previous products, at their most detailed their cost estimating relationships look very similar to the ones for parametric models. This means the cost modeling methods are not entirely mutually exclusive, with a sizable amount of overlap in each.

Generative-analytical methods, or bottom-up approaches as they will now be called for similar reasons to parametric modeling, are essentially the cost estimation equivalent of MTM, allowing granular control and detail in their cost estimates but requiring lots of data and time. As a result, they have the highest fidelity among the three types of methods and can estimate the costs of the most basic of tasks. They are subsequently able to provide detailed labor time estimates too. However, as noted by Curran et al., they require similarly granular data for inputs and so typically need an experienced designer to use them as well as for the design to be in the detailed phase [213]. This makes them difficult to use in early design.

With the most notable direct and less direct ways to estimate task times reviewed, the question is now posed of which one should be used to provide estimates for the integrated assembly sequencing and line balancing problem of interest. Looking at the less direct ways, analogous modeling will not be considered because the need for a similar design to exist is too restrictive when the purpose is to explore new designs and manufacturing systems that are unlikely to have contemporary counterparts. This is even worse for time estimation because there is no guarantee that somewhat similar tasks will yield at all similar times as the task times' dependence on the relevant parameters could be non-linear. Generative-analytical modeling is a very attractive option since it provides a great amount of granularity for cost and is the most similar to MTM-based techniques, allowing for granular time estimates as well. However, it is reliant on the user being able to provide

the extensively detailed inputs and cost relations to be able to estimate an accurate cost. As demonstrated in Ashby et al.'s textbook, it requires fine details on factors like capital costs, overhead rate, scrap rate, load factors, and tool-wear relations [216]. The work of compiling all this data by itself is likely to take longer than the actual design process, making generative-analytical modeling desirable but infeasible. Similarly, as noted earlier, the bottom-up approach generally requires input details from the design that are only really available during the detail design phase, making it unsuitable for this work.

This leaves just MTM-based techniques and parametric modeling methods. Technically, any of the former along with more detailed implementations of the latter would suffice for estimating task times because they are all able to account for different processes and geometries. Ultimately, the more detailed parametric modeling methods are preferred for a few reasons. Dedicated tools such as SEER and PRICE exist for them, meaning large amounts of time do not need to be spent defining every single motion of every task to obtain reasonably accurate time estimates. The lower requirements on granular input data also mean that parametric modeling methods can be used more easily earlier in design, as noted by Curran et al. [213], whereas MTM-based techniques are more similar to bottom-up methods and so are less easily used in the early design phases. And additionally, parametric cost modeling methods serve a dual purpose in that they can estimate costs as well as task times, whereas MTM-based techniques can only estimate the latter. Cost estimations are needed as well as task time estimations and parametric cost modeling methods can model tooling, material, and setup costs, among others, on top of labor costs. More detailed parametric estimation methods are thus the ones of choice to estimate task times.

A scalable representation structure for assembly sequencing and line balancing precedence relations was determined in the previous section and this section identified the use of more detailed parametric estimation methods to obtain variable task times able to account for geometry and process changes. This solves the issues encountered in trying to implement variable task times, which come as a result of tasks constantly changing due

to the design and the assembly sequence being able to constantly change. With this issue settled, attention is turned in the next section towards issues related to implementing spatial constraints.

5.2.3 Spatial Constraints and Throughput Upper Limits

The roadblock with spatial constraints, as explained in section 3.4, is that current ALBP approaches that account for space only account for workstation space instead of available factory space, which is more critical for aircraft due to their much larger size and the sparsity of sufficiently large factories. This is made worse by the inclusion of station paralleling to increase throughput and the product architecture being dynamic. The addition of duplicate stations can easily push required factory space beyond what is available and result in entirely infeasible line balances if factory spatial constraints were not included, depending on the throughput desired. And the product architecture, or aircraft design, constantly changing means that it is possible that a design is chosen that is physically large enough for its parts to not be able to fit inside a factory. Factory-based spatial constraints need to be considered to account for the physical space taken up by aircraft parts so that infeasible rates and designs are not implemented.

The way to consider factory-based spatial constraints can be found by using how the TSALBP handles spatial constraints as reference. The TSALBP allows tasks to be assigned to workstations until the workstation runs out of space for all further tasks assignments, at which point another workstation is opened up. Factory space constraints can be implemented in the same way by allowing for station duplication until the factory runs out of space. Additionally, should a design and line balance with no station duplications by default take up more space than is available, that particular design and line balance combination can be declared infeasible without expending further effort on it. And if all the line balances for a specific design are infeasible due to space, then it would be obvious that the design itself is infeasible to implement as a result of the parts taking up too much space.

Should all designs be infeasible, this would mean that the factory is too small and a larger one should be used or obtained. This can provide insight into tradeoffs regarding what kind of aircraft performance can be expected from building an aircraft in various kinds of spatially constrained environments. This easily allows for implementing factory spatial constraints as a replacement for workstation spatial constraints and solves that problem.

Looking now at the throughput itself, it is not static due to the desire to explore the possible production rate versus cost tradeoffs. In particular, the designer will want to know what the designs with the minimum attainable cost and the maximum feasible production rate are to have as comprehensive a picture of the design space as possible. This is to know the physical limits of the factory they are working with and how those limits affect both the aircraft and the manufacturing performance and to see what balance of cost versus throughput is best. While there is a built in physical lower limit that can be used to bound the minimum cost, given that at least one task and one station must exist, there are no built in upper limits to bound the maximum throughput if station duplication is allowed; stations can just be endlessly duplicated and paralleled to obtain an infinite production rate. Cost has typically been used as a constraint since infinite cost from the infinite throughput is not desired, but in this situation the cost is an objective and to bound it to any particular value is arbitrary in nature, extremely subjective, and defeats the purpose of trying to inform the designer of their factory's capabilities and limits. All other works in the ALB literature deal with this issue by setting the throughput to a specific value or having it be an input, even multi-objective works that have both cost and throughput as objectives like McMullen and Frazier's [177], which does not address the problem. A method to determine what the maximum possible throughput is that also minimizes subjectivity and the number of arbitrary limitations must thus be developed.

Physical restrictions are the most direct and impartial type of constraint that can prevent infinite throughputs because they represent resource constraints based on reality. Factory spatial constraints, like the ones just addressed, are the simplest to consider of such con-

straints and the most well defined–there are only a few available factories in the world with sufficient space to build commercial aircraft. The concept with which they can be used to provide physical upper bounds for throughput is therefore simple: stations can be duplicated to increase throughput until there is no more space to further duplicate stations and increase throughput, at which point the maximum production rate is achieved within the available confines of that factory. Station paralleling is basically carried out to fill up the available space in a factory until it is used up. This provides a conceptually simple solution to the problem that is neither subjective nor arbitrary. As a result, factory spatial constraints will serve two purposes and be leveraged to help cap production rates on top of just making sure a design's line balance fits inside the factory.

However, despite being simple in principle, the execution of this concept is more complex. Blindly duplicating stations is not an optimal way to increase production rates and the mechanism with which the production rate is actually varied must also be determined, both of which require a slightly more nuanced approach. A few concepts must be defined first. The throughput is set by the cycle time, which must be equal to or larger than the effective station times of all the stations. The effective station time of a station is the station's station time divided by its number of instances. If a station has a single duplicate station, then its effective station time is half its station time because there are two instances of it. This means in a line balance where the cycle time is set, all stations with effective station times larger than the cycle time will be duplicated until their effective station times are lower than the cycle time, with any other outcome yielding an infeasible line balance. Situations involving set cycle times are thus dealt with in the same way as all the other ALBPs with parallel stations.

When the goal is to maximize throughput, there is no set cycle time to impose a specific number of duplications for each station; instead, stations are duplicated to "fill up" a factory's available space until there is none left at which point the cycle time, based on its definition, is equal to whatever the largest effective station time is. Every instance of

paralleling introduces a new station instance and so uses up space and increases cost. As such, if a station is paralleled while its effective station time is lower than the current cycle time then all it does is increase cost while using up extra space without increasing throughput. This is why blindly paralleling stations is not effective. It results in many stations that are not bottlenecks to be duplicated. Every single station duplication must thus be used strategically to increase the production rate to be able to obtain the maximum rate. Based on this, the most effective procedure is to incrementally duplicate stations with the largest effective station times, the bottleneck stations, until there is no space left. This more directed approach maximizes the utility of every duplication by having every single one affect the throughput. As a result, it guarantees the minimum cycle time given the design, tasks, and task times being balanced. For this reason, it will be the procedure of choice when maximizing production rate instead of just duplicating stations at random.

The concept of being able to calculate the maximum physically feasible production rate brings with it an interesting problem that must also be addressed. In a multi-objective problem focused on cost and throughput, it is desired to have designs and line balances with a spread of different costs and throughputs: given the procedure to calculate the maximum rate, how will all the other designs and line balances with production rates less than the maximum be obtained? To provide more context, in all ALB works, cycle times are given as an input and set a priori, as in the SALBP-1, or they are indirectly determined by only having a set number of stations with one instance each, as in the SALBP-2. And in the majority of multi-objective works involving throughput in the literature, the different throughput levels are obtained essentially by solving slightly altered versions of the SALBP-2 and keeping the non-dominated solutions. With the problem being tackled in this work, however, the multi-objective throughput and cost nature of the problem means that the cycle times cannot be set a priori or given as an input. And the station duplication aspect of the problem means that merely having only a set number of stations does not actually restrict the possible cycle times at all because they can just simply be duplicated for

increased throughput. McMullen and Frazier's work is just about the only multi-objective ALBP work involving throughput and cost in the literature that considers station duplication, and even their work provides no guidance because they assume the throughput is provided by an outside source [177]. The conventional ways in the literature to obtain a spread of cycle times and throughputs thus are not useful. And on top of that, the factory space constraints are only useful in providing an upper limit on throughput by using up all the available space; they do not provide any information on the throughput levels attained by using up less than all the available space. This leads to a situation similar to the SALBP-E where, because the cycle time is so malleable, an iterative procedure must be used. The difference is, though, that the SALBP-E solution procedures are able to leverage upper and lower bounds on both the number of stations and cycle times for its iterations. Meanwhile, the problem being tackled effectively does not have limits on the number of stations due to the existence of station duplication, causing it to be significantly less bounded and making it much more difficult to solve in a guided way.

This means one of the only ways to obtain intermediate solutions that have less than the maximum physically feasible throughput is to essentially randomly guess how many duplications each station should have, so long as the total space used does not approach the maximum available space. This forces the number of station duplications to become a sort of design variable. The number of such variables could be as large as the number of tasks. Along with the fact that the number of stations, and thus the number of such variables, is not set and constantly changing, this creates an immensely complex problem. Meta-heuristic methods are usually used to handle complex multi-objective problems like this but themselves are unable to offer a more efficient approach. They too must resort to hoping they can obtain a non-dominated solution because they cannot directly try to set an objective in a multi-objective problem, they can only operate on design variables affecting the objectives. In this case that involves the station duplications which, due to their number and that number constantly changing even in the same problem, involves some degree

of guessing at minimum. Such random guessing is inconsistent and undesirable because, given the very large design space of variable product architecture and assembly sequences on top of the line balancing itself, the chances of obtaining a solution that is on the Pareto frontier and non-dominated are infinitesimally small. A better way to approach this problem given this characterization, where the upper bounds of one objective in the form of the maximum possible throughput is already found and a minimum one of zero is a given, is to directly fix the throughput objective at some level, determine the solution that optimizes the other objective, and repeat this to incrementally build up the non-dominated solution space. Searching through the optimization literature, a method describing such an approach is found in the form of the ε-constraint method [217]. In the ε-constraint method, one or more of the objectives are set as inequality constraints and held at constant values while the other objectives are optimized. In a problem with only two objectives, this means one objective becomes an inequality constraint while the other is optimized, essentially turning the problem into a single objective subproblem. The inequality constraint at which the constraint objective is set at is the ε-constraint and its value is varied and the resulting subproblems solved many times to incrementally build a picture of the design space, generally represented as a Pareto front.

One of the method's main weaknesses is that it is difficult to select an appropriate value for the ε-constraint. However, for this work's problem that is not an issue. Once the maximum physically feasible throughput has been determined, the ε-constraint can just be set to sweep through all the throughputs below it using the throughput as the constraint and minimum cost as the objective. By setting the throughput as a constraint, an upper limit is provided for station duplication beyond just the available factory space, allowing for designs and line balances with intermediate to low production rates to be obtained. This method is attractive because, while it is still iterative, it is guided. It breaks down the more complicated dual-objective problem into a series of simpler single objective ones akin to SALBP-1s in that the cycle time is set while a cost or cost-like objective is optimized.

The smaller subproblems being more like SALBP-1s with set cycle times means tried and true methods that minimize cost while involving station paralleling, like Leiber et al.'s [176], are applicable again, making these smaller subproblems more feasible to tackle. It also directs effort towards the obtaining of non-dominated solutions since single objective problems only have one optimal solution and, with the ε-constraint method, a solution at an ε-constraint is almost guaranteed to be on the Pareto front. This is as opposed to searching through the enormous variable space resulting from setting the number of station duplications as a variable and hoping a solution is non-dominated, which is what would need to be done for multi-objective meta-heuristic methods. Additionally, because the ε-constraints have physical meaning, they allow the designer to pick the throughput levels they want to explore, allowing for finer control of which areas of the design space they want to examine. Due to all these benefits, the ε-constraint method will be used as a global optimizer of sorts, responsible for breaking down the main problem into smaller subproblems that can be iteratively solved to build up the design space.

The issue of the ALB literature only considering spatial constraints of the wrong type was addressed by using the TSALBP as reference to implement factory-based spatial constraints that are the correct type. The factory spatial constraints are then used as the physical constraints with which to provide an upper bound on the maximum physically feasible production rate in a multi-objective throughput and cost problem that includes station duplication. How to effectively duplicate stations to actually maximize the throughput was then identified. A problem was afterwards encountered where, because of the problem of interest's characteristics, there are no directed ways to obtain solutions with lower-than-maximum throughputs without having to basically randomly guess. A solution was found in the form of the ε-constraint method, which can iteratively and incrementally build up the Pareto front and design space in a guided way and also simplifies the multi-objective problem into a series of simpler single objective ones. All issues related to spatial constraints and upper throughput limits have thus been addressed. This means all the individual insuf-

ficiencies of current ALBP works have been dealt with. The next step, then is to determine how to combine everything together.

5.2.4 Integration of All the Desired Problem Types and Characteristics

All of the individual capability gaps not directly addressed by existing ALBPs have been filled in and the ensuing issues encountered solved. The last step is to combine everything together, both the new capabilities as well as the relevant ALBPs. Due to the myriad of characteristics present, no works in the literature have done this yet. However, indirect guidance can be gleaned from existing works and how they integrated their characteristics can be used to help integrate the ones of interest.

One of the most notable characteristics is that the problem being tackled is not just an ALBP, but an integrated ASP and ALB problem. Based on the works by Milner et al. [132], Hong and Cho [133], Tseng et al. [142], and Lu and Yang [149], among many others, ASP-ALB integration typically has the ASP side provide assembly sequences to be used as the backbone for the precedence relations for ALB. This replaces the typical precedence graphs used to very straightforwardly assign tasks to stations. Instead, a set of overarching geometric and precedence constraints based on the manufacturing process and part/assembly geometry exists and governs and sets limits on all the assembly analyses. These are used to generate assembly sequences, which refine the constraints to be used for ALB. The aforementioned works all have their own different structures and methods to implement this. The problem being tackled in particular thus needs such a structure that is able to handle integral fabrication, variable task times, and better capture manufacturing process differences.

A structure with these capabilities was developed back in subsection 5.2.1 and subsection 5.2.2. Because of the sheer size and scope of the integrated problem being tackled, all the relevant information is stored implicitly first before being extracted and used, as opposed to directly using precedence graphs like many previous works. The part and as-

sembly geometry details are captured by liaison graphs and interference matrices while the manufacturing process details are captured by precedence If-Then rules. These create a database that can be used to make the assembly sequences and are additionally able to account for integral fabrication. After the sequences are generated, rule-based modeling is used to convert the sequencing information into task information and precedence constraints that can be used by the ALB side. Then, parametric modeling techniques are used to estimate task times and costs for all the tasks to be assigned based on the assembly sequence and geometry used. From there, line balancing is performed. This structure therefore directly connects ASP with ALB and is equivalent to the structures used in previous integrated ASP and ALB works and so is satisfactory for the problem of interest.

How ASP can be combined with ALB in general has been addressed. How it can be combined with this particular ALBP and all of its traits, as well as how all said traits themselves can be combined, is examined next. Oesterle et al. have one of the more inclusive works and what they did was first identify all the traits they were interested in, which for them included characteristics from the assembly subgraphs, equipment selection, and multi-objective problems [76]. The effects of these traits on a standard ALBP were then explained and they afterwards replaced the corresponding generic assumptions and structures, such as assembly subgraphs replacing typical precedence graphs, equipment considerations replacing resource-less task-to-station allocations, and multi-objective goals replacing the normal single-objective optimization. Leiber et al., whose work has similarly diverse characteristics, combined their characteristics in a similar manner by first identifying them and then mapping out which assumptions in a generic ALBP their characteristics would overwrite before replacing them [176]. They looked at the same problem types as Oesterle et al. except they used station paralleling instead of multi-objective optimization, resulting in them eliminating the assumption of cycle time having to be larger than the largest task times but keeping a single objective focus. Based on examination of other capable ALBPs in the literature that combined many characteristics, which include the works by Rashid

et al. [154], Chica et al. [75], Tseng et al. [142], and Lu et al. [149], among many others, this process appears to be the trend on how to perform integration of all the desired characteristics into an ALBP. All the characteristics of interest and their effects are first identified, after which a mapping of how the desired characteristics can affect and change a generic ALBP, such as a SALBP, and its assumptions and structures is carried out. Said assumptions and structures are then replaced with the desired traits, yielding a problem that combines everything desired. As a result this process will be used for this work.

The characteristics of interest in this work are all related to the different factors aiding integration of assembly and aircraft design. These, as have been stated many times, are variable throughput and cost, integral fabrication, variable manufacturing processes, spatial constraints, and the ability to perform ASP with ALB. Due to this close relationship, it is more convenient to go over how these factors affect a generic ALBP as opposed to how each specific characteristic does so. The order in which these factors and their effects on a generic ALBP will be covered is based on the significance of their impacts on the overall problem.

The variable cost and throughput factor is the most overarching because it affects how the entire problem is approached. The both of them being the main focus means the work must be multi-objective in nature, incorporating the multi-objective ALBP, and necessitates that all the designs and line balances generated will use those metrics as figures of merit. For the cost, it means that the many different types of cost, such as labor, equipment, tooling, and material, must be accounted for in greater detail. It is not sufficient to use the number of stations as an analogue for how much a design or line balance costs as a result, like is done in the SALBP-1. Similarly, as the throughput must be varied, a SALBP-1 cannot be implemented, and because it is desired to obtain a range of throughputs, just solving for the absolute minimum cycle time will not suffice either, like in the SALBP-2. Another impact of having throughput as an objective is that to have as varied a throughput as possible, the station paralleling problem's approach is used where the lower limits on cycle time

are removed through the use of station duplication. With cost also being an objective, this means physical factors such as spatial constraints must be included to cap the production rate. On top of that, as discussed in subsection 5.2.3, to properly incorporate the upper limits of production rates given a factory and its spatial constraints, the ε-constraint method must be used with throughput as the ε-constraint. This means the ALBP will not only need to be multi-objective in nature, but be so in a specific way. The maximum production rate given a particular factory along with the design and line balance yielding that rate must first be found. The lowest cost designs and line balances for progressively lower minimum production rates must then sequentially be found and the Pareto front representing the design space incrementally built up. This is a stark contrast to almost all other multi-objective optimization procedures where the Pareto front is essentially built all at once, but is currently the only known way to obtain designs and line balances with intermediate production rates in a guided, non-random manner for this type of problem. This resembles the SALBP-E and uses its iterative nature. The multi-objective throughput and cost factors thus set how the overall optimization of the ALBP must be carried out.

Incorporation of the spatial constraints factor causes many changes. The majority of them have to do with its interactions with the variable throughput and cost factor and so were already just covered. On its own the desired spatial constraints are similar to the TSALBP in that the space usage of workstations must now be tracked where it was not before. The main difference from the TSALBP is that instead of filling up a workstation's space when assigning tasks, the factory's space is filled up instead by duplicated and non-duplicated stations and their space usage. The range of the spatial constraints' effects therefore vary significantly in magnitude because, by itself, it simply requires accounting of an additional parameter while with other factors, it changes how the entire problem is approached.

The inclusion of ASP as a major factor indicates that the problem is an integrated ASP and ALB problem. It subsequently has very notable effects. The static precedence graphs

normally used to provide precedence relations for line balancing are liable to break down when assembly alternatives in the form of assembly sequences are included. Assembly subgraphs and its related ALBP are not able to handle the number of alternatives. This means that the ASP-ALB connection structure developed in previous sections must be used as it addresses this issue and several others as well. Similarly, the shifting precedence relations and dynamic part geometries that accompany this work's ASP implementation means that task times are not static either. Consequently, parametric cost and time estimation models must be used to obtain task times, which can no longer be determined a priori like in more generic ALBPs.

The integral fabrication factor allows parts to essentially be pre-assembled in a fabrication task as opposed to an assembly task. To properly and fairly compare line balances with and without integral fabrication, all fabrication tasks need to be included during the line balancing as opposed to just assembly tasks in more standard ALBPs. Additionally, integral fabrication can significantly affect precedence relations for both ASP and ALB, meaning conventional precedence graphs for ALB are not suitable. The assembly subgraphs structure from the assembly subgraphs problem is likewise not suitable because it cannot scale to large numbers of integral fabrication combinations. This means that the normally used precedence graphs are replaced by the same ASP-ALB connection structure as was used for the ASP factor.

The variable manufacturing processes factor is the last one to be covered. It is used to compare different processes to see which perform better. As a large portion of the differences in manufacturing processes are related to their fabrication tasks and not just their assembly tasks, to have a better comparison between them fabrication tasks will also need to be included in the line balancing. This is similar to the integral fabrication factor. Different manufacturing processes also require different equipment sets, necessitating resource accounting like in the equipment selection problem. Different tasks require different resources, and so tasks can only be assigned to stations that have the appropriate equipment

sets and number of workers. Similarly, it is impractical to have completely different sets of equipment in the same station, leading to zoning constraints where some tasks cannot be assigned with others in the same station. This differs from a SALBP or generic ALBP where tasks can be freely assigned to stations.

The effects of all the factors of interest and their characteristics along with how they alter a generic ALBP and its assumptions and constraints has been described. With all of this in place, solving the problem with all the desired characteristics incorporated is as straightforward as just implementing it with all of the described changes in place. This involves replacing all of the relevant generic ALBP assumptions and constraints, carrying out the outlined flow of the new problem, and obtaining the ideal line balances. The framework and the accompanying details for the actual implementation of all this are presented in section 5.4.

5.3 Hypothesis 2: Combining Different Problem Types and Techniques

The integrated ASP and ALB framework and its development described in the previous section combines all the material integration factors into one problem and solves it. These factors include integral fabrication, spatial constraints, variable manufacturing processes, and variable throughput and cost. The framework is able to do this while incorporating ASP since ASP is used to provide the precedence relations for ALB. This is done by integrating all the appropriate ALB problem types together. In the course of doing this, all the different material factor characteristics that were not addressed by any works in the ALB literature are also dealt with and incorporated into the framework. This includes accounting for factory space, calculation of maximum physically feasible throughputs, modeling of fabrication tasks, incorporating variable task times, and using more implicit precedence relations such as liaison graphs, interference matrices, and If-Then precedence rules as opposed to relying only on more explicit precedence relations such as precedence graphs. The framework thus addresses the integrated ASP and ALB problem by integrating numer-

ous analysis procedures related to it and multi-objectively optimizing it via the ε-constraint method to perform assembly tradeoffs. This framework is therefore able to fill all of the gaps presented in Gap 2 and perform all the desired material factor-related assembly analyses that can later be used to enhance the feedback loops between the assembly and aircraft design disciplines and so better integrate them, leading to the following hypothesis:

Hypothesis 2 *If a framework is used to combine assembly sequencing with assembly line balancing problem types, solution approaches, and additional capabilities most relevant to the material factors, then the integrated assembly sequencing and line balancing problem with material factor considerations can be solved such that better throughput and cost tradeoffs are obtained.*

To test this hypothesis, an experiment must be performed where the developed assembly framework is compared with the most relevant and capable method in the literature that tries to incorporate ASP, ALB, and the material factors. The developed assembly framework will then need to show that it is able to obtain better production rate and cost tradeoffs than the literature method in this experiment. This would justify the effort spent to create and use this framework as opposed to simply using the current best approach from the ALB literature. This experiment thus ties back to what was mentioned at the beginning of this chapter, that the decision to develop a new framework to answer Research Question 2 carries with it the burden that the new framework be tested against the current state-of-the-art.

The experiment has an additional purpose in that it serves as a demonstration for Conjectures 1.1 and 1.2 and showcases how the techniques discussed in those conjectures are actually applied and what the ensuing results are. If the experiment demonstrates that the new framework can do all this, shows that the framework can make all the claimed material factor tradeoff capabilities to prove Hypothesis 2, then the new framework will be able to be used to establish new assembly discipline feedback loops with the aircraft design disci-

pline. That would bring this work one step closer to proving the Overarching Hypothesis.

This first experiment, Experiment 1, is covered in section 5.5 while the implementation of the integrated ASP and ALB framework is covered in the proceeding section.

5.4 Integrated Problem Solution Framework

The flow of the framework addressing the integrated ASP and ALB problem considering the integral fabrication, spatial constraints, variable manufacturing processes, and variable throughput and cost factors is as follows. First, information related to the geometry and manufacturing process used for the assembly is extracted and stored as liaison graphs, interference matrices, and precedence relation If-Then rules. These describe the geometric and precedence constraints to be used for ASP. The parts, if any, to be integrally fabricated are determined and an assembly sequence is generated. This sequence, combined with the aforementioned geometric and precedence constraints, is translated into precedence relations usable by ALB through rule-based modeling of assembly constraints. This all constitutes the ASP-ALB connection structure and is what combines the two assembly analyses. After this is done, parametric cost and time estimation models are used to obtain all the tasks' task times and costs. Line balancing is then performed where tasks are assigned to stations according to precedence relations, resource applicability, and zoning constraints. The amount of total space used is tallied up and, if it is more than the available factory space, that line balance is deemed infeasible. If it is less than the available space, station paralleling is performed where bottleneck stations are sequentially and iteratively identified and duplicated until there is no space left. This process is repeated for as many sequences and line balances as it takes until the maximum throughput or minimum cycle time is identified. Subsequently, this entire process is repeated for incrementally higher minimum cycle time constraints with the objective being to minimize cost at each constraint level. This is used to build up a Pareto front piece by piece. The optimization is finished when the throughput constraint level reaches a throughput sufficiently close to

zero and the Pareto front is fully built. This is summarized in Figure 5.1.

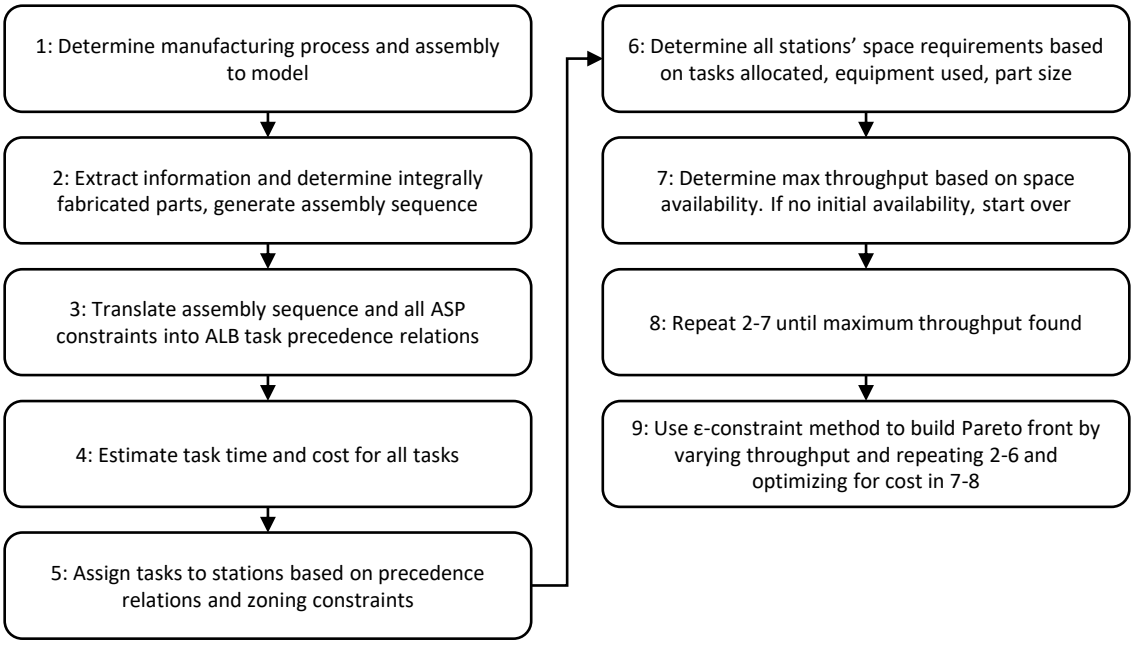


Figure 5.1: Summary of Integrated Problem Framework Steps

It was noted in section 5.1 that the assumption is made that there is to be no intermixing of manufacturing processes during the sequencing and line balancing. This means the steps in Figure 5.1 must be carried, out at minimum, as many times as there are manufacturing processes to be able to compared those processes, with each instance only analyzing one process.

5.4.1 Assembly Sequence Planning Implementation Details

The specific steps used to carry out the assembly sequence planning are presented in Figure 5.2.

Liaison graphs and their mathematical implementations, liaison matrices, were covered back in subsection 3.1.1. The liaison graph is made by generating a node for every part in the assembly and constructing edges connecting all nodes that their respective parts physically contact. The liaison matrix is $n \times n$ in size, with n being the number of parts.

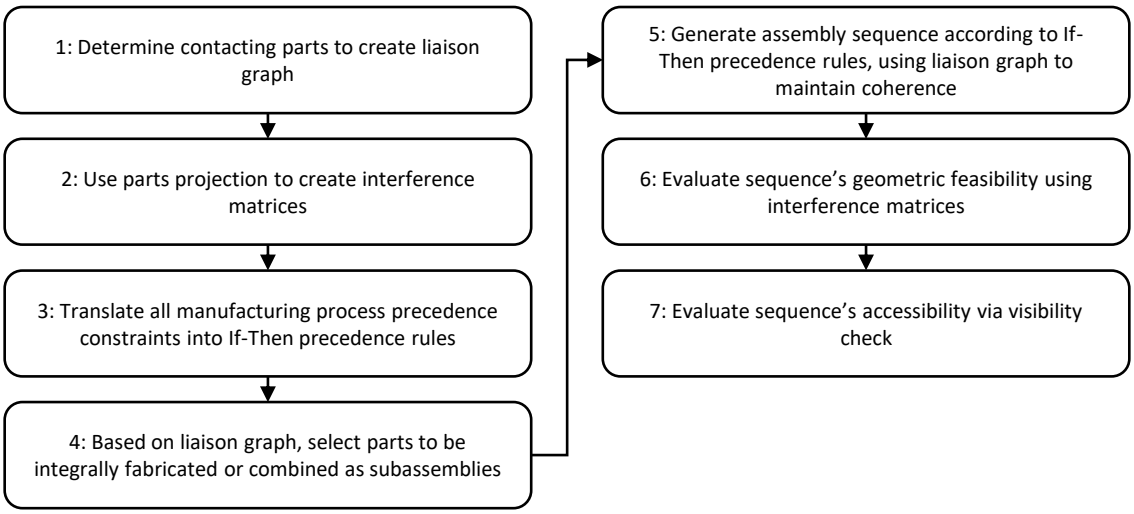


Figure 5.2: Assembly Sequence Planning Detailed Steps

Element e_{ij} is 1 if there is an edge connecting nodes p_i and p_j, basically meaning that those parts physically contact each other when they are fully assembled. It is zero in all other cases. The liaison graph is significant in that parts can only be integrally fabricated if they contact each other when fully assembled, meaning integral fabrication can only occur between parts that have an edge connecting their respective nodes in a liaison graph.

There are two different types of integral fabrication, which first need to be explained before explaining the process by which integrally fabricated parts are determined. The first type is between what will be called "minor parts" and "major parts" for the rest of this work. Major parts are the physically larger parts in the assembly such as the spars, the wingskins, fuselage skins, fuselage frames, and the ribs, specifically the ribs' webs. Minor parts are those that provide additional stiffness to the major parts and include spar stiffeners, stringers, frame stiffeners, rib stiffeners, and rib shear ties. Integral fabrication for composites in aircraft assemblies has most commonly been done between minor parts and major parts and so will be called minor integral fabrication. Major integral fabrication, therefore, is integrally fabricating two or more major parts. This is significantly rarer because it is harder to certify the integral fabrication of two major parts and was not possible

until recent technological advances were made. This includes Boeing developing the capability to integrally fabricate an entire fuselage barrel as one part instead of several separate composite skins and the advent of the stitched resin infusion process, whose stitches can provide similar reliability to fasteners and so allow major parts in wingboxes and fuselages to be integrally fabricated.

To determine the parts to be integrally fabricated, a major part is first selected. If major integral fabrication is to be done, another major part that shares an edge with the first part on the liaison graph is selected. Given the current manufacturing technology, major parts that are integrally fabricated also have all of their associated minor parts be integrally fabricated as well. Major parts' associated minor parts are the minor parts that share liaison graph edges with the major ones. Additionally, major integral fabrication can be done with more than just two parts, so parts that share liaison graph edges with the ones already selected can be continuously added. Due to limitations of current technology, it is assumed no more than three major part types can be integrally fabricated when analyzing wingbox assemblies while the entire fuselage skin can be integrally fabricated. Here, part type refers to upper wingskins, lower wingskins, front spars, rear spars, ribs, etc. If minor integral fabrication is to be done, the major part's associated minor part is selected to be the one to perform integral fabrication with. Every time a set of parts is selected to be integrally fabricated, all the liaison graph edges connected to the parts' respective nodes are removed. No more parts can be chosen to be integrally fabricated if there are no more edges.

A complementary concept to integral fabrication is the subassembly, which will be studied in this work as well since non-linear sequences are already being considered due to the inclusion of integral fabrication. Where integral fabrication uses a fabrication task to preassemble parts, subassemblies use an assembly task to preassemble parts. Given that integral fabrication already essentially creates a subassembly, just with a fabrication task, the handling of subassemblies and how they are determined and generated is exactly the same. In fact, it is possible to mix subassemblies with integrally fabricated parts, which

will be included in this work to see if such a mixture offers any advantages over using just one or the other. It is to be noted that the liaison graph being operated on when removing edges for the integral fabrication is also shared with the one used for determination of parts to be put in subassemblies. If there are no more edges after the integral fabrication selection is done, then no subassemblies can be made either.

As described back in subsection 4.3.1 and as part of Conjecture 1.2, the assemblies of interest are monotone, sequential, and coherent in nature, meaning their geometric constraints can be adequately captured by interference matrices generated using projection-based methods. Interference matrices were covered back in subsection 3.1.1. Due to the relatively simple geometries of the assemblies of interest, a maximum of three interference matrices for the wingbox and two for the fuselage are needed. The three for the wingboxes include one for each positive Cartesian direction, with information for the negative Cartesian directions obtainable by just transposing the matrices. Given a 2D cross section of a wingbox whose leading edge is pointed to the left, the +X direction goes into the page, the +Y direction points to the left of the page, and the +Z direction points to the top of the page. The two directions for the fuselages include one for the outwards radial direction and one for the longitudinal direction going out of the page, with the inward directions also obtainable by transposing the matrices. The matrices are $n \times n$ in size, with n being the number of parts. Element e_{ij} is 1 if part p_i is obstructed by part p_j as part p_i is translated in the direction of the associated interference matrix and 0 if it does not. Parts cannot interfere with themselves and so e_{ij} is zero along the matrix's diagonal. For the radial interference matrix for fuselages, the specific radial direction is different for each part and is basically an analogue to Yu and Wang's extended interference matrix where parts can be translated along their local Z coordinates [218]. The radial direction in general is an outward pointing vector whose origin is the fuselage's center, with the angle of that vector for each part roughly equal to the vector that passes through the center of the fuselage and the circumferential midpoint of the part. When parts are integrally fabricated or are part of

a subassembly, their respective rows and columns in the interference matrices are replaced by a single row and column, indicating it is essentially a single part now. This is done after the normal interference matrices without integral fabrication or subassemblies are made. The integral part or subassembly interferes with every part that its constituent parts interfere with or are interfered by, which is reflected in its row and column in the interference matrices. Examples of this for a wingbox are shown in Figure 5.3.

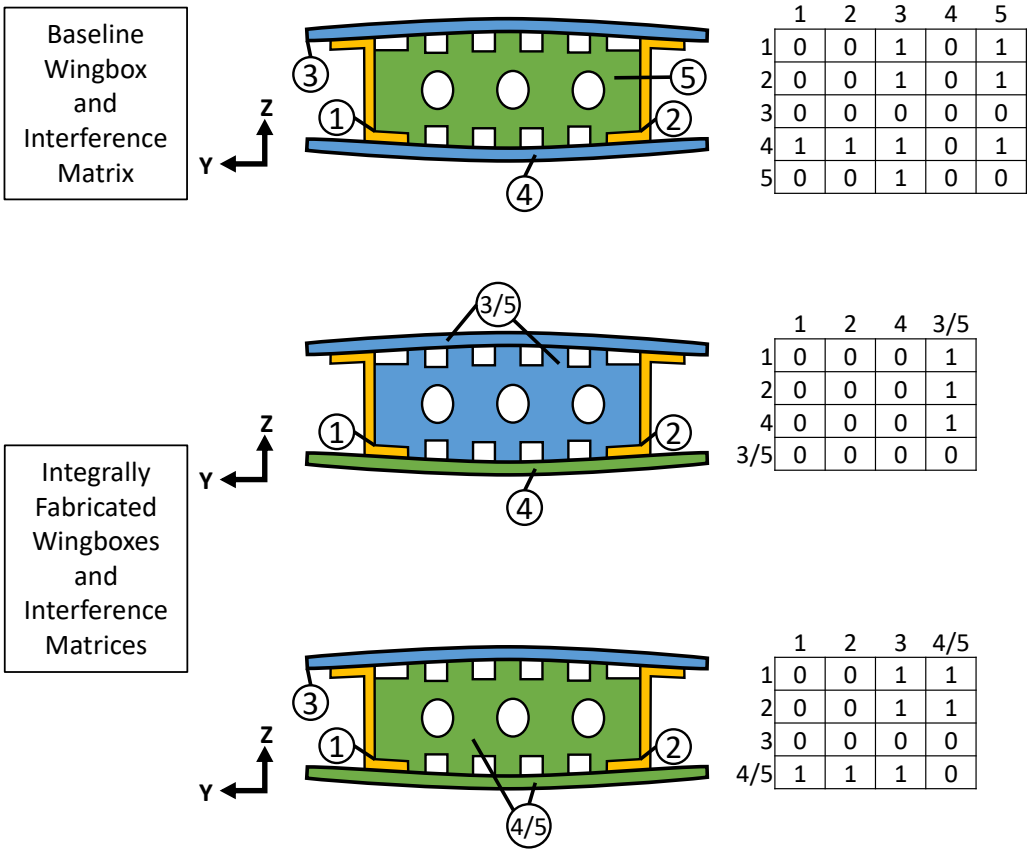


Figure 5.3: Wingboxes with Different Levels of Integral Fabrication and Their Respective +Z Interference Matrices. Notice Bottom-most Wingbox Has No Feasible Sequences

The interference matrices are created via a bounding box projection method very similar to the one discussed by Zhang et al. except that the geometries are simple enough to not need the cruciform part of Zhang et al.'s method and that the bounding boxes are not necessarily all axis-aligned [124]. The interference matrices are generated sequentially by carrying out the projection method for every part-part combination in the assembly, there-

fore necessitating n^2 projection processes given n parts. Though the number of projections can become large with a large number of parts, Zhang et al. demonstrated that it is still significantly more efficient than other methods [124]. Additionally, during early preliminary aircraft design, it is unlikely for the part count to become excessively high.

To perform the method, two parts are selected and their bounding boxes projected in the interference matrix direction onto a faraway plane normal to said interference matrix direction. For the fuselage's radial direction, the projection direction and projection plane are based on a sector starting at the center of the fuselage that is sufficiently wide to completely encapsulate the smaller of the two parts and extends beyond the fuselage. The projection direction is the vector that starts at the fuselage center and passes through the circumferential midpoint of this sector. Likewise, the projection plane is the plane far away from the fuselage that is normal to this vector. If the sector encapsulates the entire fuselage, then both parts simultaneously interfere with each other in both the outward and inward radial directions because they essentially cannot be assembled or disassembled in the radial direction.

If the bounding boxes projected onto the faraway plane do not overlap, there is no interference between the two parts and the projection process is carried out for the next part pairing. If there is overlap, multiple steps are taken. First, the maximum and minimum interference-direction coordinates for both parts are determined. If said maximum coordinate for the first part, part P_i, is smaller than or equal to said minimum coordinate for the second part, part P_j, then element e_{ij} in the interference matrix is 1 and element e_{ji} in the interference matrix is 0. This indicates that part P_j is "above" part P_i and so interferes with part P_i when P_i is being translated in the interference matrix direction. Likewise, part P_i does not obstruct P_j when the latter is being translated in the interference direction. If the minimum coordinate for the first part is larger than or equal to the maximum coordinate for the second part, the opposite is true, with element e_{ij} in the interference matrix being 0 and element e_{ji} in the interference matrix being 1.

If these relationships between minimum and maximum interference-direction coordinates between both parts do not exist, then the overlap area between the two bounding boxes is determined. Because the parts came from geometry that went through the early preliminary aircraft design process, a finite element model or CAD representation for them exists where their geometry is described by surfaces or elements of various shapes. Surfaces are collections of elements that have continuous vertices and the same normal vector. All of the parts' surfaces that are encapsulated by the overlap area, except for surfaces that are parallel with the interference direction, themselves have their bounding boxes projected in the interference direction onto a plane normal to the interference matrix direction. Elements are only projected and used if the surface is curved, like a wingskin panel, and so must be broken up into individual elements for more accurate geometric representation. In those cases they replace surfaces for the following steps. Part P_i's surfaces' interference-direction coordinates are compared with the interference-direction coordinates of all of Part P_j's surfaces whose bounding boxes overlap with Part P_i's surfaces. From there the following scenarios are possible.

Interference matrix element e_{ij} is 1 and e_{ji} is 0 if, for every surface in Part P_i that has one or more overlapping surfaces in Part P_j:

- In cases where Part P_i's surface's interference-direction coordinate is different from the overlapping Part P_j's surfaces' interference-direction coordinates, Part P_i's surface's interference-direction coordinate is smaller than all of said Part P_j's surfaces' interference-direction coordinates

  AND

- In cases where Part P_i's surface's interference-direction coordinate is roughly the same as the overlapping Part P_j's surfaces' interference-direction coordinates, Part P_i's surface has a normal vector pointed in roughly the positive interference matrix direction while all of said Part P_j's surfaces' are pointed in roughly the negative

interference matrix direction

Interference matrix element e_{ij} is 0 and e_{ji} is 1 if, for every surface in Part P_i that has one or more overlapping surfaces in Part P_j:

- In cases where Part P_i's surface's interference-direction coordinate is different from the overlapping Part P_j's surfaces' interference-direction coordinates, Part P_i's surface's interference-direction coordinate is larger than all of said Part P_j's surfaces' interference-direction coordinates

  AND

- In cases where Part P_i's surface's interference-direction coordinate is roughly the same as the overlapping Part P_j's surfaces' interference-direction coordinates, Part P_i's surface has a normal vector pointed in roughly the negative interference matrix direction while all of said Part P_j's surfaces' are pointed in roughly the positive interference matrix direction

If neither of the above scenarios occur, then both e_{ij} and e_{ji} are 1.

As an example, for a Z direction interference matrix for a wingbox where the entries for the front spar and one of the ribs are being calculated, Z-axis aligned bounding boxes are first created around the both of those parts. The front spar and said rib's bounding boxes are projected onto a XY-oriented plane far away from the wingbox. An overlap between their bounding boxes is identified. The front spar's maximum Z direction coordinate is not smaller than the rib's minimum Z direction coordinate, nor is the rib's maximum Z direction coordinate smaller than the front spar's minimum Z direction coordinate. More projections are needed. The surfaces in the overlap area that are not parallel with the Z-axis are then projected onto a XY-oriented plane far away from the wingbox. All of the front spar's projected surfaces that have different Z coordinates than the projected rib surfaces that overlap with them are first examined. All such surfaces for the spar have lower

Z coordinates than their overlapping rib surfaces. Then, all of the front spar's projected surfaces that have roughly the same Z coordinates as the projected rib surfaces that overlap with them are examined. Those spar surfaces have a roughly positive Z-axis direction-pointing normal vector while the respective overlapping rib surfaces have roughly negative Z-axis direction-pointing normal vectors. This means the rib interferes with the front spar in the positive Z direction while the front spar does not interfere with the rib at all. If the front spar is Part 1 and the rib is Part 5, this means that element e_{15} is 1 and element e_{51} is 0 in the +Z interference matrix. A visualization of this as well as an equivalent one for an exaggerated fuselage is shown in Figure 5.4.

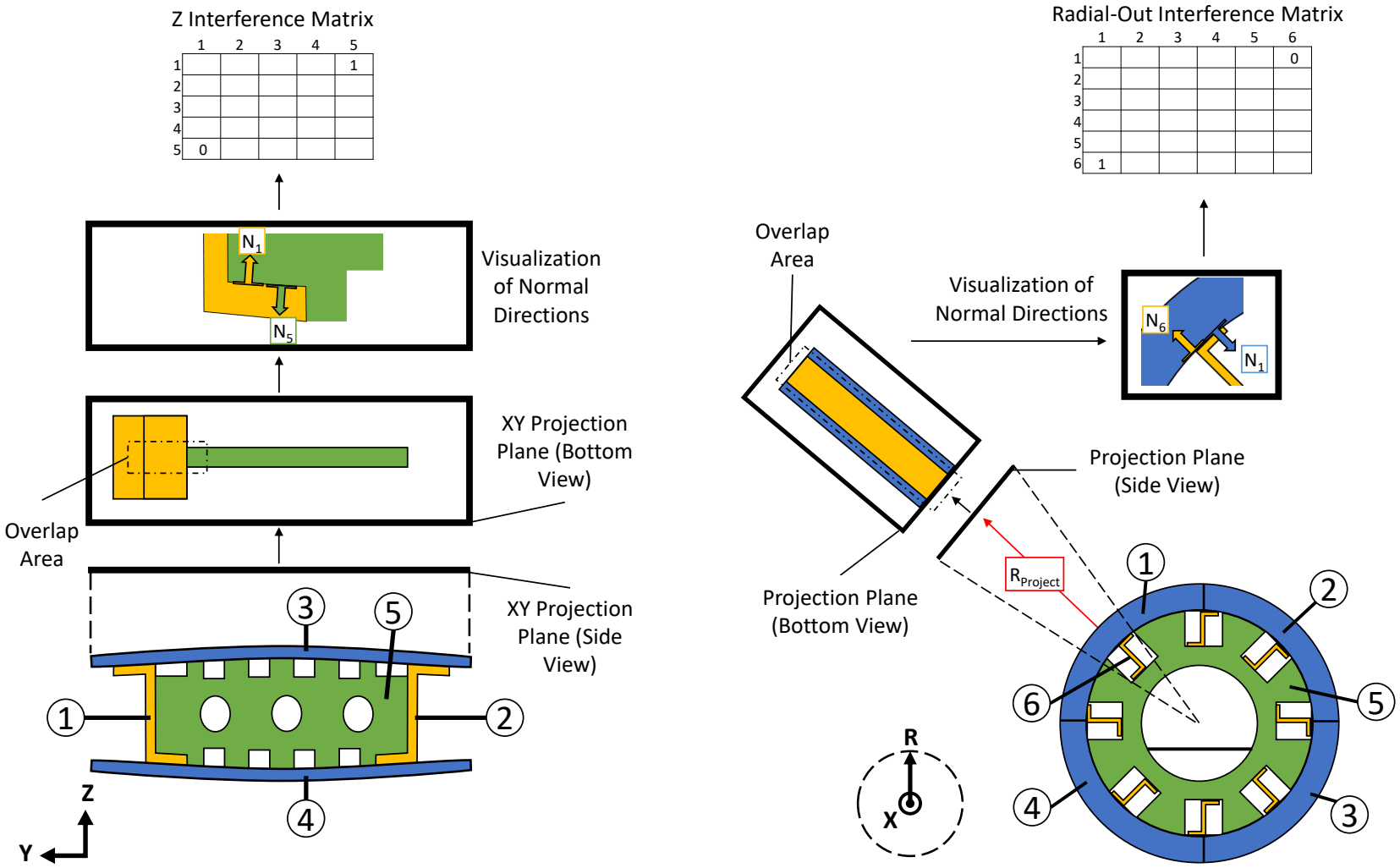


Figure 5.4: Example Interference Matrix Generation for Front Spar and a Rib, Parts 1 and 5, for Wingbox on Left, and for Fuselage Skin and Frame Stringer, Parts 1 and 6, for Fuselage On Right. Read from Bottom Up, Part Numbers in Circles, Fuselage is Highly Exaggerated for Easier Visualization

The If-Then precedence relation rules capture precedence constraints imposed by the manufacturing processes being used. These constraints generally lie outside of the part's geometry and so liaison graphs and interference matrices are unable to capture them. Examples of these rules include the following:

- IF wingbox is assembly being analyzed, THEN all minor parts must be assembled before their major parts can be assembled with other major parts
- IF aluminum CNC is manufacturing process used, THEN integral fabrication cannot be used
- IF stitched resin infusion is manufacturing process used AND IF major parts are integrally fabricated, THEN all minor parts associated with those major parts must also be integrally fabricated
- IF parts are integrally fabricated, THEN they must be integrally fabricated before they can be assembled with other major parts
- IF parts are part of subassemblies, THEN they must be preassembled into said subassemblies before they can be assembled with other major parts

The assembly sequence generation is carried out by first determining which parts will be integrally fabricated or become part of a subassembly. Only parts that share edges in the liaison graph can do either. Those parts are grouped together and must be fabricated or preassembled first, respectively. There are generally no restrictions on what is allowed to be the first part, or group of parts if they are integrally fabricated or part of a subassembly, beyond what is described in the If-Then precedence relation rules and what is feasible, which is determined later on. After the first part or part group is chosen, the next part or part group to be assembled is selected. All parts or part groups after the first must share a liaison graph edge with one of the parts or part groups already placed. This is to ensure coherence in the assembly sequence such that a part is always assembled and attached to an already

existing part or assembly. To not do so is to essentially attach the next part to nothing and just have it float there. As mentioned by Marian et al., this is technically possible given the usage of supportive tooling, but is generally not done in aircraft manufacturing and so is not preferable [141]. The assembly sequence generation is finished when no more liaison graph nodes are available to place additional parts, there are no more edges connecting the nodes of the already-placed parts to the ones not yet placed, or no parts can be further assembled due to interference. In the latter two cases, the generation must be done again with a different sequence and possibly with different integral fabrication or subassembly combinations since, while many combinations are valid, not all result in feasible sequences.

The assembly sequences in this work are represented as either a $3 \times n$ or as a $4 \times m$ part-based matrix, where n is the total number of parts and m is the total number of major parts. The first row in the matrix is for the actual ordering of the parts via their part numbers, e.g., if part 3 is the first part placed in the assembly and part 5 is the second, the first element in the first row is 3 and the second element is 5. The second row is used to indicate which parts are integrally fabricated, with integrally fabricated parts having the same nonzero positive number in the second row in their respective columns. The third row is likewise used to indicate which parts are in the same subassembly, with subassembly parts having the same nonzero positive number in the third row in their respective columns. Each set of integrally fabricated parts or subassembly has its own value in the second or third row to differentiate between them. Similarly, the values start at 1 and increment up and are distinct–the same value from the second row will not appear in the third row and vice versa. This is to quickly be able to determine how many sets of parts are integrally fabricated or part of subassemblies when looking at the sequence's matrix representation. Additionally, to not cause confusion, parts that belong to the same integral fabrication set or subassembly are placed adjacent to each other because they must be assembled or fabricated together. At least for the wingbox, there is currently no certified way to integrally fabricate the entire assembly at once. Similarly, there is no way to have the entire assembly be a subassembly

because there it is not possible to operate on every part simultaneously. At least one of them will need to be assembled at a different time than the others, causing that all-inclusive subassembly to itself become just a regular assembly. As such, for wingboxes, the number of parts that can be integrally fabricated or part of a subassembly is limited to being one less than the total number of major parts in the assembly.

The fourth row is specific to wingboxes only, meaning the $4 \times m$ matrix is also specific to wingboxes only. This is out of convenience because nigh all wingbox assemblies have a manufacturing precedence constraint that all minor parts be attached to their major parts before the major parts are assembled with each other. This is why the wingbox sequence matrix only has as many columns as major parts; minor parts can only be attached to their respective major parts and, if they must always be attached to their major part before the major part can be assembled to anything else, they become like fabrication tasks in that they are unable to interact with anything. Their place in the sequence is essentially almost fully defined by whenever their corresponding major parts are assembled, meaning they do not need to explicitly show up in a sequence. The only piece of information missing is whether they are integrally fabricated or not, which is the role of the fourth row. The fourth row indicates whether the minor part associated with its column's major part is integrally fabricated with it. Only integral fabrication is considered because for wingbox minor parts, unlike for major parts, there is no decision on whether it should be integrally fabricated or part of a subassembly–their default state is to be in a subassembly with their major part, meaning the only other option is integral fabrication. The only possible values are 0 or 1, indicating no or yes for integral fabrication. This means the only possible value is 0 for processes where integral fabrication is not possible for any parts. Fuselage assemblies, on the other hand, do not have as many constraints and so their matrix representation is the more generalized of the two. Figure 5.5 shows an example wingbox and fuselage assembly using this work's matrix representation.

After the assembly sequence has been created, its geometric feasibility is checked using

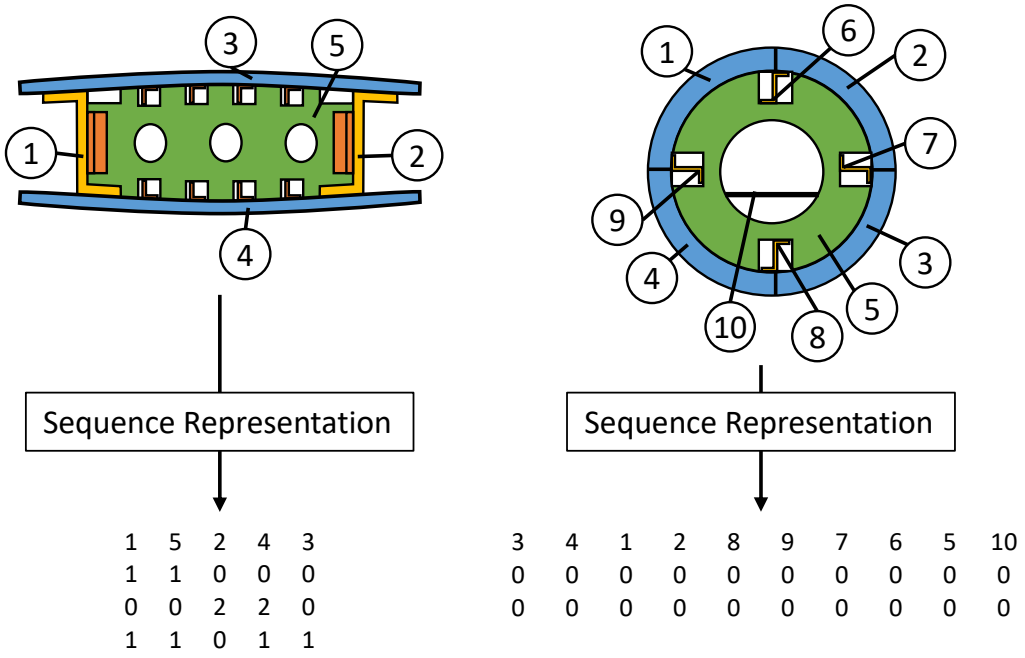


Figure 5.5: Example Sequences and Their Representations for Wingbox and Fuselage, Latter Exaggerated for Easier Visualization

the interference matrices previously generated. As mentioned in Conjecture 1.2, the level of abstraction being used for the geometric feasibility aspect of geometric reasoning treats the parts as free flying objects. This means the parts are treated as rigid, with tolerances and the effects of gravity and other forces are not considered. Assembly operations are thus reversible and allow the assembly geometric feasibility check to be performed in terms of disassembly instead, which is much simpler. To do this the assembly sequence must be read backwards starting with the part, or set of parts for integral fabrication and subassemblies, in the furthest right column(s) in the first row. According to the sequence, this part is the last one assembled or the first one disassembled. The part(s) is identified and the rows and columns corresponding to it in all the interference matrices determined. For the sequence to be feasible, at least one of the interference matrices must have a row or column corresponding to this part that only has zero values. This indicates that there is nothing obstructing and preventing this part from being translated in that matrix's positive direction

if such a row is found, or in the matrix's negative direction if such a column is found. Otherwise, the sequence is infeasible. If the sequence is still feasible, the rows and columns corresponding to that part in all the interference matrices are removed and the next part to be disassembled is identified. This process is repeated until there are no parts left to be disassembled or a part to be disassembled fails to have at least one interference matrix that has the part's corresponding row or column only contain zero values. The process works because if an assembly sequence yields a geometrically feasible disassembly sequence by reading the assembly sequence backwards then, due to assembly being reversible in this situation, it will mean the assembly defined by reading the assembly sequencing in the correct order is feasible as well. A note on this is that if a sequence is feasible by disassembling the part in a certain direction, it means it is feasible by assembling the part in the negative of that direction.

The mechanical feasibility check is somewhat more straightforward than the geometric feasibility one. As described in subsection 4.3.2, visibility checks based on determining the line of sight between joining sites and the accessibility locations will determine the accessibility of the assemblies. For the wingbox the accessibility locations are all outside the contour, meaning the goal is to see if there is a closed contour around the joining sites and, if so, how many joining sites are affected and require access holes. This involves counting the number of joining locations that can be operated on before the last part or set of parts is assembled, counting the number of new joining locations in the last step that can be operated on, and then subtracting both of those values from the total number of joining locations. Joining locations can only be operated on if their necessary parts have been attached and both sides can make line of sight with an accessibility location. The resultant value is the number of joining locations that require access holes and can be used as a relative accessibility metric. The procedure with the fuselage is less defined because its accessibility is based on how many joining locations require the interior accessibility locations after the fuselage's contour has been sufficiently closed off. The point at which

this happens has significant variability due to it depending on the equipment and tooling used, the size of the fuselage, and even the manufacturing process. As a result, only a general procedure can be provided. The general procedure is that joining locations able to be operated on when the fuselage contour is sufficiently open are counted as belonging to the "no-penalty" accessibility group. Those that are operated on when the fuselage contour is sufficiently closed are counted as belonging to the "penalty" accessibility group. When all parts are assembled, the number of how many joining locations "belong" to either accessibility group is added up and used to create a ratio between the two, which determines the relative accessibility. A note in all this as well is that the joining locations between all parts that are integrally fabricated together are eliminated.

To summarize, liaison graphs are made by examining which parts are in contact and interference matrices made using a projection-based method. These capture the geometric description of the assembly while If-Then precedence rules are used to capture the manufacturing process constraints. Whether parts will be integrally fabricated or part of a subassembly is then determined, at which point all this information is used to generate the assembly sequences. These are checked for geometric feasibility by comparing them with the interference matrices and their accessibility and mechanical feasibility are afterwards checked with a visibility-based method.

5.4.2 Assembly Line Balancing Implementation Details

After the assembly sequence or sequences have been generated, rule-based modeling is used to convert them into precedence relations that can be used to create assembly line balances. The precedence relations describe the order that the different manufacturing tasks must follow when being assigned to stations and include fabrication and assembly tasks. The list of these tasks is specific to every manufacturing process, with there typically being greater differences between fabrication tasks as opposed to assembly tasks. There is generally a separate instance of every fabrication task for each part while the number of

assembly tasks is more dependent on the number of times a part is attached to another part, which is determined by the assembly sequence. As an example, a generic composite hand layup autoclave manufacturing process has four different types of fabrication tasks, making a fabrication task set, and three different types of assembly tasks, making an assembly task set. The fabrication task set is composed of the hand layup, autoclave cure, part trim, and non-destructive inspection tasks while the assembly task set is composed of the parts fitup, drill parts, and fasten parts tasks. If there are five parts and the assembly sequence yields a sequence of four part-to-part attachments, then this means the total task list consists of five fabrication task sets and four assembly task sets. This results in twenty distinct fabrication tasks and twelve distinct assembly tasks in the list of available tasks. The subsequent precedence relations for line balancing are used to describe the order these tasks can be assigned to stations in, relative to each other. Some rules for the rule-based modeling for the example generic hand layup autoclave process are listed below.

- IF the hand layup autoclave process is used, THEN the fabrication task ordering is hand layup, autoclave cure, part trim, and non-destructive inspection and the assembly task ordering is parts fitup, drill parts, and fasten parts
- IF minor parts are integrally fabricated with a major part, THEN their fitup task with their major part must occur before the major part's autoclave cure task
- IF minor parts are integrally fabricated with a major part, THEN their autoclave cure, part trim, and non-destructive inspection fabrication tasks along with their drill and fasten minor part-to-major part assembly tasks are removed from the list of available tasks
- IF major parts are part of a subassembly, THEN all but one of their assembly task sets are removed from the list of available tasks
- IF minor parts are not integrally fabricated with their major parts, THEN their fasten

parts task where they are fastened to their major parts must occur before any of their major parts' major part-to-major part assembly tasks

The simplest structure to use to represent precedence relations for ALB is the precedence graph. Precedence graphs were shown to be unable to, by themselves, capture all the complex relations involved with integral fabrication as well as constantly-changing assembly sequences. However, a precedence graph dynamically derived from the assembly sequence being used and from the rules in the rule-based modeling would be able to capture the relations for that particular sequence and that particular process. In other words, explicitly using a precedence graph to try to capture all the complex constraints and precedence relations is infeasible due to requiring numerous instances of them, as stated by Topaloglu et al. [207]; but if only a small subset of those constraints and relations are of interest, such as when the assembly sequence, manufacturing process, and integrally fabricated parts are already selected, it is perfectly possible to represent them with a single precedence graph. For this reason, the rule-based modeling outputs a precedence graph to store all the task-based precedence relations derived from assembly sequence planning. The main limitation is that, as this precedence graph is dynamically derived, it is unique to the selected assembly sequence, manufacturing process, geometry, and integrally fabricated parts. If any of those variables are changed, a new graph is needed.

The mathematical implementation of a precedence graph is a precedence matrix, which is also easier to operate on and visualize given the possibly immense number of tasks and nodes involved and the complexity of the numerous edges and relations connecting them. Precedence matrices are thus the primary form to communicate precedence graph information in this work. Precedence matrices were covered back in subsection 3.1.1. They are $t \times t$ in size, with t being the total number of tasks. Element e_{ij} is 1 if task t_j requires task t_i to be done before task t_j can be carried out. Element e_{ij} is 0 otherwise. The precedence matrix is therefore essentially read column-wise; if a task's column only has 0s, then it can be assigned to a station at any time. If that column has any 1s, the tasks represented by the

rows with those 1s must be carried out first.

After the rule-based modeling has finished translating the ASP information into task-based information and constraints and stored them in precedence matrices, the amount of time it takes to perform those tasks and the associated costs are determined next. This is done with parametric cost and time estimation methods and tools. The available types of time outputs vary based on the method or tool used. The effects of the learning curve on task times will not be included in this work. This is because it causes large fluctuations in the task times which can render line balances optimized for later on in the production run to be infeasible early on in the production run. To try to use an average task time that accounts for learning curves is to cause the line balances to still be infeasible early on in the production run and insufficiently optimized later on as well, which is even less desirable. Therefore, learning effects will not be examined to remove the issue altogether in this work. They can be added on in future works when the focus is less on developing the first iteration able to tackle this integrated problem and more on optimizing the approach to tackle this integrated problem.

Given the nature of the integrated problem being tackled, the cost consists of the following: labor cost, equipment cost, tool cost, and material cost. Labor costs are fundamental to the work performed by the actual workers and so must be included. Similarly, equipment and tooling costs are needed to account for the differences in the types of capital resources needed for different manufacturing processes. Material costs are included to help account for different manufacturing process's efficiency and material needs: different processes can use more or less material, which changes how much cost is accrued, as well as different materials altogether, some of which also cost more than others.

A notable cost not included is the cost of space. Space costs are not included because, in real life, factories of this scale are primarily, only, and already under the ownership of aircraft manufacturers. It is impractical to obtain new factories of other sizes, particularly even larger ones, leading to the constraint that the only factory sizes considered are the ones

that already exist and owned by the manufacturer. This means the space costs are the same for all designs given the same spatial constraints. As these space costs do not differentiate designs, they do not need to be considered while analyzing the designs. Similarly, because they are already owned, they also do not cost anything. For these reasons, space costs will not be accounted for.

After these costs for the tasks as well as their task times have been estimated, the setup required for assembly line balancing is completed and the actual line balancing can be performed.

Any task can be assigned to a workstation as long as the constraints in the previously defined precedence matrix are followed and zoning constraints adhered to. The zoning constraints exist because all the tasks need a specific equipment and tooling type to be carried out and only one equipment and tooling type can be placed at each station. This causes some tasks to not be compatible with others. For example, for an automated tape layup process, the tape layup task requires an automated tape layup machine. This means it cannot be placed in the same station as an autoclave cure task that needs an autoclave due to the autoclave not being allowed to be in the same station as a layup machine because they are different equipment types. However, it is possible for a spar tape layup task to be placed in the same station as a wingskin tape layup task given a sufficiently flexible automated tape layup machine because both the equipment and tooling types are the same. Conversely, a hand layup task that does not require any dedicated equipment still cannot be placed in the same station as a manual assembly fitup task because they require different types of tooling; the hand layup task requires hand layup tooling while the assembly fitup requires assembly tooling. This means no fabrication tasks can be placed in the same station as an assembly task from their always having different tooling types. In general, the zoning constraints require that tasks can only be placed in stations with other tasks that share both the same equipment and tooling type.

Beyond following the precedence matrix relations and zoning constraints, there are no

other constraints to assigning tasks to stations. Not even cycle time constraints are applied because they will be addressed via station paralleling later on. After it has been decided that the station has a sufficient number of tasks, it is closed and no more tasks are assigned to it. At this point, the station time and space requirement for that station are calculated. The station time is the sum of all the task times of that station's tasks. The space requirement is calculated in a more complex way, which requires that its constituent parts be explained first.

In this work, the space requirements for a station are composed of space for the equipment A_{Equip}, working space $A_{Working}$, loading space A_{Load}, space for part queues A_{Queue}, and aisle space A_{Aisle}. This is represented by the following:

$$A_{ind} = A_{Equip} + A_{Working} + A_{Load} + A_{Queue} + A_{Aisle} \tag{5.1}$$

$$A = \sum_{m=1}^{TS} n_m * A_{ind_m} \tag{5.2}$$

A is total space required for all stations, TS is the total number of stations, n is the number of instances of each station, and A_{ind} is the space required for an individual station.

Equipment space is the space occupied by the equipment necessary to perform the task. Loading space is the space occupied by the part itself before it is placed into or onto the equipment. Parts generally cannot be transferred directly from one piece of equipment onto another, they must be placed in an intermediate loading location first. Similarly, there must be space for them to be unloaded after their task is finished. Manufacturing equipment can typically only work on one part at a time and so, when multiple tasks and thus multiple parts are assigned to the same station, the parts and tasks must be individually and sequentially processed. The parts not actively being worked on will need temporary storage space as they wait for their task to be processed, which is represented by part queue space. Working space is similar to equipment space in that it is the space needed for the workers to move

about and perform the tasks. It can also be used to store extra tools needed to operate the equipment or move the tooling. And finally, aisle space is space dedicated to making aisleways for parts to be able to move in between stations. Tooling space is notably not included because tooling is usually what the parts sit on and so generally moves with and is accounted with the parts.

All of this space being accounted for enables for a realistic representation of the amount of space needed to manufacture and work on parts and their tasks and also enables many different tradeoffs. The inclusion of equipment space means that large pieces of equipment that yield low task times may not high production rates due to taking up too much room and not being able to be duplicated enough times. Similarly, including part queue space discourages putting all tasks that require the same equipment and tooling types into one station because, while the equipment cost may be minimized, most of the space is essentially being used as storage and does not add any value, lowering throughput. Conversely, including aisle space, on top of adding extra realism, discourages stations with only a single task in them because it causes the aisle space to take up a proportionally larger amount of the station's space, which also does not add value and can lower throughput. Including these different types of space makes sure that, when the designs are examined more carefully or implemented, a sudden realization that there isn't enough space for the workers to walk or load the parts does not occur–reality is better capture. The additional benefit of being able to make these tradeoffs better captures similar tradeoffs that must occur in real life as well in a factory with limited space.

When all the tasks have been assigned to stations and all their space requirements determined, the total space needed is tallied up. If the space needed is larger than what the factory is able to provide, the design is infeasible and the process started over. If the space needed is less than what the factory is able to provide, then the next step is to attempt to find the maximum possible production rate. To do this, all the station times are compared to identify which one is the largest and, therefore, sets the cycle time and acts as the bottle-

neck. The number of instances of that station is virtually incremented by 1 to see if there is enough space for it to be incremented. If there is not, then the maximum station time becomes the cycle time and the production rate is determined. The production rate is equal to the inverse of the cycle time and is represented by the following equation:

$$T = 1/c \tag{5.3}$$

T is the production rate in products per month while c is the cycle time in months. Cycle times are usually represented in minutes, so to convert it to months the following conversion is used: 60 minutes per hour, 8 hours per shift, 2 shifts per day, 5 days a week, 4.33 weeks per month. This conversion is representative of what the largest manufacturers, Boeing and Airbus, currently use. This carries with it one of the main assumptions of this work, which is that the production is in a steady state. This allows station duplication to actually increase production rate as much as the equations suggest it would.

If there is enough space to virtually increment the number of instances of the bottleneck station by 1, then the increment is made real and no longer virtual. The former bottleneck station's effective station time becomes its original station time divided by its number of instances. Afterwards, the station with the next bottleneck station in the form of the one with the next largest effective station time is identified. The same process is repeated where its number of instances is virtually incremented by 1 to see if there is enough space: if there isn't then that effective station time becomes the cycle time, and if there is then the virtual increment becomes a real increment and that station's new effective station time is determined. This repeats until there is no space left, at which point the largest effective station time becomes the cycle time and determines the production rate. This is described by the following equations and Figure 5.6.

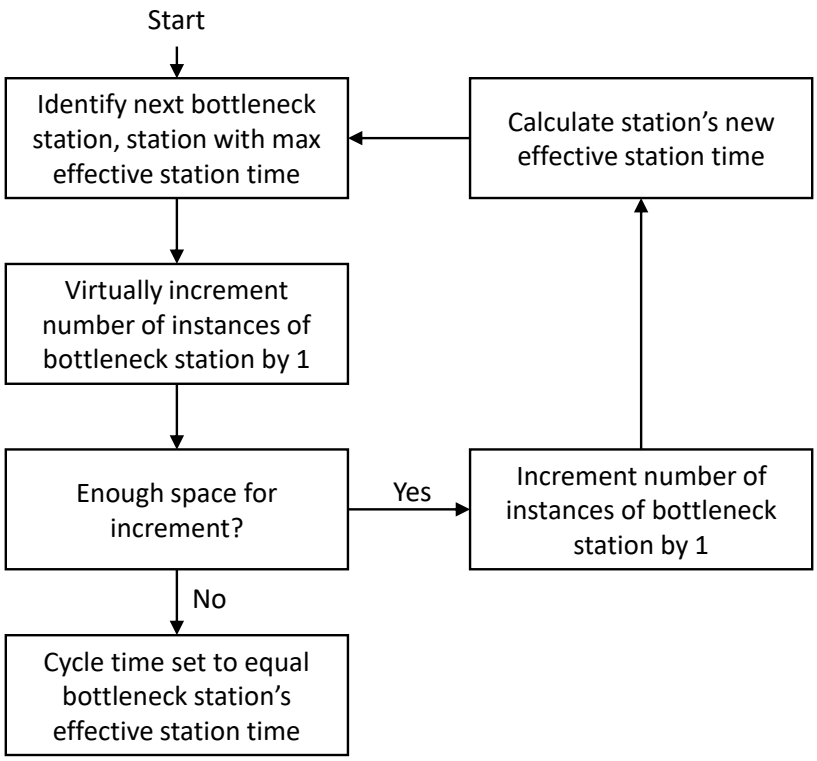


Figure 5.6: Steps to Determine Max Production Rate

$$c = \max_{1 \leq i \leq TS} ST_{eff_i} \tag{5.4}$$

$$ST_{eff} = \frac{\sum_{d=1}^{V} t_d}{n} \tag{5.5}$$

TS is the total number of stations, ST_{eff} is the effective station time, V is the total number of tasks in a station, n is the number of instances of a station, and t is task time.

Once enough designs have been iterated through that there is confidence that a specific value is close to or is indeed the maximum physically feasible production rate for the current parameters, the next step is to generate the rest of the Pareto frontier. Pareto frontiers were chosen as the visualization tool of choice to represent multi-objective solutions for this multi-objective, integrated problem for a few reasons. One is that in the majority of multi-objective line balancing works, all the different results are presented as alternative

solutions in a Pareto frontier or stored as non-dominated solutions and so effectively create a Pareto frontier, making them the most common and easiest way to communicate multi-objective results [142, 75, 76, 172, 166, 154]. And the other is that they give the designer the freedom to see and compare what solutions are available and to select the ones they deem best [142, 154].

Assuming that the x-axis in this two-objective multi-objective problem is the production rate and that the y-axis is the cost, the top rightmost corner of the Pareto front was found when the maximum physically feasible production rate was identified. The ε-constraint method is used to fill out the rest of the Pareto frontier. This is done by selecting all the other throughput levels of interest to eventually set as ε-constraints and then solving the single-objective subproblem at each constraint. Because the upper limit on production rate was already found and because it was obtained by maximizing space usage, it is guaranteed that there is at least one solution at every one of the lower-throughput constraints that is feasible. Essentially, after the rightmost edge of the Pareto front is established, the ε-constraint method is used to sweep through all of the lower throughputs and obtain non-dominated designs at each constraint level by optimizing for cost, incrementally and iteratively building the Pareto frontier until the throughput constraint can be made no lower. This is visualized in Figure 5.7.

The specific cost objective to be solved for is the Average Per Unit Cost (APUC). This is chosen over a generic average cost because equipment and tooling costs are very high and so typically amortized over the entire production run. In this case it is amortized over the number of products outputted in the production run. It is represented as:

$$APUC = C_{total}/N \tag{5.6}$$

C_{total} is the total cost over the entire production run and N is the number of products or units built over the entire production run. The constituents of the total cost, as mentioned earlier, are the labor cost C_{labor}, equipment cost C_{eq}, tooling cost C_{tool}, and material cost

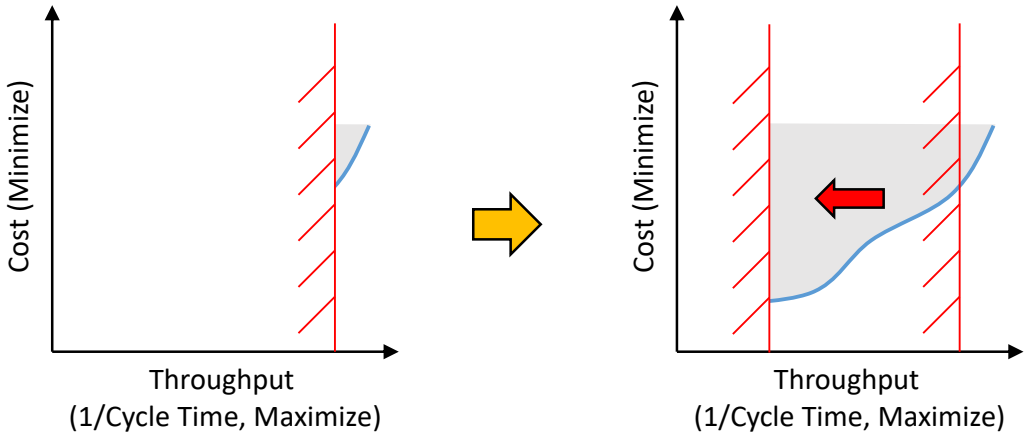


Figure 5.7: Visualization of How ε-Constraint Method Is Used to Build Up Pareto Front

C_{mat}. They are related to the total cost in the following way:

$$C_{total} = C_{labor} + C_{eq} + C_{tool} + C_{mat} \tag{5.7}$$

Workers get paid for being at their workstations even if there is idle time. Additionally, according to Amen, the wage rate at each workstation is conservatively equal to the most difficult, or costliest, task to perform at that station [219]. The labor cost is thus calculated as the product of the total time spent in manufacturing and the sum of the highest wage rate ω_k in each workstation for all m unique workstations, their duplicates n, and the number of workers per station z. The total time spent in manufacturing is the product of the number of units built N and the cycle time c. The total labor cost is shown as follows:

$$C_{labor} = N * c * \sum_{k=1}^{m} n_k z_k \omega_k \tag{5.8}$$

The equipment cost varies from equipment set to equipment set. Additionally, for aircraft manufacturing, the cost of equipment is seldom completely static and tends to be scalable based on some significant geometric parameter, which is usually related to the equipment's length or the area occupied by the equipment. As an example, for an auto-

clave, the autoclave's diameter is a significant contributor to its cost while its length is only a secondary driver of cost. Despite the equipment cost being scalable, the exact scalability relationship is different for every set of equipment and so only a general form is given below, where n is the number of instances of a station, m is the total number of unique stations, and C_{meq} is the equipment cost at each unique station. It is assumed that the equipment does not need to be replaced during the production run.

$$C_{eq} = \sum_{i=1}^{m} n_i C_{meq_i} \tag{5.9}$$

The cost of the tools and tooling is simply the sum of the cost of all the individual tooling needed. It is assumed that they do not need to be replaced and that a given set is used at each station. The cost of all the tooling is the sum of the tooling cost at each unique station C_{mtool} multiplied by the number of station instances n for all the unique stations m and is as follows:

$$C_{tool} = \sum_{i=1}^{m} n_i C_{mtool_i} \tag{5.10}$$

The material cost is not just the processing weight of all the materials at each fabrication station but is also the cost of all the fasteners and similar disposable materials at all the stations as well. The material cost is thus just the material cost incurred for each task C_{tmat} for all tasks B multiplied by the total number of products produced N:

$$C_{mat} = N * \sum_{k=1}^{B} C_{tmat_k} \tag{5.11}$$

The single-objective, APUC minimization subproblem is solved in much the same way as the production rate maximization problem with several key differences. The station duplication is stopped as soon as the bottleneck station's effective station time is under the target cycle time, essentially when the production rate exceeds the set throughput ε-constraint. And if a design cannot get its bottleneck station's effective station time under

the target cycle time, then it is infeasible. This is shown in Figure 5.8. The process is repeated for a sufficient number of designs until the minimum APUC for the throughput ε-constraint has been achieved.

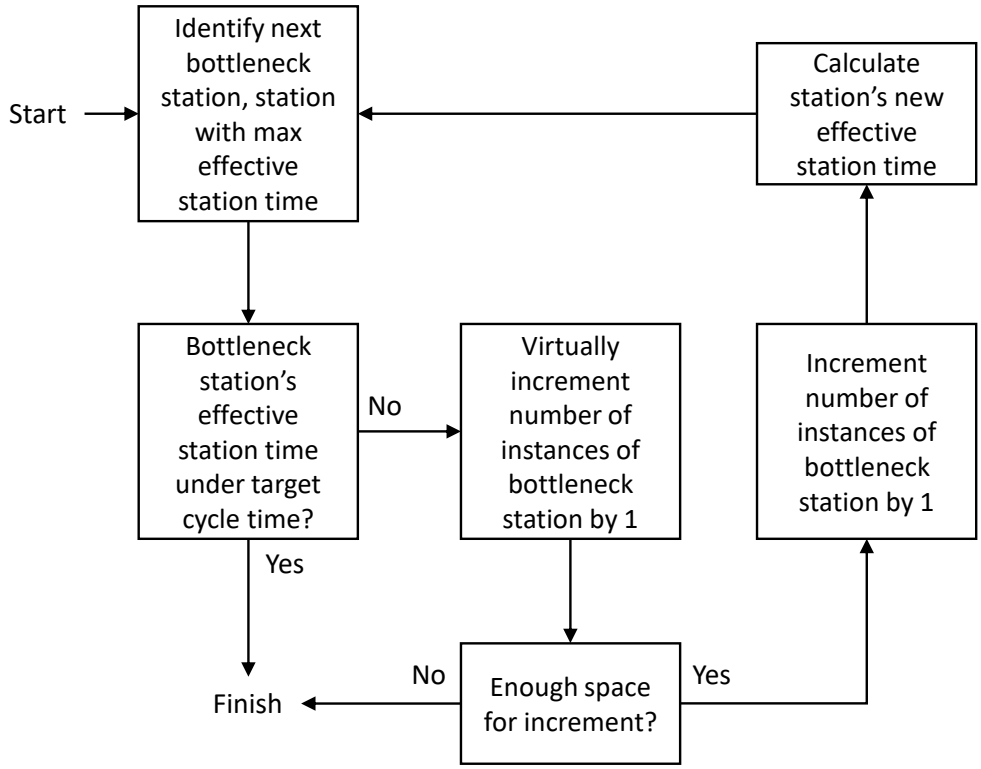


Figure 5.8: Station Duplication Steps Given Throughput ε-Constraint

Although the APUC is being optimized for at each throughput constraint and there is a guarantee of a feasible solution, it will not always be necessarily the case that the optimized solution at that constraint level is a non-dominated solution. This is because while lower throughputs tend to decrease costs due to requiring fewer station duplications, this is offset at some point by the fact that the cycle time will be large enough for there to be large idle times. Idle time is basically the cost of inefficiency and can eclipse the cost savings of having fewer station duplications and therefore fewer sets of equipment and tooling. Special attention must thus be made to check that the minimum cost designs at each throughput constraint level are actually non-dominated instead of just assuming they are and blindly adding them to the Pareto front.

The Pareto front and the multi-objective optimization is finished once all the throughput levels and the resultant subproblems have been swept through and solved and the non-dominated solutions identified.

Although the overall procedure of how the integrated problem is approached has been described, how to actually iterate and sift through the designs to find the solutions has not been discussed. The ε-constraint method is used to break up the larger multi-objective problem into a series of smaller and simpler subproblems. But it does not describe how to actually solve the subproblems. The next section goes over the actual steps used to do this, essentially the wrapper around this entire framework and how solutions are obtained.

5.4.3 Genetic Algorithm Implementation Details

According to Jimenez, who surveyed a large number of ASP works, more advanced methods such as meta-heuristics are preferred when there are a large number of parts or the ASP is very complex [80]. As such, given the size and complexity of the integrated problem of interest, meta-heuristic methods will be used over more traditional, exact ones such as graph-based methods. Based on a survey of a variety of survey papers, the most popular meta-heuristic techniques for ASP and ALB are GAs and ACO algorithms [76, 81, 97]. Of these, genetic algorithms will be the one used to solve the integrated multi-objective ASP and ALB problem of interest. This is for several reasons. GAs are more commonly used and well-studied for ALBPs in the ALB literature than ACO algorithms, meaning there are more references on how to implement complex problems with them. And as Abdullah et al. pointed out, one of their main benefits is the simplicity of their implementation, which is needed for as complex of a problem as the one being tackled [81]. Conversely, there are essentially no works in the ALBP literature that describe how to implement ACO algorithms whenever the station paralleling problem is involved, making it unclear on how to do so or if it is even possible. Another major reason is that it is far simpler to implement a GA to address the maximum production rate subproblem than to try to implement an ACO

one. All works in the ALB literature that use ACO require and have a method to close stations off and start assigning tasks in a new station. This is because the representation schemes for different designs in ACO do not inherently store task-to-station allocations, they require an outside rule or heuristic to translate the string of tasks into allocations. But as described in subsection 5.2.3, none of the normal station-closing rules or heuristics apply when cycle times are not static, the number of stations is not set, and station duplication is allowed, meaning there is no clear way to efficiently close off a station while trying to find the maximum production rate. Meanwhile, the general representation scheme in GAs easily accommodates essentially any arbitrary closing of stations, so GAs do not have this problem. Because of these reasons, a GA is used to solve the integrated problem.

Genetic algorithms are defined by how the designs are encoded, how the initial population is generated, how the parents for crossover are chosen, how the crossover itself operates, what mutation schemes are used, and how the replacement occurs to seed the individuals for the next optimization cycle. The way in which these are all implemented for this work will be covered next.

Before how the designs are encoded is explained, what the designs are composed of must first be determined. To fully define a design, all of its independent design variables that, together, are able to fully describe it need to be identified. Given the current framework, said independent variables are: the manufacturing process used, the categorical geometry variables, the spatial constraint being examined, the assembly sequence and line balance being analyzed, and the number of instances of each station in the line balance. Categorical geometry here is defined as any geometry variables that have a large impact on precedence relations such as the shape of the spar, rib, wingskin, or fuselage frame. The manufacturing process, categorical geometry variables, and spatial constraint are all categorical in the sense that changing them while the optimization is ongoing is akin to running multiple optimization instances simultaneously. This is a result of designs with different processes, categorical geometries, or space constraints not being able to perform crossover

with designs with different such variables. Designs with different such categorical variables have sufficiently different constraints that the crossover of those designs is akin to generating new solutions from scratch with few to no genes passed down, defeating the point of a GA. Thus, trying to include those categorical variables all in the same instance yields the same results as optimizing them in separate ones except that the former is significantly more complex due to having to deal with constraint compatibility issues. Because of this, those categorical variables are overarching variables that will not change during the optimization process and so do not need to be encoded or operated on, only stored. The number of instances of each station is a variable in that it independently determines the throughput and cost of the solution given a line balance. However, it is a variable that is generated during the ε-constraint process and is not independently set by the designer or operated on by the GA. As such, it also will not be encoded, only stored. Based on all this, only the assembly sequence and line balance need to be encoded. As this chapter is focused on the assembly side of trying to integrate the aircraft design and assembly disciplines, continuous aircraft design and geometry variables are not considered until Chapter 7.

All of the other information can be derived from the above identified information. The geometric and precedence constraints, task times, task costs, and final throughput and APUC can be obtained given any a combination of the process, sequence, and geometry variables. As such, those pieces of information are all dependent, intermediate parameters that do not need to be stored or encoded because they can be actively generated and regenerated.

Because only the assembly sequence and line balance variables are independent, actively operated on, and need to be encoded, two chromosomes, one for each, are required. The assembly sequence chromosome is exactly the same as the representation shown in Figure 5.5 and so is not discussed further. The line balance chromosome is a $2 \times \lambda$ matrix where the first row is an ordered list of task numbers, representing their respective tasks,

and the second row is the stations those tasks are assigned to. λ is the number of available tasks in the task list. Every manufacturing process has a total of Λ possible tasks, but depending on the sequence and parts chosen to be integrally fabricated or in subassemblies, some of those tasks will not be applicable and are removed from the list of available tasks. An example wingbox assembly sequence chromosome and line balance chromosome is shown in Figure 5.9. The assembly sequence and line balance are not correlated with each other in that example.

Sequence Chromosome

Part Number	1	5	2	4	3
Major Integral Fabrication	1	1	0	0	0
Major Subassembly	0	0	2	2	0
Minor Integral Fabrication	1	1	0	1	1

Line Balance Chromosome

Task Number	1	5	2	3	6	7	4	8	9	10	11
Station Number	1	1	2	3	4	5	6	6	7	8	9

Figure 5.9: Example Assembly Sequence and Line Balance Chromosome Encoding

How both sets of chromosomes are generated for the initial population is covered next, starting with the assembly sequence chromosome. One of the several weaknesses of using GAs for ASP is that they have a difficult time dealing with infeasible sequences that break either the geometric or precedence constraints. These sequences can come from either the initial population generation or from the crossover process. Many works handle this by using repair operators to alter the sequences until they follow the constraints, which can be a time consuming and complicated task. To avoid this, it is preferable to always generate feasible sequences in the first place and to always have the results of crossover be feasible sequences so that repair operators are never needed. Marian et al. have an effective way to do this through their guided search procedure [141]. Because the structure used to store ASP constraints for this work does not look exactly like Marian et al.'s, changes have been

made to their guided search procedure to accommodate for this.

Although assembly sequences are read from left to right and subsequently generally generated from left to right, geometric feasibility checks using interference matrices are significantly easier to do when assembly by disassembly assumptions are used, which is done right to left. As a result, the assembly sequence chromosome for this work is generated from right to left, which is in reverse order to how it is generally done. First, integrally fabricated parts or parts that are to be in subassemblies are identified and the respective nodes in the liaison graph combined and the edges regenerated to account for this. All the parts or sets of parts at this point are added to the possible candidate list.

Then, a part, or set of parts if it is integrally fabricated or in a subassembly, is randomly chosen from the possible candidate list and checked for several features, which if it meets will allow it to be moved to the candidate list. The first feature is that if the part or set of parts and its node are removed from the liaison graph, there are no nodes or group of nodes separated from the rest of the graph due to a lack of edges. This is to ensure coherence because if a part or set of parts is removed and does create loner groups in the liaison graph, it indicates that that part or set of parts is connecting the other part(s) and so must be assembled earlier or chosen to be a candidate later on. The second feature is that if the chosen part or set of parts is involved in an If-Then precedence rule or rules where one part must be assembled before the other, the chosen part or set of parts is the one that must be assembled last in those rule(s) relative to all the parts in the possible candidate list. This is because this altered guided search procedure operates from right to left, meaning parts or sets of parts that are to be disassembled first are added to the sequence chromosome first. And in such a precedence rule involving part order, it means the part assembled last should be added first to the chromosome. The third feature is that the part or set of parts chosen must pass a special interference check. Temporary interference matrices are made by taking the full ones and removing all rows and columns not related to the chosen part(s) and the parts in the possible candidate list. If the chosen part(s) do not have any non-

zero values in just one of any of their associated rows or columns in any of the temporary matrices, they pass the interference check. How this works is that if the chosen part(s) pass this interference check, it means they can be disassembled even if all the other parts in the possible candidate list have already been assembled. Due to assembly and disassembly being reversible, it equivalently means the chosen part(s) can be feasibly assembled given said assembly state regardless of the order the possible candidate list parts are assembled in. And because the assemblies being analyzed are sequential and monotone in nature, part(s) that can be feasibly (dis)assembled in this way are guaranteed to not interfere with the (dis)assembly of parts earlier in the assembly sequence, or further to the left in the sequence representation/chromosome. If the chosen part(s) possess all these features, it is added to the candidate list.

The above process is repeated until all parts in the possible candidate list have been checked to see if they can be added to the candidate list. Once this is done, a random part or set of parts is selected from the candidate list and placed at the rightmost end of the sequence chromosome but left of any preexisting, already-placed part(s). This part or set of parts is removed from the possible candidate list, its corresponding node is removed from the liaison graph, the candidate list is cleared, and the entire process is repeated until the possible candidate list is empty. In situations where the possible candidate list is not empty but no parts can be added to the candidate list, the entire process is started over to generate a new sequence.

The procedure is summed up in the following list, **List A**:

Step A1: Instantiate liaison graph, interference matrices, If-Then precedence rules

Step A2: Select parts to be integrally fabricated or part of subassemblies if desired

Step A3: Update liaison graph and interference matrices to reflect selection

Step A4: Add all parts to possible candidate list, initiate candidate list

Step A5: Randomly choose part, or set of parts if integrally fabricated or in subassembly, from possible candidate list not currently in candidate list and not yet checked to be in candidate list

Step A6: Check if removal of chosen part(s) from liaison graph results in loner nodes or groups of nodes not connected to each other

Step A7: If chosen part(s) are part of precedence rules related to ordering, check if they are the part(s) that must be placed after all the other parts in the possible candidate list

Step A8: Create temporary interference matrices by truncating interference matrices from Step A3 to only include parts in possible candidate list. If there are no non-zero values in any of chosen part(s)'s associated rows or columns in any of the interference matrices, chosen part(s) pass special interference check.

Step A9: If chosen part(s) do not fail checks in Steps A6-A8, add it to candidate list

Step A10: Repeat Steps A5-A9 until all parts in possible candidate list have been checked to see if they can be in candidate list

Step A11: Randomly select part(s) in candidate list, add it to rightmost end of assembly sequence chromosome but left of already-added parts, if any

Step A12: Remove selected part(s) from possible candidate list, remove its node from liaison graph, clear candidate list

Step A13: Repeat Steps A5-A12 until no more parts in possible candidate list. If no parts can be added from possible candidate list to candidate list and possible candidate list is not empty, start over from Step A2

This procedure is repeated as many times as necessary to generate the desired number of individuals in the initial population. The basis of the procedure on Marian et al.'s

guided search minimizes the number of times that the procedure must be repeated without generating a chromosome by maximizing the odds that the sequence is feasible.

A similar concept is used to generate the initial line balance chromosomes where only feasible tasks are considered for task-to-station allocation. First, the precedence matrix is generated and checked such that all of the relations stored in it are correct given the manufacturing process, integral parts, subassemblies, geometry, and assembly sequence chosen based on the ASP-ALB translation precedence rules. Then, all unavailable tasks as determined by those same rules are removed. From there the list of candidate tasks is made by searching the precedence matrix for all tasks whose columns consist only of zeroes. One of these is randomly selected and assigned to a station whose station number is 1, indicating it is the first station. The selected task's corresponding row and column is afterwards removed from the precedence matrix. The list of candidate tasks is regenerated by again searching the precedence matrix for all tasks whose columns consist only of zeroes and another task randomly selected. If zoning constraints force the selected task to be in a different station from the previous task, the task is placed in a new station, indicated by incrementing the station number up 1. If zoning constraints, or lack thereof, allow the two tasks to be in the same station, it is randomly decided whether the selected task should be in the same station as the previous one or in a new station. Should the former happen, the selected task is assigned to a station whose station number is the same as the previous one's. Should the latter happen, it is assigned to a station whose number is incremented up 1. The process is repeated until all available tasks have been assigned to a station. The procedure is summarized as follows in **List B**:

Step B1: Generate precedence matrix that conforms to all manufacturing process, integral parts, subassemblies, geometry, and assembly sequence constraints

Step B2: Remove all unavailable tasks from precedence matrix

Step B3: Use precedence matrix to identify and add all feasible tasks to candidate list

Step B4: Randomly select task to assign to station. If task is first task, station number is 1

Step B5: If task is not first task, check zoning constraints with previous task. If selected task cannot be in same station as previous task, selected task's station number is incremented by 1. If selected task is able to be in same station as previous task, randomly determine if selected task will be in same station. If yes, selected task's station number is same as previous task's. If not, increment selected task's station number by 1

Step B6: Update precedence matrix by removing selected task's associated row and column, clear candidate list

Step B7: Repeat Steps B3-B6 until no more tasks left

This is done until as many line balance chromosomes are generated as assembly sequence chromosomes.

To select the individuals, or parents, for the potential crossover, the binary tournament selection operator is used. Two individuals from the population are selected and compared, with the better one becoming the first parent, and two more individuals are selected from the population and compared, with the better of those two becoming the second parent. The definition of which individual is better depends on which subproblem is being solved. If the maximum physically feasible production rate subproblem is being solved, the individual with the higher production rate or the lower cycle time becomes the parent. If the minimum APUC subproblem is being solved, the throughputs of both individuals are checked first. The throughputs must be equal to or higher than the current throughput ε-constraint. If one individual has a throughput lower than the ε-constraint but also a lower APUC than the other individual, the other individual is still preferred and becomes the parent. If both individuals have throughputs higher than the ε-constraint, then the one with the lower APUC becomes the parent. If neither individuals' throughputs are higher than the throughput ε-constraint, then the one with the higher throughput becomes the parent.

This emphasis on throughput even in the APUC subproblem is because the point of the ε-constraint method in this work is to generate solutions with a minimum throughput and try to minimize APUC among those solutions to sweep through the Pareto front. This means there is no point in trying to minimize APUC if the minimum throughput cannot be met. Half as many sets of parents are chosen as the initial population size so that, after crossover is performed, the total number of children is the same as the total number of individuals. There is not a guarantee that every set of parents will go through crossover, this being dependent on the crossover rate. After the parents are selected, a random number between 0 and 1 is generated for every set of parents and checked with the crossover rate. If the crossover rate is lower than the random number, then the crossover for those parents simply involves generating two offspring, with one identical to one parent and the other identical to the other parent. If the crossover rate is higher than the random number, the crossover operator is used. The crossover operator is described next.

Because the assembly sequence and line balance are intricately connected via the assembly sequence forming part of the line balance's precedence constraints, conventional crossover operators are not applicable; they all assume that all the precedence relations between the parents are the same, and justifiably so. If precedence relations for both parents are different, the resulting constraint incompatibilities will likely cause most of the inherited genes to be infeasible and yield children whose quality is not much higher than the quality of randomly generated individuals. In the integrated problem being tackled, both the assembly sequences and line balances of the parents need to undergo crossover. Since the line balances' precedence relations depend on the sequences, which are almost certainly not the same between parents, the line balances will have different precedence relations. This creates significant difficulties. In fact, this is why the manufacturing process, categorical geometry, and spatial constraint variables are static for each optimization instance; if they were not, the assembly sequences would also have different precedence constraints and make the situation even more difficult to handle. As it is, a new crossover

operator that accounts for the different line balancing precedence relation changes is created for this work. There is a desire to not have to incorporate any repair operators because of their complexity and so it is extremely beneficial if the children assembly sequences and line balances are always feasible. As a result, the developed crossover operator, similar to the initial population generation, makes significant use of the main concepts in Marian et al.'s guided search procedure [141].

The interconnectedness of each individual's assembly sequence and line balance means that the crossover operator requires two distinct but connected phases, one for the assembly sequence and one for the line balance. To help keep track of what genetic information belongs to which individual, some terms are clarified first. Parent A and Parent B are the parents chosen to perform the crossover. Child A is associated with Parent A in the sense that it retains all or most of Parent A's genes before what Marian et al. call the cut point while inheriting all or most of Parent B's genes after the cut point. Similarly, Child B retains all or most of Parent B's genes before the cut point while inheriting all or most of Parent A's genes after it. The cut point, also called the crossover point, is a point on both parents' chromosomes where genetic information to the right of it is swapped between both parents to create the offspring. The cut point is at the same location between both parents to make sure the same amount of genes are exchanged. It is randomly chosen such that it is always in between two parts or two tasks. This ensures that at maximum all but one gene is exchanged and at minimum at least one gene is exchanged, lest the crossover just yields the parents again.

In the crossover operator's first phase, Parent A and Parent B's assembly sequence chromosomes undergo crossover in a, relatively speaking, standard manner to yield Child A and Child B's assembly sequence chromosomes. In the second phase, Child A and Child B's interference matrices and precedence matrix are first updated to reflect their assembly sequences' constraints. The interference matrices are updated because changing which parts are integrally fabricated or part of a subassembly will affect the interference matrices.

As this information is encoded within the assembly sequence, it means it also undergoes crossover with the assembly sequence and can change between parents and their children. The interference matrices must thus be updated to account for this. After the updating of the matrices, crossover for the line balance chromosomes occurs. Because Parent A does not have exactly the same precedence relations as Child A, and similarly so for Parent B and Child B, Child A and Child B cannot directly copy their parents' respective line balance chromosomes before the cut point. In the same manner, Child A and Child B cannot directly copy the other parents' line balance chromosomes after the cut point. What is done instead is that Child A, while not violating its precedence relations, tries its best to copy Parent A's line balance chromosome before the cut point and Parent B's line balance chromosome after the cut point. The same is done for Child B with Parent B before the cut point and Parent A after the cut point. Because the child line balance chromosomes only try to copy their parents' chromosomes instead of directly copying them and do so while following their new precedence relations, the result is a line balance that is always compatible with its preceding assembly sequence and so is always feasible with no repair operators necessary. The crossover operator is summarized as follows in **List C**:

Step C1: Carry out crossover first phase: perform crossover for Parent A and Parent B's assembly sequence chromosomes to obtain Child A and Child B's assembly sequence chromosomes

Step C2: Update Child A and Child B's inherited interference and precedence matrices from their respective parents to have their respective precedence relations actually reflect what their assembly sequence chromosomes depict

Step C3: Carry out crossover second phase: perform crossover for Parent A and Parent B's line balance chromosomes. This is done by having Child A try to copy Parent A and Parent B's line balance genes before and after the cut point, respectively, while following its precedence relations, and then having Child B do the same with the

opposite parents

The exact procedure for each of the crossover operator's phases is discussed next. In the crossover's first phase, the parents' assembly sequences undergo crossover. Since the assembly sequence chromosome encodes information regarding integral fabrication and subassemblies, a change in the assembly sequence can also result in a change in which parts are integrally fabricated or part of a subassembly. The first part of the crossover first phase is therefore focused on determining how that changes when different genes are inherited while the second part is focused on using Marian et al.'s guided search procedure to inherit the rest of the genes. To start, the cut point for the assembly sequence chromosomes is determined. The cut point for assembly sequences is a value between 2 and l, with l being the number of parts in the assembly sequence. All columns in the chromosome lower than the cut point value are deemed as being "before" the cut point while all columns in the chromosome equal to and higher than the cut point value are deemed as being "after" the cut point, e.g., if the cut point is 2 in a 5-part sequence, column 1 is before the cut point and columns 2-5 are after it. Child A attempts to keep all of Parent A's assembly sequence chromosome before the cut point while inheriting Parent B's after the cut point. From here on, Tentatively Inherited Opposite Parent's Genes (TIOPG) will refer to all the genes after the cut point that the child chromosome is trying to inherit from the opposite parent, while Tentatively Inherited Associated Parent's Genes (TIAPG) will refer to all the genes before the cut point that the child chromosome is trying to inherit from its associated parent. The latter is called tentative because some circumstances will force even the associated parent's genes to be changed due to infeasibility. The TIOPG is checked to see if it has any duplicate parts that are already present in the TIAPG. Any duplicates will result in those columns in the TIOPG being zeroed out. A copy of the TIAPG and TIOPG at this moment are created called TIAPG-Original and TIOPG-Original, indicating it is the original set of the associated and opposite parent's genes before and after the cut point, respectively, minus any duplicate parts.

Rows 2 and 3 in the TIOPG and the TIAPG of the assembly sequence chromosome, which indicate which parts are integrally fabricated or in subassemblies, are then checked for "singletons." Singletons are all nonzero values in either row 2 or 3 that do not have a matching value in another column in the same row to show the parts associated with those columns are integrally fabricated or in a subassembly. All singletons are zeroed out in their respective TIOPG or TIAPG because they no longer have another part to be integrally fabricated or in a subassembly with. This occurs because the cut point can sometimes split up subassemblies and integrally fabricated parts, leaving "singleton" parts. After the singletons are removed, the assembly sequence chromosome TIOPG are temporarily appended with the TIAPG and the resulting integrally fabricated parts and subassemblies identified and checked for validity. This is done by taking parts with matching values in rows 2 and 3 and seeing if there are edges in the liaison graph that connect them to each other. Invalid integrally fabricated parts or subassemblies are those whose parts do not have edges linking them all together. Additionally, for wingboxes, valid part sets are composed of at most 3 major parts, and so if more than 3 are in a part set, the part set is invalid. If there are invalid such part sets, row 2 or 3 of the associated part(s)'s column(s) in the TIOPG is zeroed out first. Afterwards, the invalid part sets check is performed again. If invalid part sets still exist, row 2 or 3 of the associated part(s)'s column(s) in the TIAPG is zeroed out. This removes all of the invalid part sets.

At this point, the TIAPG and TIOPG are separated again and no longer temporarily appended. If any part(s) in the valid part set(s) are located in both the TIAPG and TIOPG, then the associated part(s) and their column(s) are removed from the TIOPG and placed in the TIAPG adjacent to their associated integrally fabricated part's column or their associated subassembly part's column. This is because integrally fabricated parts or subassemblies are fabricated or pre-assembled together and so, in the assembly sequence representation, placed next to each other. If the TIOPG has more than one column after all this then the TIOPG-Original is checked to see if there are any nonzero values in rows 2 and 3, indicat-

ing that there used to be an attempt to form an integrally fabricated part or subassembly. If there are, then the parts in the TIOPG are checked to see if any valid subassemblies or integrally fabricated parts can be made. If they can be made, a copy of the TIOPG at this state is created called TIOPG-Copy. Then, the TIOPG is updated to have either a valid integrally fabricated part or subassembly, depending on what the TIOPG-Original had. This is done to try to inherit the presence of a subassembly or integrally fabricated part in the TIOPG-Original if one existed.

Once all these operations regarding integral fabrication and subassemblies are finished, the second part of the crossover first phase regarding performing the actual sequence crossover is performed. How the crossover is performed is very similar to how the assembly sequence chromosome is initially generated and is detailed below in **List D**:

Step D1: Instantiate liaison graph, interference matrices, If-Then precedence rules

Step D2: Instantiate leftmost end of assembly sequence chromosome such that it is the same as the TIAPG

Step D3: Update liaison graph and interference matrices to reflect integral fabrication and subassemblies in TIOPG and TIAPG

Step D4: Add all parts not in the TIAPG to possible candidate list, initiate candidate list

Step D5: Randomly choose part, or set of parts if integrally fabricated or in subassembly, from possible candidate list not in candidate list and not yet checked to be in candidate list

Step D6: Check if removal of chosen part(s) from liaison graph results in loner nodes or groups of nodes not connected to each other

Step D7: If chosen part(s) are part of precedence rules related to ordering, check if they are the part(s) that must be placed after all the other parts in the possible candidate list

Step D8: Create temporary interference matrices by truncating interference matrices from Step D3 to only include parts in possible candidate list and TIAPG. If there are no non-zero values in any of chosen part(s)'s associated rows or columns in any of the interference matrices, chosen part(s) pass special interference check

Step D9: If chosen part(s) do not fail checks in Steps D6-D8, add it to candidate list

Step D10: Repeat Steps D5-D9 until all parts in possible candidate list have been checked to see if they can be in candidate list

Step D11: The rightmost end of the assembly sequence chromosome but left of parts already added during crossover is where the next part(s) chosen from candidate list will be added. This location will be to the right of any TIAPG locations. If, in the TIOPG, there is a part(s) in this location and that part(s) is in the candidate list, then that part(s) is assigned to this location(s) in the assembly sequence chromosome. If not, randomly select a part(s) in the candidate list and assign it to this location

Step D12: Remove selected part(s) from possible candidate list, remove its node from liaison graph, clear candidate list

Step D13: Repeat Steps D5-D12 until no more parts in possible candidate list. If no parts can be added from possible candidate list to candidate list and possible candidate list is not empty, implement fail-safe crossover first phase procedures

There are several major differences between the steps used to generate the initial sequence chromosomes and the steps listed here to perform crossover with them, though some steps have remained relatively unchanged. Step D1 is the same as its counterpart in the assembly sequence chromosome initial generation. Step D2 is a new step and done because, during crossover, a portion of the child's assembly sequence chromosome is already present via the TIAPG; a new sequence is not being created from scratch. The graph and matrices in Step D3 are updated to account for this. Parts in the TIAPG in Step D4

are not checked to see if they can be added to the possible candidate list because they have already been assembled. Steps D5-D7 did not change from the assembly sequence chromosome initial generation. Step D8 is different only insomuch that the parts in the TIAPG are already assembled and always included in the checks with the interference matrices. Steps D9-D10 are unchanged. One of the largest differences is in Step D11, which dictates how the genes from the TIOPG can actually be inherited by the child. If, at the sequence chromosome location being examined, there is a part, integrally fabricated part, or subassembly at the same location in the TIOPG that also exists in the candidate list, then that portion of the TIOPG is inherited. The random selection of part(s) is only reserved for when the TIOPG part(s) at that location are not contained in the candidate list. This is the same inheritance scheme as in Marian et al.'s crossover operator, where assembly sequence information is only inherited if it is feasible at the same gene location [141]. Step D12 is unchanged from its initial sequence generation counterpart. The fail-safe crossover first phase procedures exist in Step D13 mainly because there are many instances where valid integrally fabricated parts or subassemblies are made but end up yielding interference matrices and geometric constraints that make it impossible to create a feasible sequence. These are next to be explained.

Because the issues tend to be related to subassembly and integral fabrication part sets, the fail-safe procedures are focused on incrementally rolling back such changes, if any were made, until a feasible assembly sequence can be generated. The first procedure, should the crossover process be unable to finish generating a sequence, is to change the TIOPG back into the TIOPG-Copy and then attempting the crossover process again. This reverts the changes involved with trying to inherit any integral fabrication or subassembly genes from the TIOPG-Original. Should the first procedure fail, the second procedure is to revert the TIOPG back into the TIOPG-Original but without any subassemblies or integrally fabricated parts, essentially zeroing out the TIOPG-Original's second and third rows in its assembly sequence chromosome. This is done to try to still have the child

inherit the TIOPG-Original's part ordering even if it cannot inherit the integral fabrication or subassemblies. These two procedures are why the TIOPG are named the way they are: the genes they contain are only tentatively inherited and may need to be changed to ensure a feasible assembly sequence can be generated. Should the second procedure also fail, then it is likely the problem lies in the TIAPG. By this point, it is becoming apparent that most of the genes able to be inherited by the child primarily produce infeasible sequences and so the third procedure is to just treat this similarly to Marian et al.'s mutation operator and generate a new assembly sequence chromosome from scratch for the child [141]. Regardless of which fail-safe procedures are used, if any at all, the end result is the same: feasible assembly sequence chromosomes are created for the children. For fuselage assemblies, this ends the crossover first phase. For wingbox assemblies, there is one last step that must be carried out in the form of addressing the minor parts, which are row 4 in the assembly sequence chromosome.

Crossover for the minor parts' integral fabrication is simple compared to the process for the major parts. All the minor parts that are integrally fabricated in the TIAPG-Original and the TIOPG-Original, indicated by there being a 1 in the 4th row of the sequence chromosome in the columns of their associated major parts, are simply also made integrally fabricated in the final feasible sequence. Similarly, minor parts that are not integrally fabricated in the TIAPG-Original and TIOPG-Original are left as not being integrally fabricated. The only exceptions are minor parts that are forced to be one or the other due to the If-Then precedence rules.

The summary of everything that occurs during the crossover first phase is shown below in **List E** as it pertains to Child A. The process is repeated with the appropriate changes, such as changing which parents to obtain the TIAPG and TIOPG from, to get Child B's assembly sequence chromosome.

Step E1: Randomly select cut point to create assembly sequence TIAPG and TIOPG from appropriate parents

Step E2: Remove duplicate parts in TIOPG

Step E3: Remove singleton values in rows 2 and 3 of the sequence chromosome in the TIAPG and TIOPG

Step E4: Create copies of TIAPG and TIOPG at this point called TIAPG-Original and TIOPG-Original

Step E5: Temporarily append TIAPG and TIOPG and check validity of resulting integrally fabricated parts and subassemblies based on values in rows 2 and 3 of the sequence chromosome

Step E6: If integrally fabricated parts or subassemblies invalid, zero out rows 2 and 3 in TIOPG and check for validity again

Step E7: If integrally fabricated parts or subassemblies still invalid, zero out rows 2 and 3 in both TIAPOG and TIOPG

Step E8: Separate TIAPOG and TIOPG

Step E9: If constituent parts of valid integral fabrication or subassembly sets are present in both TIAPG and TIOPG, remove the constituent part(s)'s column(s) from TIOPG and add them to TIAPG in the column(s) adjacent to the associated set's part(s)

Step E10: Create copy of TIOPG at this point called TIOPG-Copy

Step E11: If TIOPG-Original had any non-zero values in rows 2 or 3 of its chromosome, try to make valid subassembly in TIOPG if non-zero values were in row 3, valid integrally fabricated parts in TIOPG if non-zero values were in row 2, either if non-zero values in both

Step E12: Perform actual crossover as described in Steps D1 to D13 of second part of crossover first phase

Step E13: If Step E12 does not yield feasible sequence, change TIOPG back to TIOPG-Copy and try Step E12 again

Step E14: If Step E13 does not yield feasible sequence, change TIOPG back to TIOPG-Original, zero out rows 2 and 3 of chromosome, and try Step E12 again

Step E15: If Step E14 does not yield feasible sequence, generate brand new sequence via initial assembly sequence chromosome generation

Step E16: If analyzing wingbox assembly, perform crossover for minor parts

Before the crossover second phase is performed where crossover is done for the line balance chromosomes, Child A and Child B's precedence matrices are updated to match their final assembly sequences from the crossover first phase. Their list of available tasks, the tasks' task times, and the tasks' task costs are similarly updated.

Although the crossover second phase is less conventional than the first phase, it is much simpler to perform. Conceptually, it is much the same as the crossover first phase in that a one-point crossover is used. First, the cut point is determined. For line balances the value of the cut point is allowed to range between 2 and the minimum between λ_A and λ_B, where A and B are the children and λ is the number of tasks in their list of available tasks. The cut point is applied on Parent A and Parent B's line balance chromosomes to obtain the line balance chromosome TIAPG and TIOPG for Child A and Child B. From here on, the rest of the explanations will focus on Child A, whose line balance TIAPG is from Parent A and whose line balance TIOPG is from Parent B. The same explanations apply for Child B except that the TIAPG and TIOPG are obtained from the opposite parents.

Line balance crossover operations can only be meaningfully performed if the line balances involved share the same precedence relations, which in this case means the same precedence matrices. If the precedence matrices are different, there is a high probability that the majority of the inherited genes will not be feasible and cause crossover to be little better than the initial population generation, defeating the point of a genetic algorithm.

Parent A does not have the same precedence relations as Parent B and, due to the crossover first phase, it is almost certain that neither have the same precedence relations as Child A because they all have different assembly sequences and integrally fabricated parts and sub-assemblies. The concept of Marian et al.'s guided search approach is used to amend this by having the children only inherit genes from either parent if it is feasible to inherit them [141]. This results in offspring that are always feasible with no need for a repair operator to fix anything afterwards. Though the guided search approach was made for ASP, there is little conceptually that prevents it from being used for ALB.

Unlike with the crossover first phase, Child A cannot just try to copy its line balance TIAPG to start off its own line balance chromosome due to the likely difference in precedence relations between Child A and Parent A. It thus has to carry out the guided search approach for both the line balance TIAPG and TIOPG, instead of just the TIOPG. One other major difference is regarding how genes are inherited. During the crossover first phase, genes were inherited from the TIAPG and TIOPG on a positional basis, i.e., if at specific chromosome location there is a feasible and inheritable gene in the TIAPG or the TIOPG, then it will be inherited. Inheriting genes on a positional basis is perhaps the most common way to implement inheritance in both the ASP and ALB GA literature because it is very simple. However, it is not very useful for the crossover second phase because the changing assembly sequences and precedence relations cause the number of available tasks to constantly be changing. As the line balance chromosome length is based on this, it is also constantly changing. Positional-basis inheritance schemes work because the chromosome lengths stay constant and result in a high chance of a feasible and inheritable gene being in the same position between parent and child chromosomes. When the lengths differ, this is no longer necessarily the case. If chromosome locations where genes are feasible are shifted or offset, which frequently occurs when precedence relations and number of available tasks are changing, it can result in situations where the child chromosome is never able to find a feasible and inheritable gene from the parent for any given location because the

parent's genes' feasible locations are all two genes over. The offspring thus is never able to inherit anything, resulting in a crossover process that is essentially the same as randomly generating an individual and defeating the point of a GA.

Due to the immense disadvantages of positional-basis inheritance schemes, a station-based inheritance scheme is used instead for the line balancing crossover. Instead of storing where the parent's genes are located in their chromosome and trying to inherit that, what is instead stored and inherited is which tasks belong to the same station and which ones have a station to themselves. This eliminates the dependency on the ordering and location of the tasks because the only information that matters is which tasks share the same station, which can be applied to any location on the chromosome. It is also simple, scalable, and flexible, as opposed to a scheme where the relative ordering of tasks is stored, which would run into significant issues with tasks not being available or having different precedence relations between parent and child chromosomes. Additionally, the point of the positional-basis scheme is to preserve the ordering of the tasks as well as trying to keep the same tasks in the same station through the ordering-preservation. With the ordering-preservation not being practical with changing precedence relations, at least the same-station-preservation is kept intact with this same-station inheritance scheme, which is why it will be used. The resulting procedure for the crossover second phase is described below in **List F** for Child A. It can be applied to Child B if the roles of the parents are reversed.

Step F1: Update child's precedence matrix based on its assembly sequence, integral parts, and subassemblies

Step F2: Remove all unavailable tasks from precedence matrix

Step F3: Randomly select cut point to create line balance TIAPG and TIOPG from appropriate parents

Step F4: Use precedence matrix to identify and add all feasible tasks to candidate list

Step F5: If first task is being assigned, randomly select task to assign to station 1

Step F6: If task is not first task, check previous task and current chromosome location. If chromosome location is before cut point, refer to TIAPG, if after cut point, refer to TIOPG. If previous task shared a station with other tasks in TIAPG/TIOPG, check if other tasks are in candidate list. If they are, randomly select one of them currently in candidate list and assign it to station, station number is same as previous task

Step F7: If previous task did not share a station with any tasks in TIAPG/TIOPG, randomly select task in candidate list and assign it to station, station's station number is incremented by 1

Step F8: If task is not first task and no tasks were assigned with either Steps F6 or F7, randomly select a task from candidate list and assign it to station. Check selected task's zoning constraints with previous task. If selected task cannot be in same station as previous task, selected task's station number is incremented by 1. If selected task is able to be in same station as previous task, randomly determine if selected task will be in same station. If yes, selected task's station number is same as previous task's. If not, increment selected task's station number by 1

Step F9: Update precedence matrix by removing selected task's associated row and column, clear candidate list

Step F10: Repeat Steps F4-F9 until no more tasks left

The majority of these steps are the same as for the initial line balance generation. The most noteworthy differences are in Steps F6 and F7, which incorporate the described same-station inheritance scheme. The result of this process is a feasible line balance chromosome that has inherited some genes from its parents and does not need any repair operators to fix it. This wraps up the crossover second phase and concludes the crossover operator in general.

This work's crossover operator took major inspiration from Marian et al.'s work due to it being able to handle complex assemblies [141]. The same inspiration is taken for the mutation operator, resulting in a very simple mutation operator. Due to the complexity involved with using the guided search approach, there is no straightforward way to implement a mutation operator using it without having the mutation operator look extremely similar, if not identical, to the crossover operator. Marian et al. thus did not even attempt it and so simply generated a new individual for their mutation operator. This work does the same. After crossover is performed, there is a low chance for either or both of the offspring to undergo mutation. Should mutation be performed, one of two mutation types is randomly picked to be carried out. The first mutation type keeps the offspring's assembly sequence chromosome but generates an entirely new line balance for it. The second mutation type generates a brand new assembly sequence first and afterwards generates an entirely new line balance based on that assembly sequence. Two different types are used because a local minimum can sometimes be better escaped by only changing the line balance while at other times a new assembly sequence is needed as well. Changing the assembly sequence changes the precedence relations, necessitating that the line balance be changed as well; the assembly sequence cannot be changed by itself without affecting the line balance. Because assembly sequence and line balance generation has already been covered, they will not be further discussed. Offspring that undergo mutation are replaced by their mutated versions. The mutation operator, because it uses the initial sequence and line balance generation procedures, thus indirectly uses the guided approach and so, like everything else, always results in feasible individuals in terms of line balance.

Situations will arise, however, where individuals are infeasible in terms of space occupied in that, given their line balance with only one instance of every station, the individual still takes up more space than is available. In such a scenario, the individual's production rate is set to a very low number and its cost set to a very high number so that, when fitness evaluation is performed later on, the spatially infeasible individual has no chance of being

selected to move on to the next generation.

After crossover and mutation have both been completed, there will exist individuals whose number is double that of the initial population P, half of which are composed of the initial population itself and half of which are the offspring resulting from crossover and mutation. The number of individuals allowed to go on to the next generation is the same as the initial population, and so all these individuals must be evaluated to see which ones will be replaced and which will make up the next generation's initial population. The evaluation process is very similar to the one used for the parent selection, with similar justifications. If the maximum physically feasible production rate subproblem is being solved, all the individuals are ranked based on their throughput. As this replacement process is elitist, the top P individuals become the initial population for the next generation and the rest are replaced and removed. If the minimum APUC subproblem is being solved, the throughputs of all the individuals are checked first and the individuals sorted into two groups, one where the individuals have a throughput equal to or higher than the current throughput ε-constraint and one where it is lower. All of the individuals in the first group are ranked according to their APUC, with lower APUC being better, and the top P individuals are chosen to become the initial population for the next generation and all other individuals replaced and removed. If there are less than P individuals in the first group, the entire group is chosen and attention is shifted to the second group whose individuals have throughputs lower than the ε-constraint. The second group's individuals are ranked according to their throughput, with lower APUC only being considered if individuals have the same throughput. A number of the top individuals in the second group are chosen such that, combined with the individuals in the first group, there are a total of P individuals to seed the next generation's initial population. All other individuals are replaced and removed.

The stopping criteria for each subproblem is based on how much better the best solution in each generation is than the stored best solution. At the end of every generation, the best solution is determined based on which subproblem is being solved. If the maximum phys-

ically feasible throughput subproblem is being solved, the best solution is the one with the highest throughput. If the minimum APUC subproblem is being solved, the best solution is the one that meets or exceeds the ε-constraint with the lowest APUC. Best solutions are compared to each other in terms of throughput for the maximum throughput subproblem and APUC for the minimum APUC subproblem. If the best solution in the current generation is not better than the stored best solution by at least certain percentage, a counter is started that begins at 1. The counter increments by 1 every generation if that generation's best solution is not better than the stored best solution by at least certain percentage. If the counter reaches a generation number limit without seeing sufficient improvement, the optimization is stopped and the best stored solution is considered the optimized solution for the subproblem. If a generation's best solution is better than the stored one by at least a certain percentage, that generation's best solution becomes the stored best solution and the counter is reset. The best solution in the first generation automatically becomes the stored best solution and the counter is allowed to start, at earliest, in the second generation.

The APUC subproblems are solved in order of decreasing throughput ε-constraints. To limit the amount of time spent generating initial populations, the APUC initial populations use the same aircraft design variables, assembly sequence, and line balance as the max throughput subproblem's individuals in its initial population. This does not affect the throughput and cost quality of the resulting designs because the main factor affecting that is how stations are duplicated, which changes at every ε-constraint's subproblem.

To ensure that the best solution obtained through the APUC subproblem at each throughput ε-constraint is never completely dominated by the best solution from the previous throughput ε-constraint's APUC subproblem, a special seeding process is performed. The best solution from the previous throughput ε-constraint's APUC subproblem is always included in the current APUC subproblem's initial population. For the first APUC subproblem to be solved after the maximum physically feasible throughput has been determined, the best solution used to perform the seeding is the solution that yielded that maximum

throughput. This ensures that at least one solution will always meet the throughput ε-constraint, which can otherwise be very difficult at very high throughput levels. It similarly ensures that every APUC subproblem's best solution will be at least as good as the previous subproblem's because, in the worst case scenario, this initial seeding causes them to be the same.

5.5 Experiment 1: Comparison of Assembly Frameworks

5.5.1 General Setup

Experiment 1 consists of using the new framework and the literature state-of-the-art to analyze the same assemblies with the same input information and seeing which is able to provide better tradeoffs based on their outputs, both in terms of the resulting Pareto front and the stored information. The definition of "better tradeoffs" in this experiment is diverse. The first and most quantifiable condition for one of the methods to be better is that they must be able to yield Pareto fronts of higher quality, i.e., output more non-dominated throughput and cost solutions. In this regard the better method is able to output better actual results and values. But having better results is not sufficient on its own because those better results could have been obtained via dubious assumptions or in such a way that it is not obvious why they are what they are, hampering further efforts to improve them. The second, more important condition is that the better method must be better able to accommodate the material factors and examine how those factors impact the cost and throughput. The ability to see how the newly integrated material factors impact the throughput and cost provides for the data traceability needed to justify why results look the way they do. It additionally allows tradeoffs to be made by seeing which material factors should be altered for further optimization. The better method having better accommodations for the material factors enables such alterations in the first place, which is why it is also part of the second requirement. The method that meets both these conditions is considered to be able to provide better tradeoffs, though at the minimum the second condition must be met; meeting

the second condition means more tradeoffs can be made and made in a traceable and justifiable way, which is more important than having solutions with better values that are less justified.

The focus on the material factors dictates that the independent variables in this experiment revolve around them. This means that, beyond the obvious variables in the form of the assembly sequence and line balance being involved due to this being an integrated ASP and ALB problem, the main independent variables are the manufacturing process used, the parts being integrally fabricated, and the factory's spatial constraints.

Metrics to determine which method is more accommodating to the material factors as well as able to observe their impacts are difficult to define due to the qualitative nature of being able to make tradeoffs. This can only be determined by examining each method's final outputs and cannot be defined a priori. Metrics for determining the quality of the Pareto fronts, however, can be determined a priori. These metrics, according to Zitzler et al., are called quality indicators [220]. Also according to Zitzler et al., when trying to compare two different Pareto fronts, the only types of indicators able to conclusively determine that one is better than the other, or dominates the other, are binary indicators. Binary indicators are indicators specifically made to compare two different Pareto fronts, as opposed to unary indicators which evaluate only one. A binary indicator must thus be at least one of the indicators used to compare the Pareto fronts generated by the two methods in Experiment 1.

The binary indicator of choice for this experiment, and for this work in general, is the $\epsilon - Indicator(I_\epsilon)$, also sometimes called the multiplicative $\epsilon - Indicator$. $I_\epsilon(PF_1, PF_2)$ measures the smallest factor ϵ that must be multiplied with the solutions of the first Pareto front being compared PF_1 in order for all of its solutions to dominate the solutions of the second Pareto front being compared PF_2. This type of domination is called ϵ-domination. $I_{epsilon}$ is chosen because it is a commonly used indicator that can definitively determine whether one algorithm to create a Pareto front generates a better-converged front than an-

other algorithm [220]. It is additionally inexpensive to compute. Given that y are the individual solutions in the first Pareto front PF_1, z are the individual solutions in the second Pareto front PF_2, $I_\epsilon(PF_1, PF_2)$ is calculated for the n number of objectives in each as:

$$I_\epsilon(PF_1, PF_2) = max_{z \in PF_2} min_{y \in PF_1} max_{1 \leq i \leq n} \frac{y_i}{z_i} \tag{5.12}$$

$I_\epsilon(PF_1, PF_2)$ is a measure of how much PF_1 ϵ-dominates PF_2. Similarly, $I_\epsilon(PF_2, PF_1)$ is a measure of how much PF_2 ϵ-dominates PF_1. Both need to be calculated to compared the two Pareto fronts. If $I_\epsilon(PF_1, PF_2)$ is less than 1 and $I_\epsilon(PF_2, PF_1)$ is greater than 1, then it can be conclusively stated that the first Pareto front is completely better than and dominates all of the second Pareto front. If this condition is not met then, in general, lower $I_\epsilon(PF_1, PF_2)$ values are better because they indicate that one Pareto front is closer to completely dominating the other.

The $I_{epsilon}$ indicator is primarily used to see if one Pareto front dominates the other, but says little about how diverse the solutions are or how much of the design space they cover. A useful indicator that can do this and provides a more holistic measure of the Pareto front's quality is the hypervolume ratio indicator HVR [221], calculated as:

$$HVR = \frac{HV(PF)}{HV(PF^*)} \tag{5.13}$$

PF^* is the true Pareto front, compared to PF which is only the approximate Pareto front. If the true Pareto front is not known, it is common practice to combine together all the different Pareto fronts being calculated, extract all the non-dominated solutions among them, and treat those as the true Pareto front. HV is a Pareto front's hypervolume, or the volume of space taken up by the Pareto front's solutions [222]. It is calculated by selecting a reference point, often a point that takes extremely suboptimal values in all the objectives being examined, and using it to calculate the hypervolume encapsulated by the Pareto front.

If trying to compare multiple different Pareto fronts, the reference point must be the same for all of them. A ratio is used in this case to prevent one metric from overshadowing the other simply due to it having values or units scaled significantly higher [221]. HVR is chosen as the metric to combine convergence and diversity because, while it is a unary indicator, it is perhaps the only one of its kind able to ensure that Pareto fronts with higher values are at least not worse than Pareto fronts with lower values [220]. It additionally has a special trait where if a Pareto front's HVR value is equal to unity, it guarantees that that Pareto front dominates the ones it is being compared with [223]. This makes it a representative and relatively accurate metric. As the equation implies, HVR values range between 0 and 1 with better Pareto fronts having a HVR value closer to unity.

Additionally, it can be desirable to visualize the Pareto fronts generated and judge them more qualitatively because the quality indicators are inconclusive. Because generating Pareto fronts is stochastic by nature, with there being no guarantee that multiple runs of the same algorithm on the same problem will generate the same front, multiple Pareto fronts will need to be generated to get an idea of their true shape. This can make it difficult to visualize because a large number of different fronts representing different runs can quickly clutter up a graph. For this reason, Knowles's summary attainment surfaces will be used to visualize Pareto front distributions [224]. Summary attainment surfaces essentially plot out a Pareto frontier that dominates a user-defined portion of all the Pareto frontiers it is summarizing. More details on them can be found in Knowles's work [224].

The quality indicators for the Pareto fronts and how they can be visualized have been dealt with. The last metric of interest is that, sometimes, it is desirable to ascertain how quickly a method takes to obtain the optimal solutions; the best solutions that take colossal amounts of time to obtain are not useful. A computational expense metric is thus desired. As different algorithms and computers can cause large amounts of variability in a computational time metric if it were used, the number of generations required by a method to converge is used instead. It depends less on the hardware used and also better relates the

efficiency of a method, with more efficient ones requiring less generations. With this, all the metrics needed to compare the framework and the literature state-of-the-art have been determined.

With the experiment's more general details set up, the specifics are now focused on. A method or approach must be selected to test against. Such an approach was identified back at the start of section 5.2 as the work done by Oesterle et al. [76]. Their work is relatively comprehensive in that it combines the assembly subgraphs problem, the TSALBP, and the equipment selection problem while being multi-objective in nature. While it does not include the station paralleling problem and does not address the issues not handled by current ALBPs, it is still able to incorporate all the other problem types that are related to the material factors and is well-tested. Additionally, it was shown in their work that the NSGA-II, one of the most well known GAs, is one of the best multi-objective algorithms. Based on this an NSGA-II implementation of their work is representative of their overall approach, meaning their work utilizing a GA will be compared to the developed framework that also utilizes a GA; this allows for a more one-to-one comparison and is further incentive to use their work to test against. For these reasons the developed framework will be compared with Oesterle et al.'s work.

Due to the nature of the ASP and the material factors on the ALBP being addressed in Experiment 1, several changes need to be made to Oesterle et al.'s approach in order to even enable its use. Because ASP exists in place of assembly alternatives, a bank of predetermined assembly sequences to choose from and all the required resulting precedence matrices will replace their assembly subgraphs structure. Essentially, if a change in assembly or product alternatives is desired in their approach, an alternative assembly sequence will be selected instead. Because they open and close their assembly stations based on whether the station times exceed a user-defined cycle time, and such a user-defined cycle time is not provided for this problem, the cycle times are chosen to be the product of whatever the maximum task time is and a random value between 1 and 2. Values lower than 1

cannot be used due to the lack of station duplication considerations in their approach.

This all causes some notable changes to have to be made to their crossover and mutation operators. For the crossover operator, crossover is only able to occur if the two parents share the same assembly sequence due to the possibly significant difference in precedence relations otherwise. When their assembly sequences are different, new line balances are generated for the offspring, who keep their associated parent's sequence. The crossover otherwise occurs much like a typical one-point crossover where, after the cut point is determined, the offspring copies the TIAPG and inherits the TIOPG via a position-based inheritance scheme that uses guided search to always ensure feasible line balances. Of Oesterle et al.'s four mutation operators, the equipment change one is not used because there is only one feasible equipment type for every task. The station change one is similarly impractical to use due to zoning constraints causing the random assigning of tasks to different stations generally infeasible. The swap task and insert task mutations are slightly altered for ease of use. They now both use the guided search approach so that whenever tasks are swapped or inserted, the result is always feasible. For the swap mutation, this is done by selecting a task to swap, placing it in the earliest station possible, and then using guided search to try to copy the rest of the original line balance. For the insert mutation, a random location in the chromosome is selected and a random feasible task inserted there, with guided search being used to try to copy the rest of the original line balance. Additionally, because the swap task mutation is also meant to change the assembly alternative being examined, a new mutation scheme called "swap sequence" is created to capture its intent: whenever the swap sequence mutation is invoked, it randomly selects another assembly sequence to use and subsequently generates a brand new line balance that can be used with it.

These changes mean that the approach being compared with the new framework is called Oesterle et. al's approach despite it being changed somewhat to suit the assemblies and problems being analyzed. However, the spirit and intent of Oesterle et. al's approach remains the same and, despite some slight capability augmentations such as the using of

guided search, their approach's overall capabilities are still about the same. This means their original, unchanged approach is still equivalent to the one with some changes that is being compared against. This subsequently means Experiment 1's comparisons are still valid.

Attention is now turned to the assemblies to be analyzed by the developed framework and Oesterle et. al's approach. Due to the need to have the material factors as variables, the assemblies must thus have sufficient manufacturing information for both their fabrication and assembly steps to be modeled, must have said information for multiple manufacturing processes, must be able to be made from composite materials for integral fabrication, and must have enough data such that their task times for any of these variations is known or that these task times can be estimated by parametric estimation software. Additional requirements include the existence of tooling cost, equipment cost, and equipment space data for all the different processes that can be used to make the assembly. The most commonly analyzed ASP assembly used in numerous ASP works focusing on developing new approaches is an engine transmission assembly, an example of which can be found in De Fazio and Whitney's work [85]. This assembly does not meet the above requirements.

There are no similarly and singularly representative assemblies in the ALB literature. What the ALB literature has instead is a large number of well known and solved ALBP instances that many works use to benchmark their methods. These do not meet the above requirements either because they essentially only have task time data for one configuration and do not have anything regarding integral fabrication, different processes, equipment cost and space data, and multiple other key pieces of information. Works like Chica et al.'s get around this by creating data to use for those assemblies and problem instances [75]. The issue with this is that the point of those problem instances is they have a known answer. To have to create data for them, as is needed to use them for Experiment 1, is to significantly change them enough that there is no more known answer, at which point there is little benefit to using them over any other assemblies with the same amount of information. On

top of this, such information may not be able to created even if it were desired: it would be extremely difficult to come up with a process to fabricate and assemble a composite transmission assembly and especially to have it incorporate integral fabrication. This means the typically-used instances are not necessarily the best assemblies to analyze because they don't have enough data themselves and it might not even be possible to generate that data for them in the first place. It subsequently means that there is essentially free reign on what assemblies are analyzed so long as there is sufficient data for it.

Because this work is focused on aircraft assemblies, it is desirable to have the assemblies in question be aircraft assemblies. The aircraft assembly with sufficient data to be used for Experiment 1 is the F-86 wingbox assembly used by Sirirojvisuth and the majority of the MInD works described in subsection 2.2.2 [58, 60, 62, 63, 29]. The F-86's aircraft and assembly design has been completely declassified for many years, allowing those works to build an extensive repository of information related to how its wingbox was designed, fabricated, and assembled, as well as how to extend that to many different manufacturing processes. Combined with the usage of parametric estimation tools and methods and the availability of manuals describing the F-86's structure, its manufacture can be easily simulated for a large variety of different configurations, including all the ones required to address the material factors. The F-86 wingbox is therefore the assembly of choice to be analyzed by the developed framework and Oesterle et al.'s method for Experiment 1. In fact, due to it being just about the only assembly, never mind aircraft assembly, for which sufficient amounts of the required manufacturing information exists, it is to be made the experimental platform that all the more assembly analysis-focused experiments will operate on. All of its characteristics, manufacturing information, and the data related to it are located in Appendix A.

One caveat must be mentioned regarding the F-86 experimental platform as well as the nature of this work. As explained by the MInD works that first developed the F-86 experimental platform and even as mentioned throughout this work's appendices that describe it,

the experimental platform as well as this entire work do not claim to provide improved or more accurate cost estimates as compared to actual validated cost estimating relationships [60, 61, 29]. The tangible, exact, numerical trends and values displayed and obtained in all the results in this entire work should not be used to design a real aircraft wing. Despite the experimental platform being designed to be as realistic as possible, it is still primarily a platform to perform design experiments on and does not exactly match real life. Rather, its purpose is to be grounded enough that tradeoff and visualization capabilities demonstrated using it as the test subject carry enough validity and weight to prove they can used for real-world designs when validated data is actually provided. The focus is thus on the tradeoff capabilities, not the physical tradeoffs of the experimental platform itself, as is apparent from the fact the F-86 is not a viable fighter aircraft today and so will not go into production again.

5.5.2 Implementation Details

The geometry of the F-86's wingbox for each of the manufacturing processes is needed in order to be able to estimate the task times and costs using parametric estimation methods and tools. This is obtained via the process described in section A.1. What the general shape of the geometries for every part look like are shown in Appendix B. The wingbox is sized as opposed to using its official documentation values because some needed values were not documented and the values do not exist for different material systems and manufacturing processes. Of note is that the usage of NASA's FLOPS program and Corman et al.'s [187] Rapid Airframe Design Environment (RADE) framework to perform the aircraft design and structural optimization that yields the final part geometries is the implemented form of early preliminary aircraft design described in Conjecture 1.1. The parameters used to obtain the wingbox dimensions are the F-86's default characteristics, which are shown in Table 5.1. They are adapted from [225].

Based on the sizing and due to the usage of a wingbox, the major part types are iden-

Table 5.1: Major F-86F Parameters, Adapted from [225]

Parameter	**Value**
Wing Sweep (25% Chord)	35°
Wing Taper Ratio	0.52
Wing Planform Area	313.4 ft^2
Aspect Ratio	4.88
Span	39.1 ft
Length	37.5 ft
Height	14.7 ft
Empty Weight	11,125 lbs.
Design Takeoff Weight	18,152 lbs.

tified as the front spar, rear spar, upper skin, lower skin, and ribs. The minor part types are the front spar's stiffeners, the rear spar's stiffeners, the upper skin's stringers, the lower skin's stringers, and the ribs' shear ties, stiffeners, and rib spar joints. For convenience, the spar joints are always fabricated and assembled with the shear ties and are considered synonymous.

Several observations were noted when performing preliminary geometric reasoning for this wingbox assembly. The details are covered in section A.2, but the main detail of importance for Experiment 1 is that the order that the individual ribs are assembled in, relative to each other, does not affect the assembly sequencing at all. The feasibility and accessibility of the sequences does not change regardless of the order the individual ribs are inserted. This applies to all the minor parts as well. Their individual assembly orderings relative to their own part types do not impact accessibility or feasibility. Because of this, all the individual ribs and each minor part type's parts are treated as one part. If one rib or minor part is to assembled, then all the individual parts of the corresponding part type are to be sequentially assembled at the same time or one after the other. Additionally, it is impractical to integrally fabricate or subassemble some individual parts in a part type

and not others. Therefore, if one rib or minor part is to be in a subassembly or integrally fabricated, the same is done to all the individual parts of that part type too.

This means that there are effectively only five major parts for assembly: the front spar, the rear spar, the upper skin, the lower skin, and the ribs. Their respective minor parts are front spar stiffeners, rear spar stiffeners, upper stringers, lower stringers, shear ties and spar joints, and rib stiffeners. In terms of the assembly sequence representation, the numbering scheme for the wingbox in Figure 5.4 is used: the front spars are represented by the number 1 in the first row, the rear spar with the number 2, the upper skin with 3, the lower skin with 4, and the ribs with 5.

For the parametric cost and time estimation, it is more convenient to use an established tool as opposed to making a new method from scratch if the tool has the proper capabilities. Perhaps the best parametric estimation tool identified from the ones surveyed by Layer et al., Boehm et al., and Hueber et al. is Galorath Inc.'s SEER [212, 226, 227]. SEER is a widely used commercially available cost estimating tool with decades of data and experience to verify its parametric cost estimates. These estimates are based on historical data from various different field, including aerospace, and are constantly updated. The SEER suite has a variety of different products, with the most pertinent being SEER for Manufacturing, or SEER-MFG.

SEER-MFG will be used to provide the labor, tooling, and material cost estimates as well as the task time estimates. SEER-MFG uses a process-based methodology in the sense that its user interface requires the selection of the process and details to be used for a particular task and, based on those selections, outputs results estimated via internal parametric equations in its library. This makes it resemble a bottom-up approach akin to the generative-analytical class of methods despite being a parametric cost and time estimation tool. As a result, the inputs required for SEER-MFG are more detailed and process-oriented, allowing the tool to accurately estimate costs if the designer is able to provide the necessary manufacturing process information [228]. Though more effort is required for the input,

SEER-MFG's combination of a bottom-up and parametric approach allows it to be used in early design. Its bottom-up aspects allow for details available in preliminary and early preliminary design, such as the fabrication, assembly, and costing of ribs, stringers, and spars, to be accurately captured, while the parametric aspects allow for designers to easily use the tool without being experts in manufacturing. SEER-MFG is specifically able to build up composite and aluminum materials according to how they are sized and produced, making it conducive for the structural analysis, material selection, and process determination aspects of aircraft design. It is thus a good complement for the aircraft design discipline. On the manufacturing side it is able to estimate labor, tool, and material costs incurred by the fabrication of any part or assembly of any two or more parts based on how large the parts are, the process used to make or join them, which specific tool options are used, etc. SEER-MFG has proven effective enough at estimating aircraft manufacturing cost [229] and conducive enough to aircraft structural analysis and design that it was the tool of choice to integrate the aircraft design and manufacturing disciplines by past MInD works and other works [61, 60, 29, 230]. It will therefore be leveraged to provide the aforementioned costs as well as the task times.

The details regarding SEER-MFG's various outputs, how it calculates them, and the information and assumptions used to aid its calculation are covered in section A.4. The most noteworthy details are regarding how SEER-MFG, from now on referred to as just SEER, calculates labor time. SEER splits its labor time into three different categories: direct, setup, and inspection and rework. Direct labor is the time spent performing the task in question, such as performing the hand layup in a hand layup task or trimming the part with a water jet in a water jet trim task. Setup labor is all the time preparing the part, the equipment, and the tooling to get it ready for direct labor. This includes the time spent cleaning and preparing the tools and tooling and loading and unloading the part. Inspection and rework time is distinct from the Non-Destructive Inspection (NDI) tasks because it refers to in-process inspection done by the workers as opposed to a dedicated NDI machine.

Although SEER outputs the total labor time for each task, that time is not necessarily the task time. SEER's total labor time estimate assumes there is only one worker performing the task, an assumption which cannot be changed in SEER. The total labor time thus is only the task time if one worker is working on the task, which is a situation that almost never happens in real life aircraft manufacturing. Adding additional workers to a task reduces the task time, but not in a linear way. According to Siedlak et al.'s work, the task time is 60% of the total labor time if 2 workers work on a task, 40% with 3 workers, and by extrapolation, 30% with 4 workers and 25% with 5 workers [29]. On top of this, adding workers to a task is only beneficial to the non-automated parts of a task because more workers will not make a machine work faster. Workers thus only apply their full labor time reduction potential to manual tasks and the non-direct labor components of automated tasks. The list of automated vs non-automated task types and the number of workers used for each in this work is shown in section A.4. The number of workers for each task type will not be changed or varied as a simplifying assumption to reduce the otherwise staggering number of line balances that would need to be done given the large number of added variables. The task time is ultimately obtained by taking SEER's total labor time estimate and scaling it based on the number of workers on that particular task.

Of note is that the accessibility of an assembly sequence as determined by the geometric reasoning process can be used in SEER to alter the labor time and, therefore, the task time of a task. This usage of accessibility to alter how long it takes to finish a task represents how the accessibility determined from Conjecture 1.2 is implemented. Additionally, integral fabrication and subassemblies are accounted for by changing how many parts are assembled at once and what their resulting joint widths are or by changing how large the parts are to represent their integrated size.

SEER is able to output a labor cost, but its labor cost is not used because it does not account for any inefficiencies or idle time in a workstation. The actual labor cost used for each instance of a station per wingbox produced is the product of the cycle time, the

labor rate, and the number of workers at that station. This labor cost calculation works out to be the same as that shown in Equation 5.8. The labor rate itself in that equation is based on past MInD works, which uses a burdened labor rate of $200 per hour for both the fabrication and assembly tasks. A burdened rate accounts for the labor of indirect workers such as the people in the engineering department, the HR department, etc. The number of workers per station in that equation was discussed just previously and is based on the task type carried out by each station.

SEER's outputs are only able to be used to obtain the task times, material costs, and tooling costs. It is unable to provide equipment costs, their spatial constraints, or a station's total spatial requirements. How this is all obtained and calculated is discussed in detail in section A.5. The main pertinent details are that, because almost all equipment cost and space requirements are proprietary, it is assumed that the equipment for each task type is representative of those in real life and flexible enough that it can operate on all parts but that is sized to the largest part in the equipment's station. Minor part assembly tasks can be assigned in the same station as major part assembly tasks, but they will need separate sets of tooling. The tooling is based off of pulsable jigs that can support the entire wingbox assembly while being to move around the factory [231], enabling line balancing of the assembly tasks. The space requirements themselves are as described in Equation 5.1 and include space for the equipment itself, the loading and unloading space for the part, space for handling for workers to move around in, space for parts to be stored in queue while other tasks in the same station are carried out, and aisle space to transport the parts between stations. Part transport time between stations is beyond the scope of this work and is not accounted for. It is assumed that every fabrication task needs its own tooling and that every assembly station must have at least one set of tooling. All sets of tooling and equipment are assumed to last the entire production run without needing to be replaced.

5.5.3 Additional Setup Details, Design Variables, and Procedure

To carry out the comparison, several parameters need to be set for the developed framework's GA. Based on preliminary tests, the best results for the developed framework's GA were obtained when the population size is set to be 200 individuals, the same as in Oesterle et al.'s work [76], and the crossover rate set 80% and the mutation rate to 10%, the same as in Chica et al.'s work [75]. The stopping criteria is that if the current generation's best individual is not 1% better than the previous best individual for 10 straight generations, then the solution is assumed to have been converged on and the optimization stopped. To capture the stochastic nature of generating Pareto fronts, 5 Pareto fronts are generated at every variable setting to obtain the "average" Pareto front, which is a compromise between not generating enough Pareto fronts to capture their variation and spending too much time generating too many fronts on just one of the many variable settings. The number of wingboxes produced in a production run, and thus the number of units to amortize the tooling and equipment costs by and set the labor and material costs, is set to 2000. This means 1000 full wings, each composed of 2 wingboxes, will be built, which is representative of how many aircraft are built in a successful commercial aircraft program. When comparing Hypervolume Ratios, a reference point is needed to calculate the hypervolume. The reference point's APUC is set to be $1.2 million per wingbox, guaranteed to be higher than all the costs of the individuals that will be on the Pareto but not too much higher, while the monthly throughput is set to 0. Due to there not being an established true Pareto front, the best solutions from both the developed framework and Oesterle's implementation for each set of Pareto fronts are treated as the true Pareto front. For the developed framework, after the maximum feasible throughput has been determined, 29 evenly-spaced ε-constraints between 0 and the maximum throughput are set such that there are a total of 30 throughput levels for each Pareto front.

For Oesterle et al.'s GA implementation that is used as the baseline, the parameter settings are kept the same as in their work using the NSGA-II implementation: a population

size of 200, a crossover rate of 55%, and a mutation rate of 3%. The generation limit for the NSGA-II is set to be 250 generations.

Because the focus of this Experiment is on the material factors, design parameters related to those material factors are used as Experiment 1's variables. These variables include: the manufacturing processes used, the spatial constraints used, and the setting of which parts are integrally fabricated or in subassemblies and which are not.

A total of four different manufacturing processes will be examined for Experiment 1: Hand Layup (HLU)/Autoclave, HLU/Vacuum Assisted Resin Transfer Molding (VARTM), HLU/Stitched Resin Infusion (SRI), and Computer Numerical Control (CNC) Machined. The HLU/VARTM process from now will just be referred to as VARTM and the HLU/SRI process as just SRI. The HLU/Autoclave process is a composite process where pre- impregnated, or prepreg, carbon fiber is laid up by hand onto a mold, which is afterwards placed inside of an autoclave that applies tremendous amounts of heat and pressure to cure it. The VARTM process has dry plies of carbon fiber being laid down by hand instead of prepreg ones. A vacuum bag is then placed over the plies and the mold it is on and a vacuum is applied to compress and consolidate the plies, draw resin into the bag, and impregnate the plies. The bag, mold, and impregnated plies are moved into an oven that applies a tremendous amount of heat to cure the plies. SRI is similar to VARTM except that the plies are stitched together first with thread before they are vacuum bagged, with the stitching allowing SRI to be the only process that allows for integral fabrication of major parts. The CNC Machining process is a metallic process where a block or billet of metal is machined into the correct shape using milling and drilling machines. A detailed list of all their constituent tasks, how they are modeled, and the relevant modeling assumptions are described in section A.3. The relevant If-Then precedence rules for each process are listed in Appendix C. More specific modeling details for the CNC Machined process are described in Appendix D. The equipment space, equipment cost, and factory space modeling details and assumptions for all the manufacturing processes are in Appendix E.

There are a few different reasons these specific processes were chosen to be a part of the F-86 experimental platform and analyzed in Experiment 1. One of the primary ones is that the HLU/Autoclave, VARTM, and CNC Machining processes are all very well documented in previous MInD works and so are easily implementable. SRI is very similar to VARTM and so, given a few changes based on works by Linton et al. [232] and Velicki and Jegley [233], just as easily implementable. The main reason is that each process has a particular characteristic related to the material factors of interest that can be used to show the associated tradeoffs. HLU/Autoclave operates on composite materials and is one of the benchmarks for all composite processes. VARTM operates on composite materials as well and has a more advanced version of integral fabrication for its minor parts. SRI also uses composite materials and is one of the only known processes able to support integral fabrication of major parts, not just minor parts. And CNC Machining operates on aluminum materials, providing a material system contrast with the composite processes and also being the baseline that the aircraft industry has traditionally used for many decades. The tradeoffs that can be made with these four is why they are of interest.

Three different spatial constraint settings will be used to show the effects of having factories of various sizes. The settings are for the factory to have 333,000 ft^2, 500,000 ft^2, and 1,000,000 ft^2 of space. The 1,000,000 ft^2 setting is to represent a very large factory, specifically Boeing's Composite Wing Center in Seattle which has 1.2 million ft^2 of space, rounded down to a whole 1 million ft^2 [234]. The 500,000 ft^2 setting is used to represent an intermediately large factory, specifically Mitsubishi Heavy Industries' Composite Wing Center for the Boeing 787 in Nagoya, Japan [235]. The 333,000 ft^2 setting is used to represent how much space was available at North American Aviation's Downey, California plant to manufacture wingboxes. North American Aviation is the manufacturer of the F-86 and according to the U.S. Centennial of Flight Commission had an F-86 manufacturing plant in Downey before switching over to rocketry [236]. According to legal documents, the Downey plant had 1,000,000 ft^2 of space [237]. The full factory space is dedicated to

the entire aircraft, so the amount of space dedicated to just the wingbox was determined by looking at other aircraft programs and seeing how much space they allocated to the wings. For the Boeing 787 aircraft, Boeing uses about 878,000 ft^2 of space for its fuselage manufacturing [238], Mitsubishi Heavy Industries uses 500,000 ft^2 of space for the 787 wing manufacturing as mentioned above [235], and Boeing uses about 600,000 ft^2 of space for the final assembly line where the wings and fuselage are joined [239]. Taking those assemblies as being the bulk of the aircraft and their space usage as representative of the space usage of all aircraft, the wingbox assembly makes up about one-third of the total space used to manufacture an entire aircraft. As a result, it is estimated that 333,000 ft^2 of space at the North American Aviation Downey plant is devoted to the F-86 wings, hence why it is one of the settings and considered the default one whenever comparisons are made. To simplify the calculation of available space and for demonstration purposes, it is assumed that all of the listed factory space settings are available for manufacturing use and that all of it can be used to manufacture the F-86 wingbox as opposed to further setting aside space for offices, cafeterias, meeting rooms, etc.

The developed framework can freely adjust which assembly sequences to use and which parts to make integrally fabricated or part of a subassembly. However, Oesterle et al.'s implementation is not able to do so and is only able to choose among a set number of assembly alternatives. The assembly sequences for each process, and subsequently which parts are set to be integrally fabricated or part of subassemblies, that represent their assembly alternatives are shown in Figure 5.10. These were chosen specifically to implement at least one each of minor part integral fabrication usage and usage of subassemblies for each manufacturing process and usage of major integral fabrication for the SRI process. However, there is no guarantee that these are necessarily the best assembly sequences or the best combination of integrally fabricated parts and subassemblies. Indeed, there is no way to know without an outside comparison, which is one of the weaknesses of only having set alternatives.

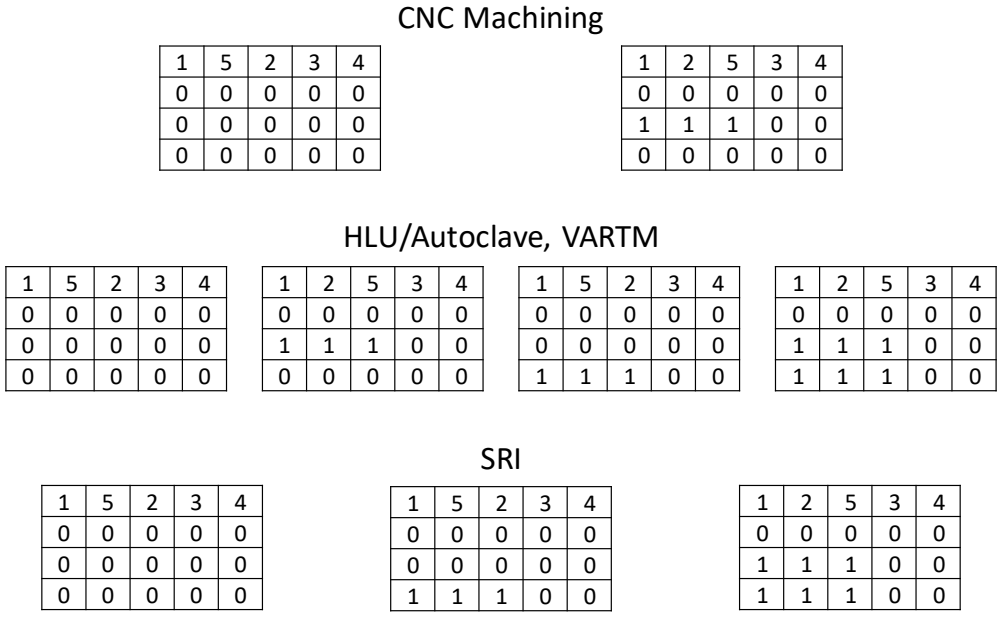


Figure 5.10: Possible Assembly Sequences and Integral Fabrication/Subassembly Part Sets That Represent Different Assembly Alternatives for Oesterle et al.'s GA Implementation for Each Manufacturing Process Considered

Given all the setup described and with all the variables set, the procedure for Experiment 1 is as follows. Oesterle et al.'s NSGA-II implementation is performed for all 4 manufacturing processes and 5 Pareto fronts for each are generated. The developed framework is then used to generate 5 Pareto fronts for all 4 manufacturing processes as well. The two works' results are compared in terms of the quality indicators mentioned earlier as well as on what kinds of tradeoffs they can produce regarding the material factors. Afterwards, the developed framework is used to generate 5 more Pareto fronts for all the manufacturing processes for the different spatial constraints to observe what kinds of tradeoffs can be made when different spatial constraints and factory sizes are used. To see how flexible the developed framework is with its material factors, the developed framework is used to create two more sets of Pareto fronts, one where it is limited to the same sequences as the alternatives set for Oesterle et al.'s implementation and one where integral fabrication and subassemblies are not allowed. These are all again compared with the results from Oesterle et al.'s implementation to see the impacts of not including some of the material

factors. That last comparison marks the conclusion of Experiment 1.

All the assumptions on the F-86 experimental platform are listed out throughout Appendix A. A summary of the major ones were discussed in the previous section. The last notable ones are related to a combination of the material factors, specifically the different processes as well as how integral fabrication applies to them. Structurally, as described in section A.3, due to the lack of available properties for all the different material systems, all the composite processes are assumed to use compatible carbon fiber materials that have roughly the same properties.

It is also assumed that minor part-to-major part integral fabrication, which includes co-bonding and co-curing, is advanced enough to no longer require any fasteners for the HLU/ Autoclave and VARTM processes. It is additionally assumed that for the SRI process, all integral fabrication requires the parts to be stitched together, even minor parts to major parts to add additional fatigue resistance and differentiate it from the other processes. Stitching requires a larger joint width for the stitches to exhibit additional joint strength beyond just the baseline co-cure bonding strength [240]. Thus, to have them be able to replace fasteners for major part integral fabrication and prevent the stitching itself from damaging the plies whenever they are used, all the widths of all the joints on all the parts are increased in size. This makes the wingbox slightly heavier when using the SRI process with the benefit of being able to integrally fabricate parts.

The autoclave and oven curing tasks each contain curing and bagging/debagging activities and they are assumed to be able to be carried out relatively independently. This is described further in section A.4 and especially in section A.5. To summarize, in stations with curing tasks, these activities are accounted for separately and duplicated separately when station duplication is performed. Stations with curing tasks thus have two different effective station times, one for each activity, with the effective station time that determines whether the curing station is a bottleneck being the larger of the two activities' effective station times. This is done for two reasons. The first is that the physical process of curing

takes 360 minutes for the autoclave cure tasks and 148 minutes for the oven cure tasks and does not change regardless of how many parts are cured simultaneously in them, making the curing activity completely independent from the bagging activity; it therefore does not make sense to duplicate the curing activity if the bagging and debagging activities are causing the bottleneck. The second is that the autoclaves and ovens are some of the single most expensive piece of equipment. As such, it makes even more sense to delineate them from the bagging and debagging activities so that they are duplicated as seldomly as possible, which is also what is done in real life factories.

It is assumed that the assembly line is completely pulsed, with all parts moving to the next station after an amount of time equal to the cycle time, and that the time spent moving the parts to the next station is insignificant. To do otherwise would introduce an immense amount of complexity for little payoff because accounting for the movement times is not needed to capture and demonstrate this work's desired tradeoff capabilities. This is thus a prime topic for future works after the desired tradeoff capabilities have already been obtained and showcased. Subsequently, as a result of this assumption, the automated ground vehicles used to transport the parts between stations are not modeled and their costs not accounted for either.

5.5.4 Results and Discussion

The hypervolume ratios and the average number of generations required to generate each set of Pareto fronts for Oesterle et al.'s implementation and the developed framework is shown in Table 5.2. The average summary attainment surfaces obtained by Oesterle et al.'s implementation and the developed framework are shown in Figure 5.11 for all the manufacturing processes. The best summary attainment surfaces achieved by both are shown in Figure 5.12. The distribution of $I_\epsilon(PF_1, PF_2)$ values when comparing the two implementations are shown in Figure 5.13 in the form of box plots. In all the figures, OI refers to the results obtained by Oesterle et al.'s implementation and DF refers to the results

obtained by the developed framework. The perfectly vertical and horizontal lines in the summary attainment surfaces are simply artifacts of using summary attainment surfaces and just visualize the area enclosed by the non-dominated solutions; there are no actual solutions that yield infinite cost. The spatial constraint here is that the factory is 333,000 ft^2 in size.

Table 5.2: Hypervolume Ratio and Average Number of Generations for Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework for All Processes

Manufacturing Process	**OI HVR**	**OI Average Number of Generations**	**DF HVR**	**DF Average Number of Generations**
HLU/ Autoclave	0.191	250	1.00	415
VARTM	0.148	250	1.00	459
SRI	0.172	250	1.00	422
CNC Machined	0.219	250	1.00	408

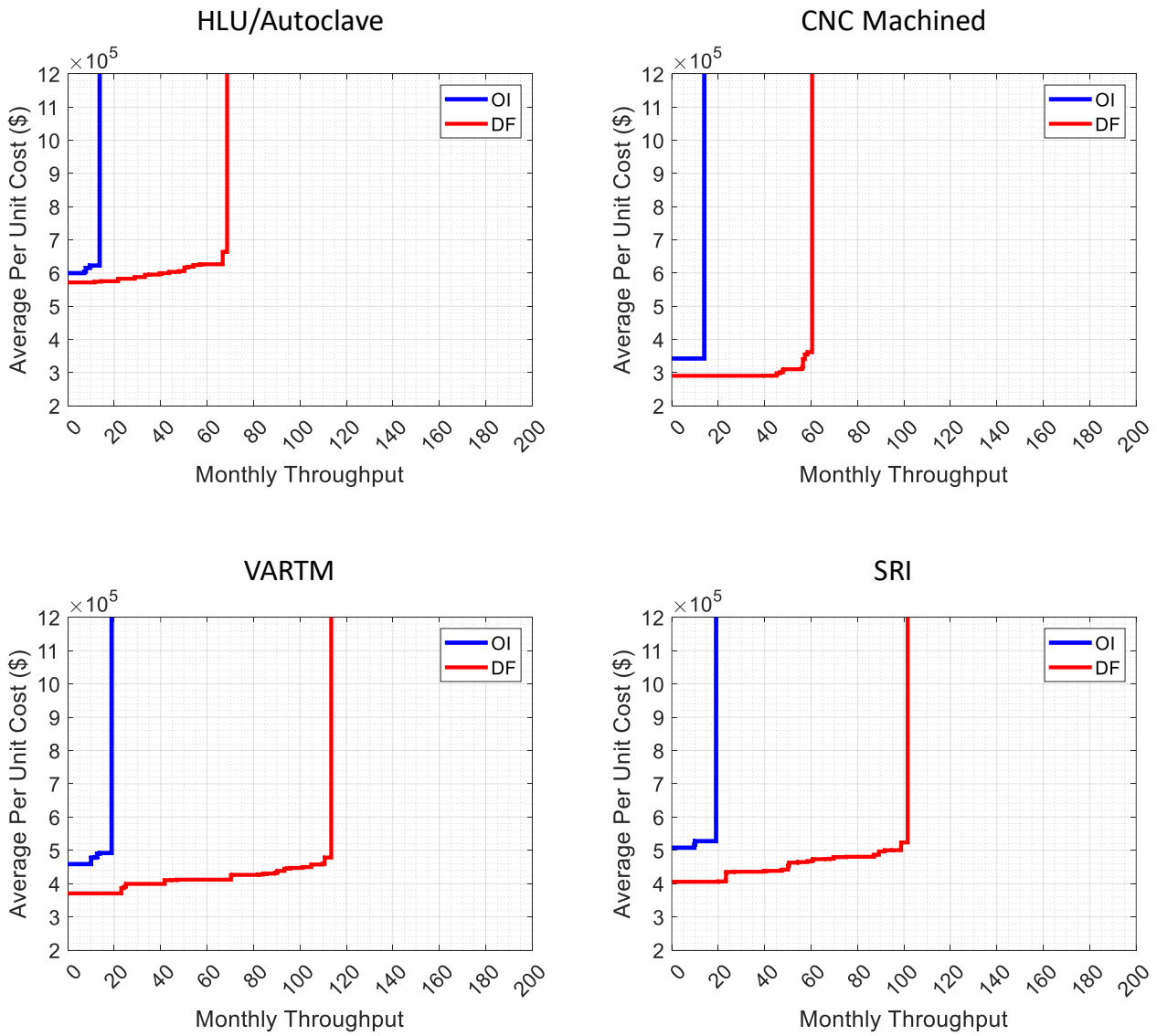


Figure 5.11: Average Summary Attainment Surfaces Obtained by Oesterle et al.'s Implementation and Developed Framework for All Processes

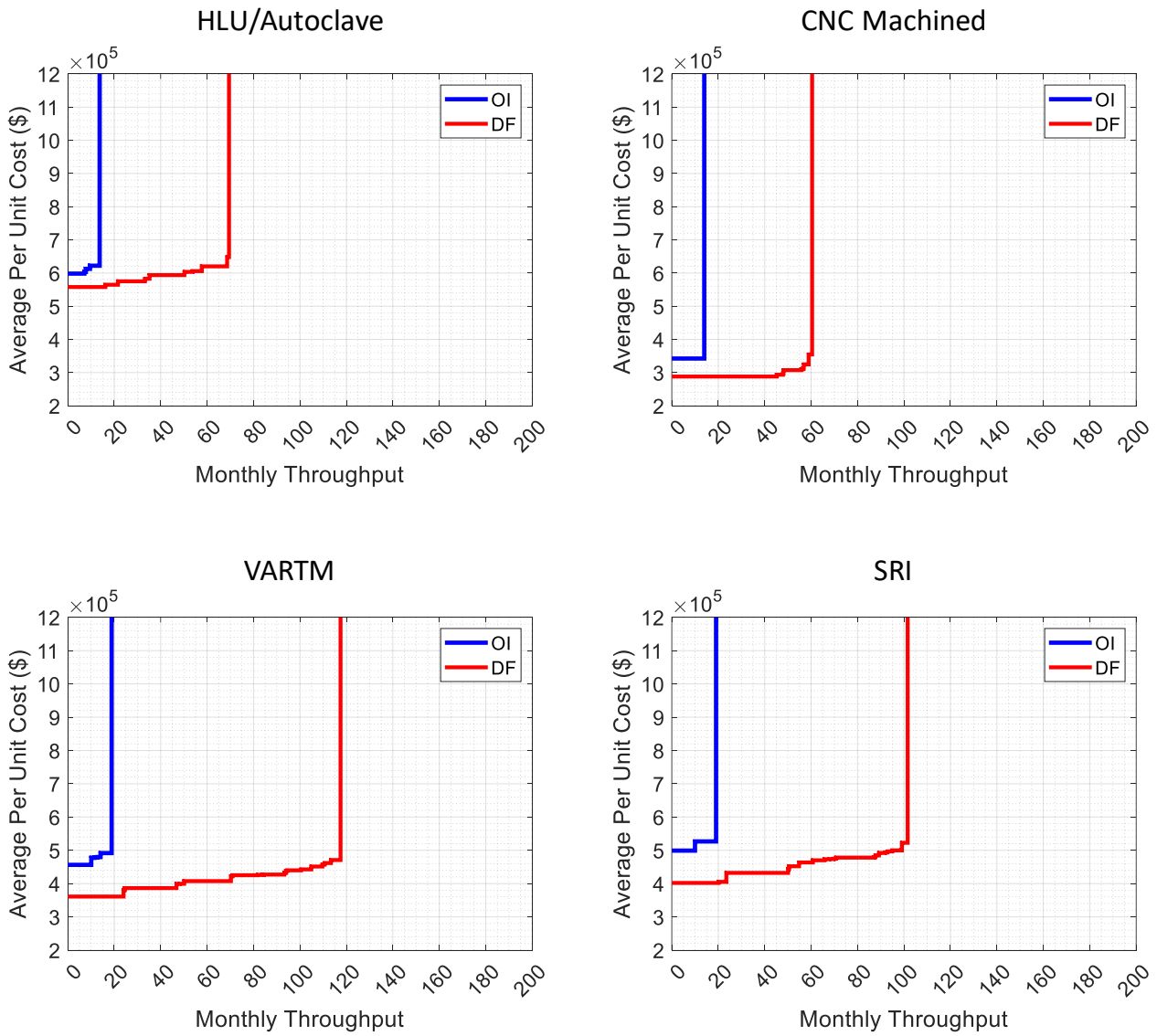


Figure 5.12: Best Summary Attainment Surfaces Obtained by Oesterle et al.'s Implementation and Developed Framework for All Processes

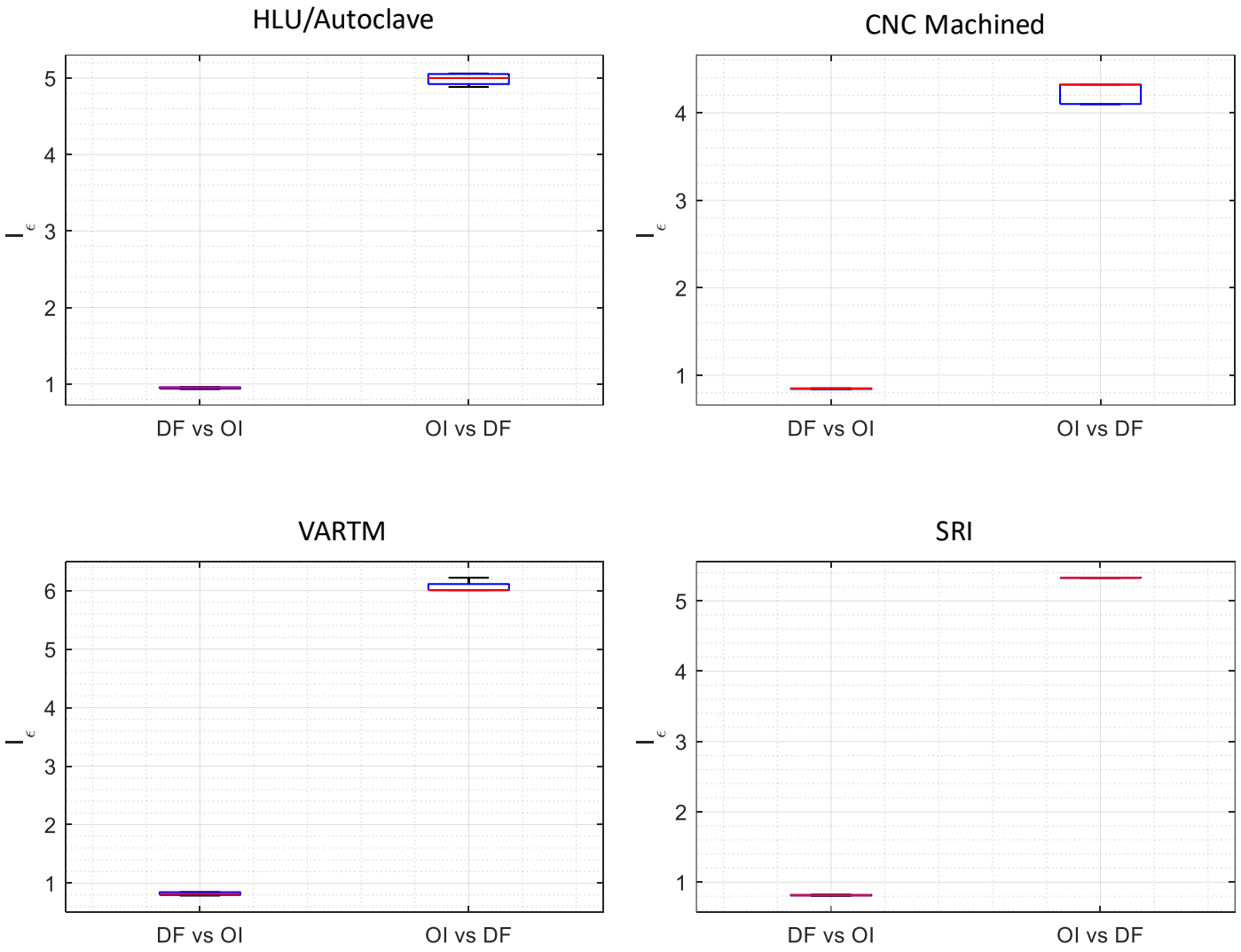


Figure 5.13: Boxplots Showing $I_\epsilon(PF_1, PF_2)$ Values Obtained by Comparing Oesterle et al.'s Implementation and Developed Framework for All Processes. Lower Values Show Better Performance

As is made very apparent by examining all of the table and all the figures, the framework developed specifically to tackle all of the different material factors generates solutions that completely dominate those generated by Oesterle et al.'s NSGA-II implementation, which represents the current state-of-the-art in terms of purely ALB and ASP-focused works that have considerations for the material factors. The summary attainment surfaces generated by the developed framework completely encapsulate Oesterle et al.'s. The $I_\epsilon(DF, OI)$ values are less than 1 and the $I_\epsilon(OI, DF)$ values are greater than 1 while the HVR values of all the developed framework's Pareto fronts are equal to unity, mathematically confirming that this is the case as opposed to just visually. This is true for all the different processes and demonstrates that the developed framework can generate designs that yield lower APUC and significantly higher production rates. Though it does take about double the time to calculate, as indicated by Table 5.2, it can be seen that the developed work's non-dominated solutions' throughputs are more than double that obtained with Oesterle et al.'s implementation, which offsets the longer wait.

However, as discussed before, it is not sufficient just to show that a framework or implementation is able to achieve better values–it must show why it is able to obtain those better values to provide justifiability and data traceability for those values. This instills confidence that those values are based on logic and so can be trusted. Diagnostic results used to support why the above Pareto fronts look the way they do are thus presented. Figure 5.14 show how much space the solutions from each method use as a function of the production rate constraints. Figure 5.15 shows a box plot distribution of the average station utilization ratio for each solution for both methods. The station utilization ratio is essentially an alternative way to demonstrate line efficiency: it is the effective station time divided by the cycle time and indicates what percentage of the cycle time was spent performing useful work at every workstation. Ratios closer to unity indicate more efficient stations and line balances. Figure 5.16 shows the average number of station duplications in each solution as a function of production rate. It gives an idea on the trend regarding how many duplications are needed

at every throughput level and, thus, how the space available is utilized. Figure 5.17 depicts how many minor and major parts are integrally fabricated or are part of a subassembly as a function of production rate. All minor parts by default are integrally fabricated whereas all major parts are in a subassembly save for the SRI process, where combinations of the two are possible. Figure 5.17 therefore shows how optimal integral fabrication and subassembly usage changes, if it changes at all, based on the desired throughput level.

Figure 5.18 shows the average number of tasks in the list of available tasks. Figure 5.19 depicts the average task time and Figure 5.20 depicts the average number of stations in each solution. Figure 5.21 and Figure 5.22 show how the capital costs, which are the combined equipment and tooling costs, and labor costs fluctuate as a function of production rate, respectively; these values are for the entire production run and are not on a per unit basis. And, since the design is not being varied and it only changes due to a change in manufacturing process, the total material cost for the HLU/Autoclave process is $329 million, for the VARTM and SRI processes is $235 million, and for the metallic process is $52.8 million.

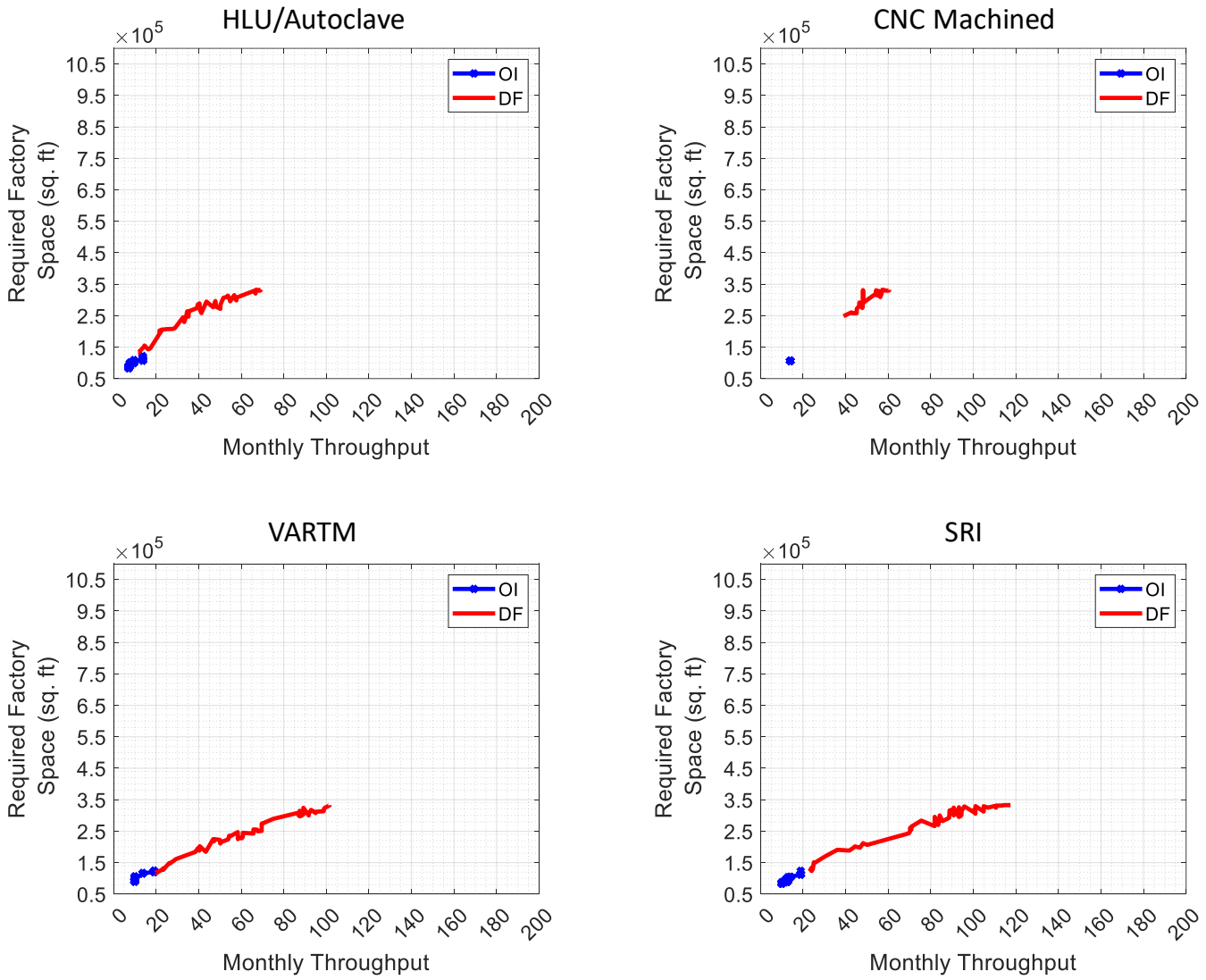


Figure 5.14: Space Used in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework as a Function of Throughput for All Processes

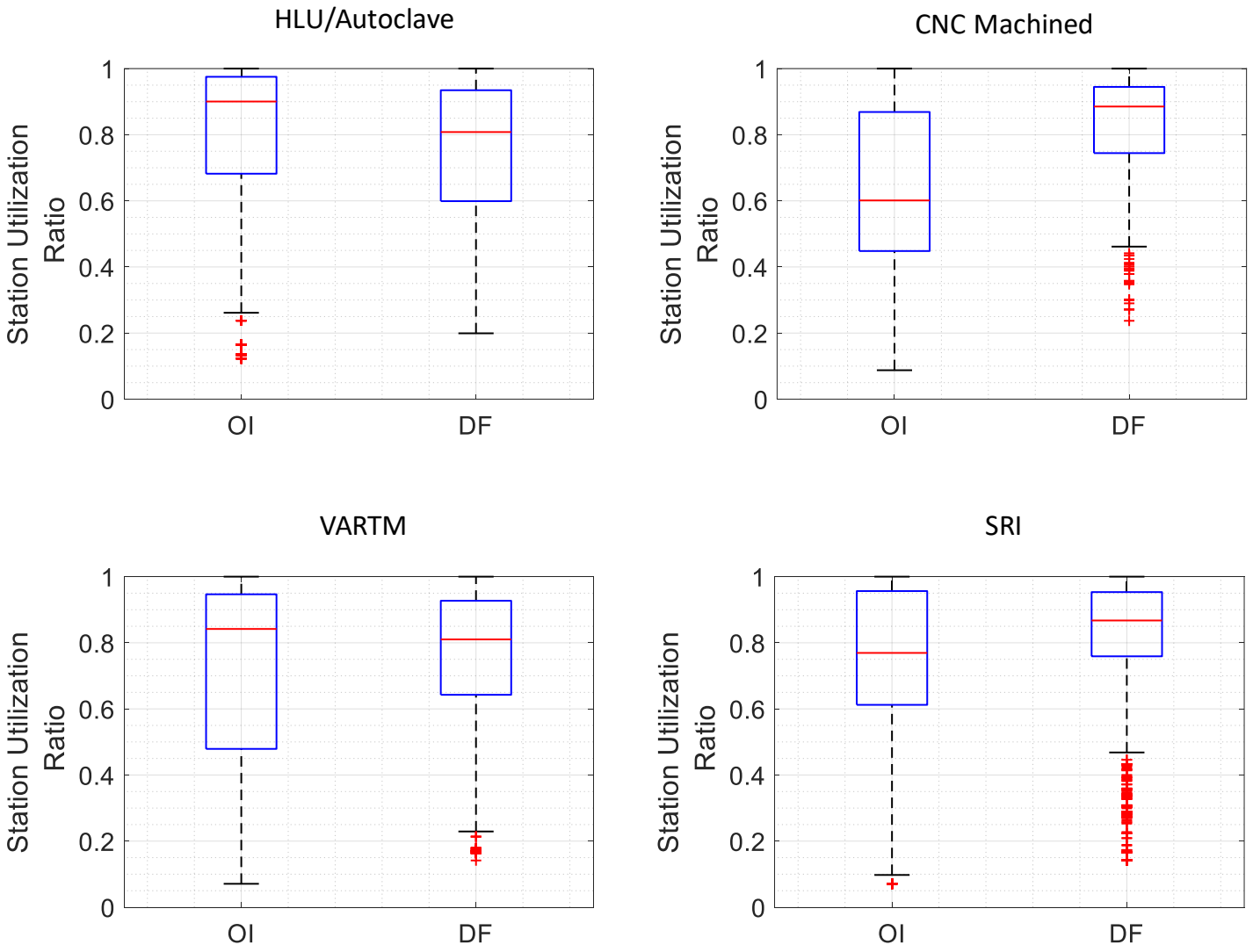


Figure 5.15: Boxplots Showing Distribution of Average Station Utilization Ratio in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework for All Processes

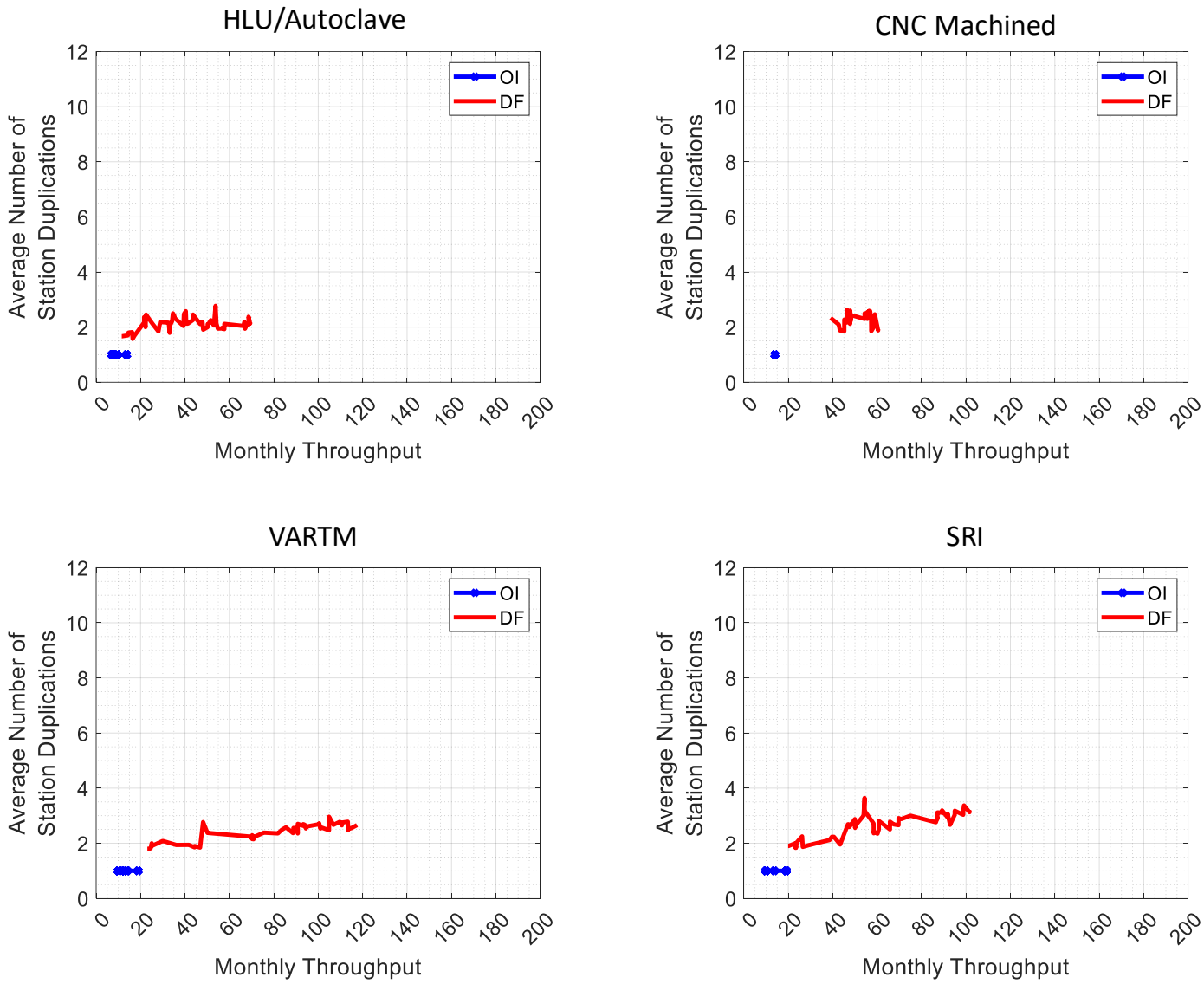


Figure 5.16: Average Number of Station Duplications as a Function of Throughput in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework for All Processes

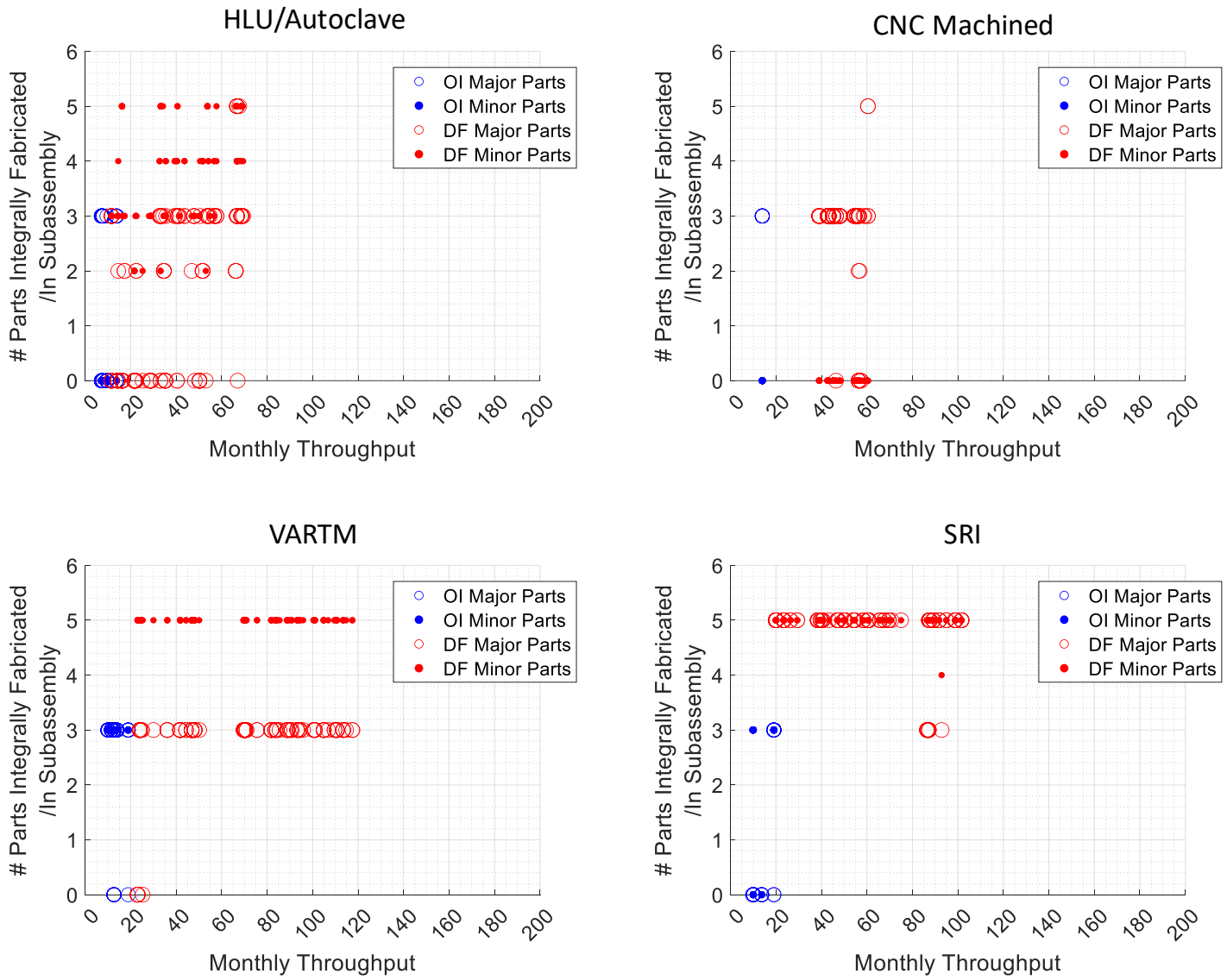


Figure 5.17: Number of Minor and Major Parts Integrally Fabricated or in a Subassembly as a Function of Throughput in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework for All Processes

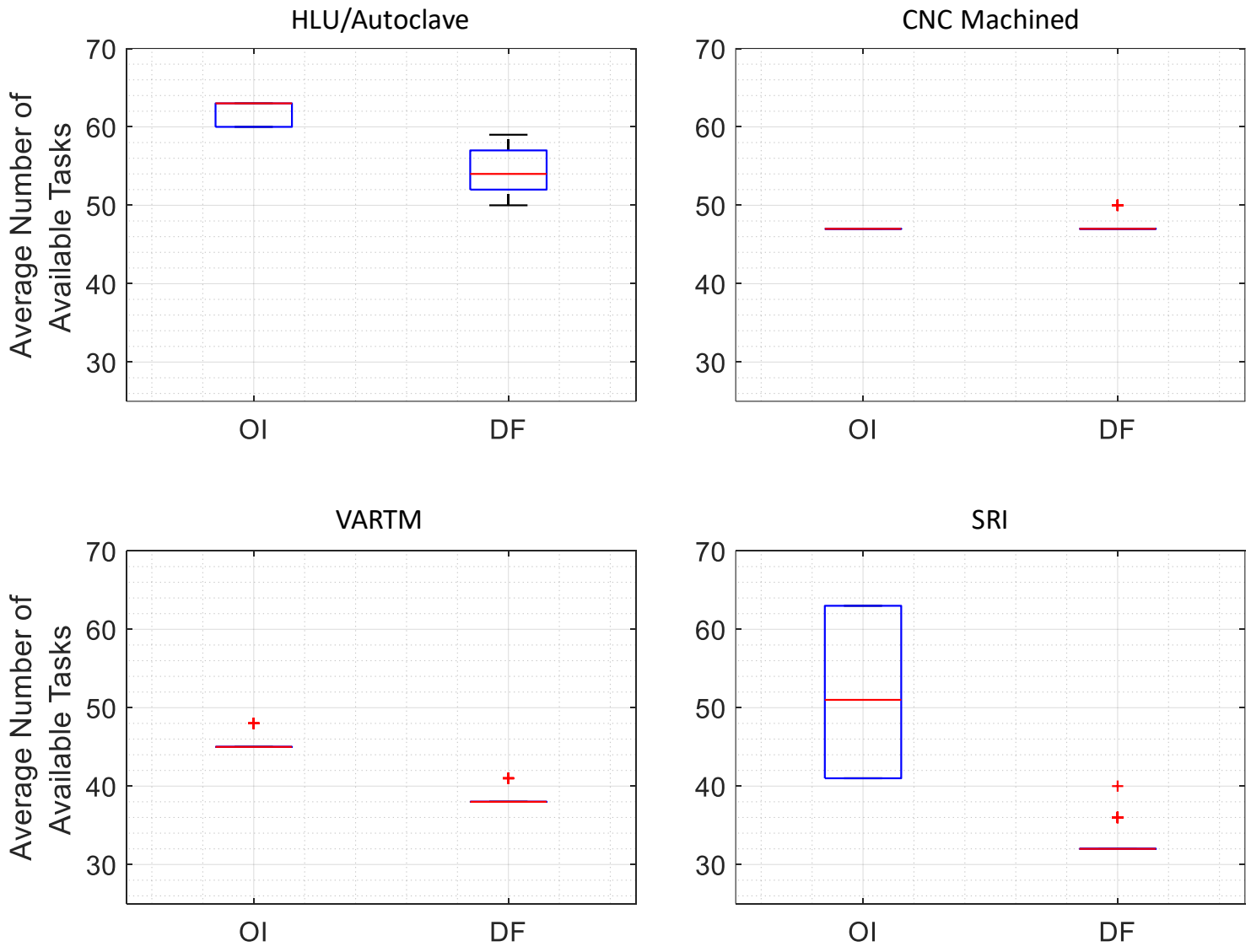


Figure 5.18: Boxplot Distribution of Number of Available Tasks in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework for All Processes

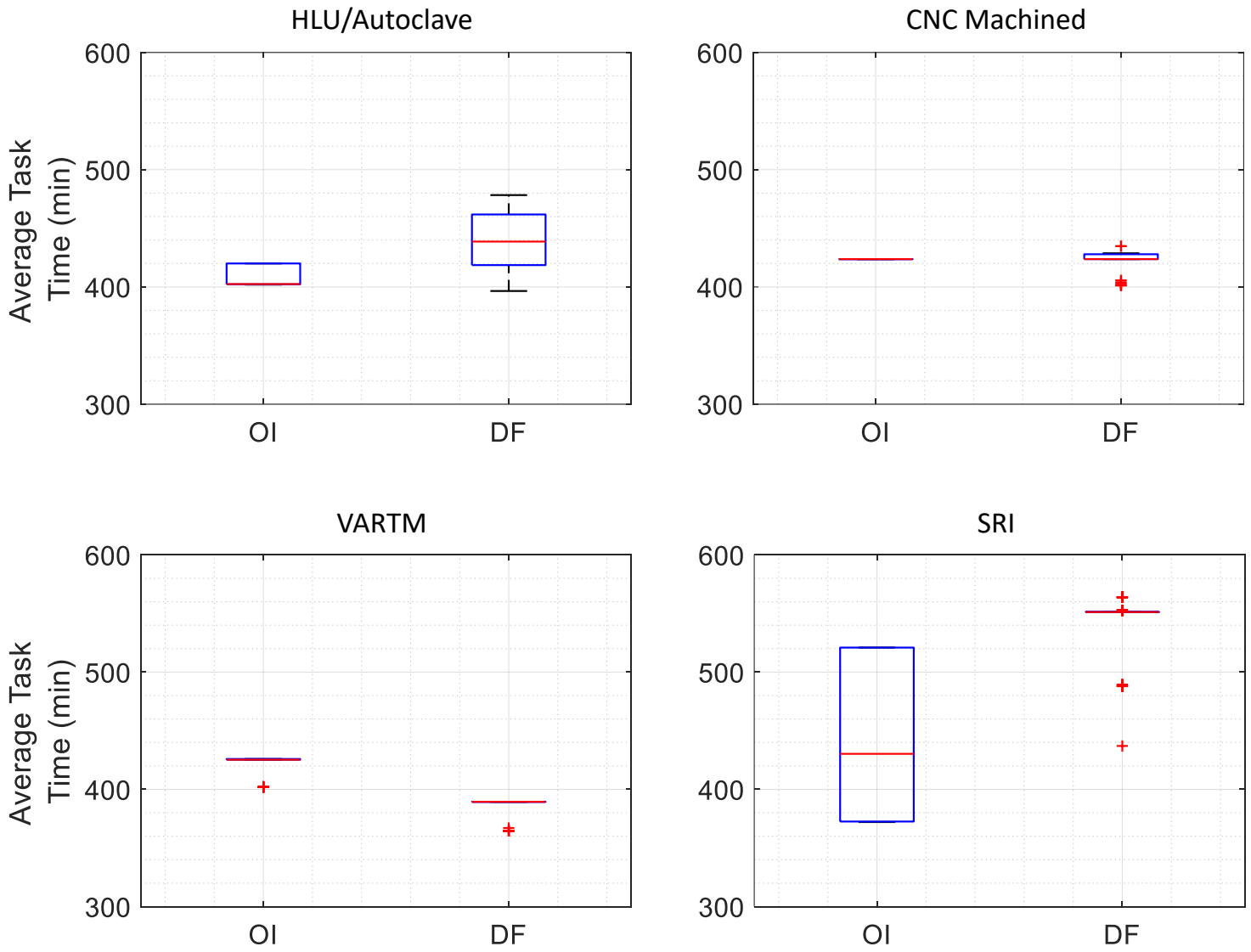


Figure 5.19: Boxplot Distribution of Average Task Times in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework for All Processes

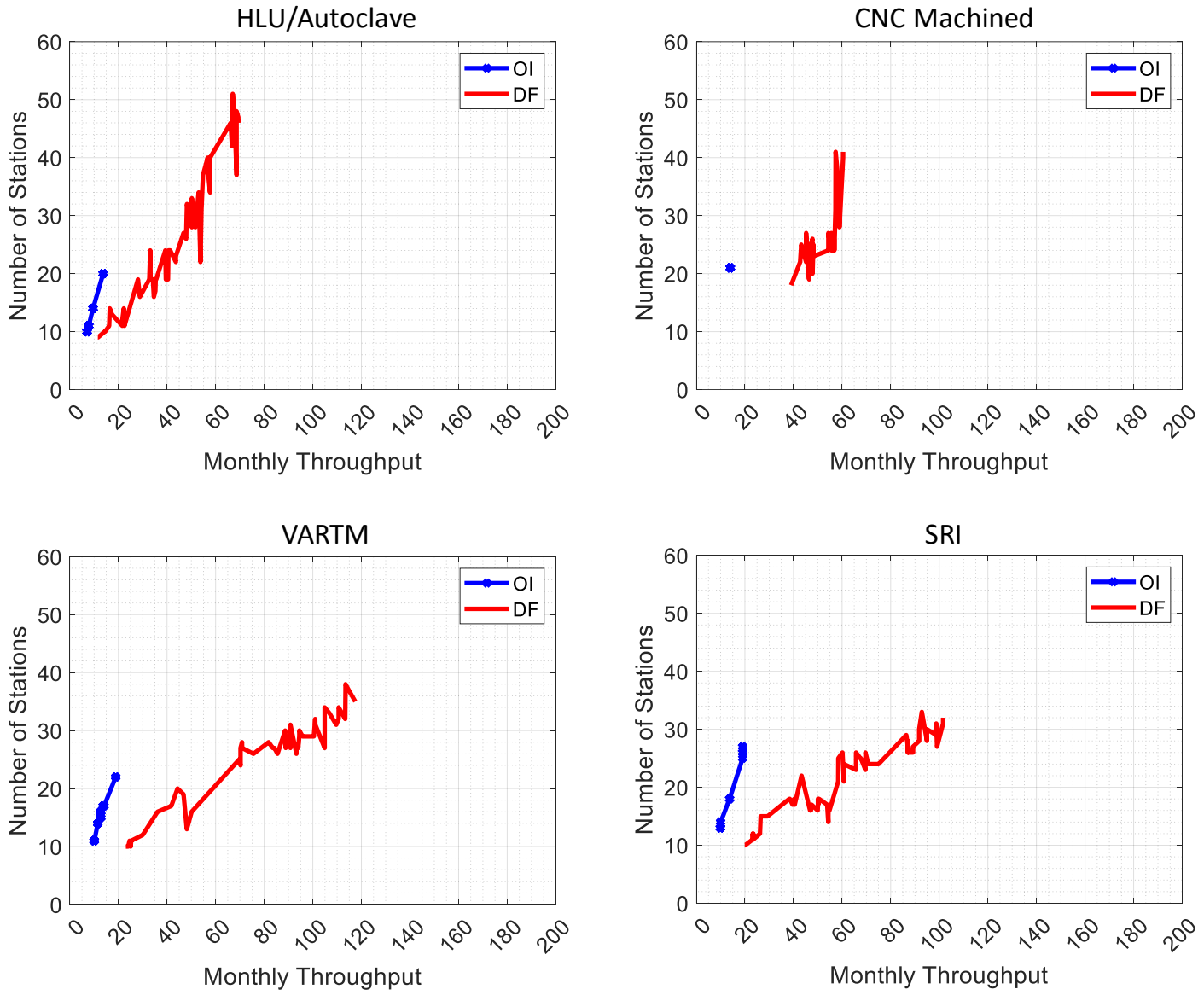


Figure 5.20: Number of Stations in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework as a Function of Throughput for All Processes

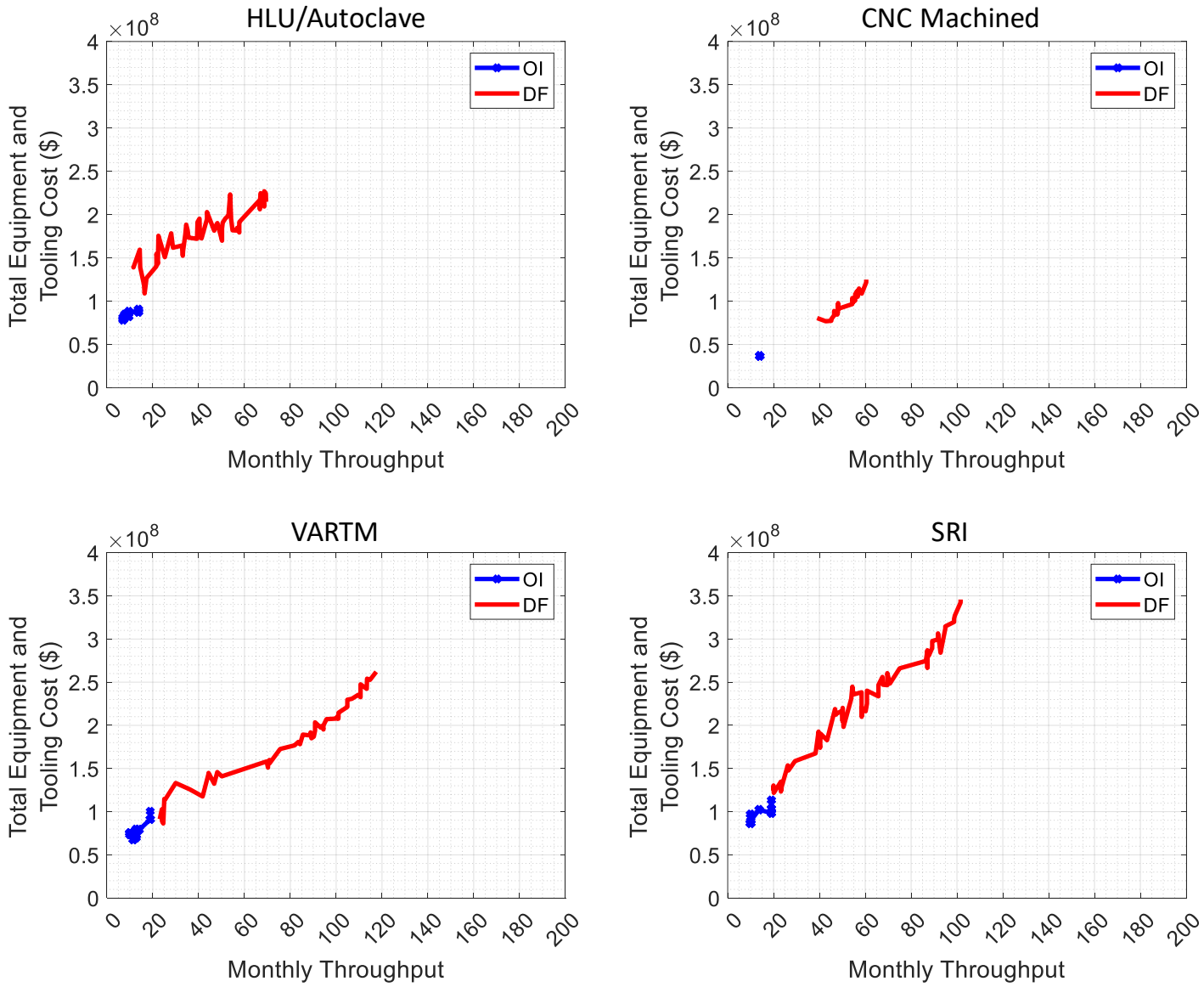


Figure 5.21: Combined Equipment and Tooling Cost in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework as a Function of Throughput for All Processes

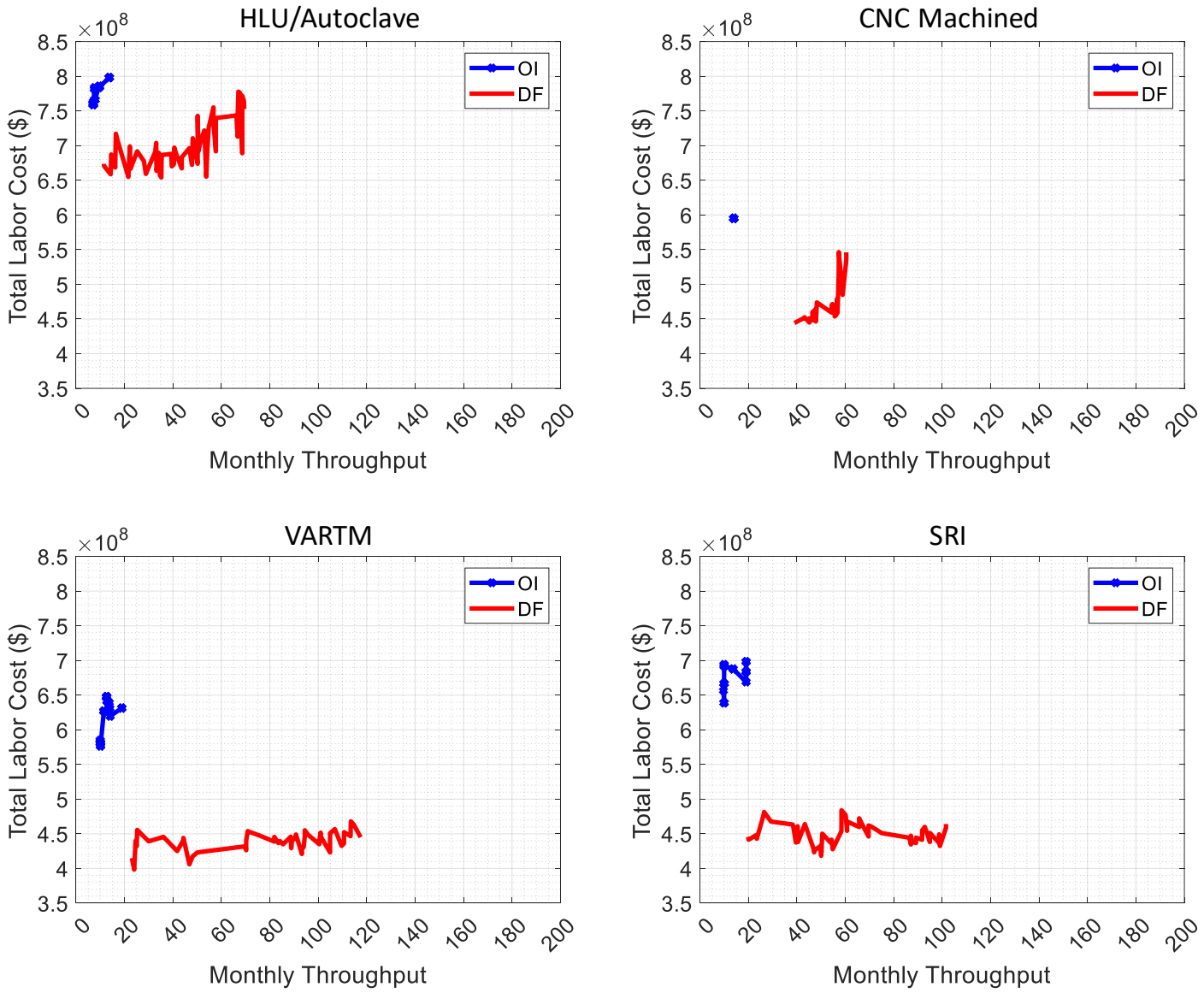


Figure 5.22: Total Labor Cost in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework as a Function of Throughput for All Processes

Based on Figure 5.17, it can already be seen that one of the main reasons Oesterle et al.'s implementation's results do not have as low a cost or as high a production rate is that the assembly sequences, integral fabrication, and subassembly part sets available to it as assembly alternatives do not represent the best ones available. For the VARTM and HLU/Autoclave processes, the ideal assembly sequences had more minor part integral fabrication, and for the SRI process, there was more major part integral fabrication and subassembly usage in the best assembly sequences as well. It results in what can be seen in Figure 5.18, where the developed framework consistently has an equal or lower number of available tasks that it must assign to stations, which helps trend to a lower labor cost in Figure 5.22 due to requiring less workers. This demonstrates the limitation of solely relying on expert-provided assembly sequences and integral fabrication part sets and subassembly part sets: while the expert may be able to get the exact best combination, there is a very high chance they will not, resulting in sub-optimal sequences and part combinations. This is as opposed to the developed framework that evaluates an enormous number of sequences and combinations on its own and can independently come up with an optimal sequence and combination and, through these diagnostic results that the developed framework is also able to generate, show why.

The spatial constraints material factor has perhaps the most notable and direct effect on why the developed framework's results are so dominating. As can be seen in Figure 5.14, the developed framework is able to fully utilize the space available to it to at least improve production rate and is able to do so via station duplication, which Oesterle et al.'s implementation cannot do as shown in that figure and in Figure 5.16. This has a knock-on effect not only on throughput, where duplicate stations decrease effective station time and decrease cycle time and improve production rate, but on labor cost as well. Station duplication allows station workloads to be more evenly distributed and improve their efficiency, which reduces cost, demonstrated by the fact that even as more duplication and space is used to increase throughput, total labor costs stay comparatively stagnant in Figure 5.21

despite capital costs rising dramatically in Figure 5.21. This is emphasized even more in Figure 5.15 where it can be seen that the lack of station duplication causes Oesterle et al.'s implementation's solutions to almost always have lower station and labor efficiency than the developed framework's solutions.

Oesterle et al.'s implementation has comparable capabilities to the developed framework when looking at manufacturing processes as both sets of solutions show the same trends. The VARTM process is able to achieve the highest throughput due to a combination of one of the lower capital costs, seen in Figure 5.21, moderate labor costs as seen in Figure 5.21, and the lowest average task times as shown in Figure 5.19. The latter point complements the fact that higher throughput levels tend to favor stations with very few tasks, as seen in Figure 5.20. Given the VARTM process's lower task times it is ultimately able to have more duplications of stations with only a single task in them, evident in Figure 5.16, and uses that to obtain higher throughputs than all the other processes. Its having lower capital equipment costs than the SRI indicates that the advantages of stitching for manufacturing do not outweigh the throughput penalties of needing a stitching machine. Meanwhile, based on all those figures, the CNC Machining process has comparable labor and equipment costs to the VARTM process at low throughput levels and is able to have the lowest APUC in Figure 5.11 and Figure 5.12 because it has lower material costs. These trends are as evident in Oesterle et al's solutions as in the developed framework's, meaning that though the former is unable to obtain results of the same quality, it can at least indicate the manufacturing process trends.

The developed framework thus far has shown it is able to obtain better results due to better consideration of the material factors. It has also shown it is able to trace the reasons why its results are better back to the material factors used. However, part of Experiment 1 is to demonstrate that the developed framework better accommodates the material factors by being able to directly alter them to make further tradeoffs, as opposed to just passively including them for data traceability or just setting them as constraints. This is to better show

that the developed framework not only includes the material factors but has the capability to actively manipulate them to study what effects they may yield. The purpose is to further cement that there is a significant gap in handling material factors between the current state-of-the-art and the developed framework due to the former not being able to make any of these material factor changes. Therefore, two additional sets of results are presented, one where spatial constraints of 500k ft^2 and 1M ft^2 are examined alongside Oesterle et al.'s implementation's results to see how they compare, and one where the developed framework is either limited to the same sequences and part combinations as Oesterle et al.'s implementation or is unable to use any integrally fabricated parts or subassemblies in its assembly sequences.

The set of results where additional spatial constraints are considered is examined first. All 4 manufacturing processes are examined in all the following figures. Figure 5.23 contains the average summary attainment surfaces for both sets of spatial constraint's solutions along with Oesterle et al.'s implementation's solutions for comparison. The best summary attainment surface is not shown since it is already obvious the developed framework's results will dominate Oesterle et al.'s by a large margin and so is redundant. Plots showing the distribution of $I_\epsilon(PF_1, PF_2)$ values are not shown for similar reasons. In the legend, OI still stands for Oesterle et al.'s implementation while DF1 refers to the developed framework's solutions that have the 500k ft^2 constraint and DF2 refers to the developed framework's solutions that have the 1M ft^2 constraint. Figure 5.24 depicts the station utilization boxplot distribution. Figure 5.25, Figure 5.26 and Figure 5.27 show how the number of station duplications, number of stations, and factory space required, respectively, vary as a function of the throughput level. Figure 5.28 and Figure 5.29 visualize how the capital and labor costs respectively vary as a function of the production rate. The average number of tasks, task times, and number of parts integrally fabricated or in subassemblies is not shown since it is almost identical to the initial set of results.

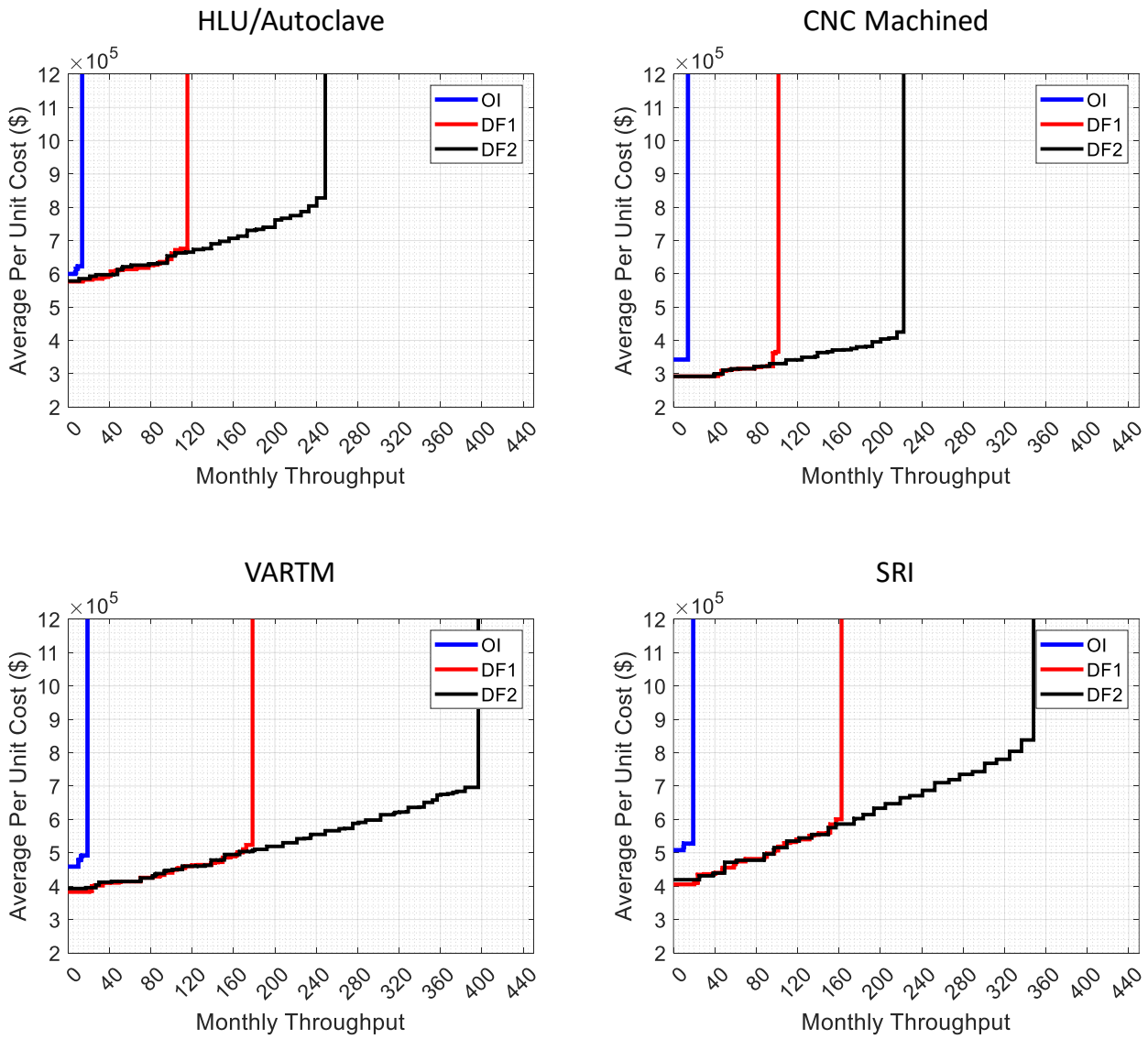


Figure 5.23: Average Summary Attainment Surfaces Obtained by Oesterle et al.'s Implementation and Developed Framework for All Processes, DF1 is 500k ft^2 Constraint and DF2 is 1M ft^2 Constraint

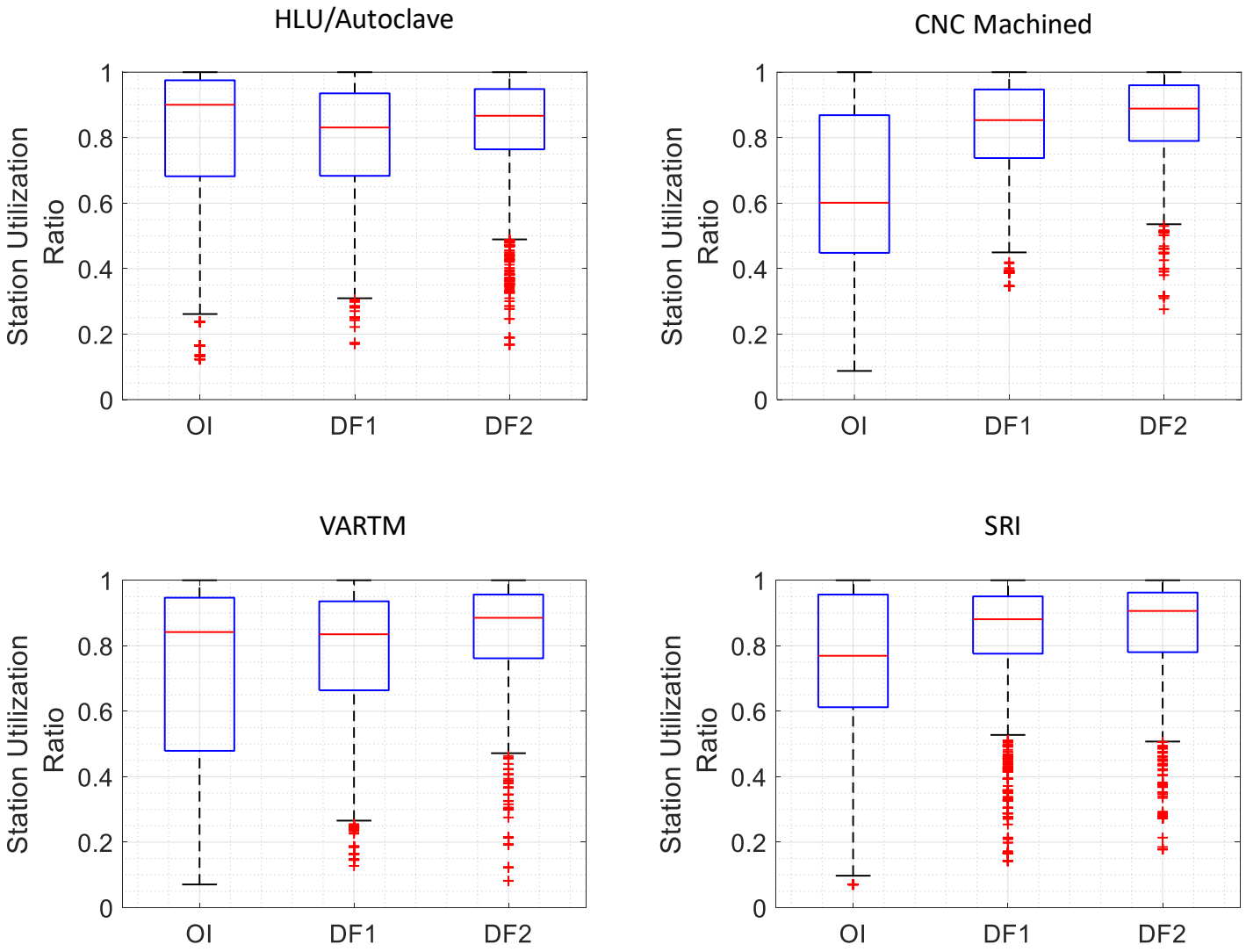


Figure 5.24: Boxplots Showing Distribution of Average Station Utilization Ratio in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework for All Processes, DF1 is 500k ft^2 Constraint and DF2 is 1M ft^2 Constraint

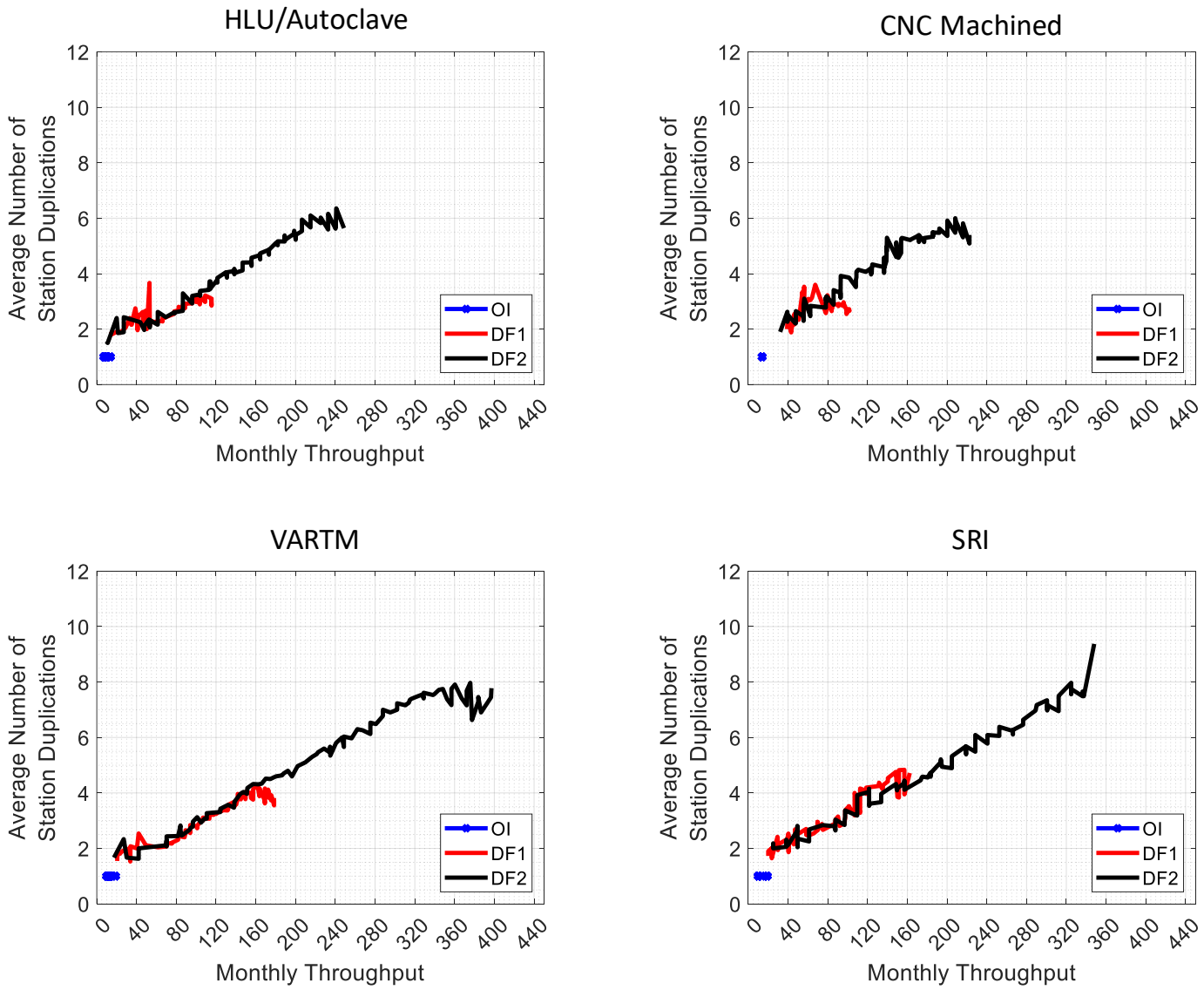


Figure 5.25: Average Number of Station Duplications as a Function of Throughput in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework for All Processes, DF1 is 500k ft^2 Constraint and DF2 is 1M ft^2 Constraint

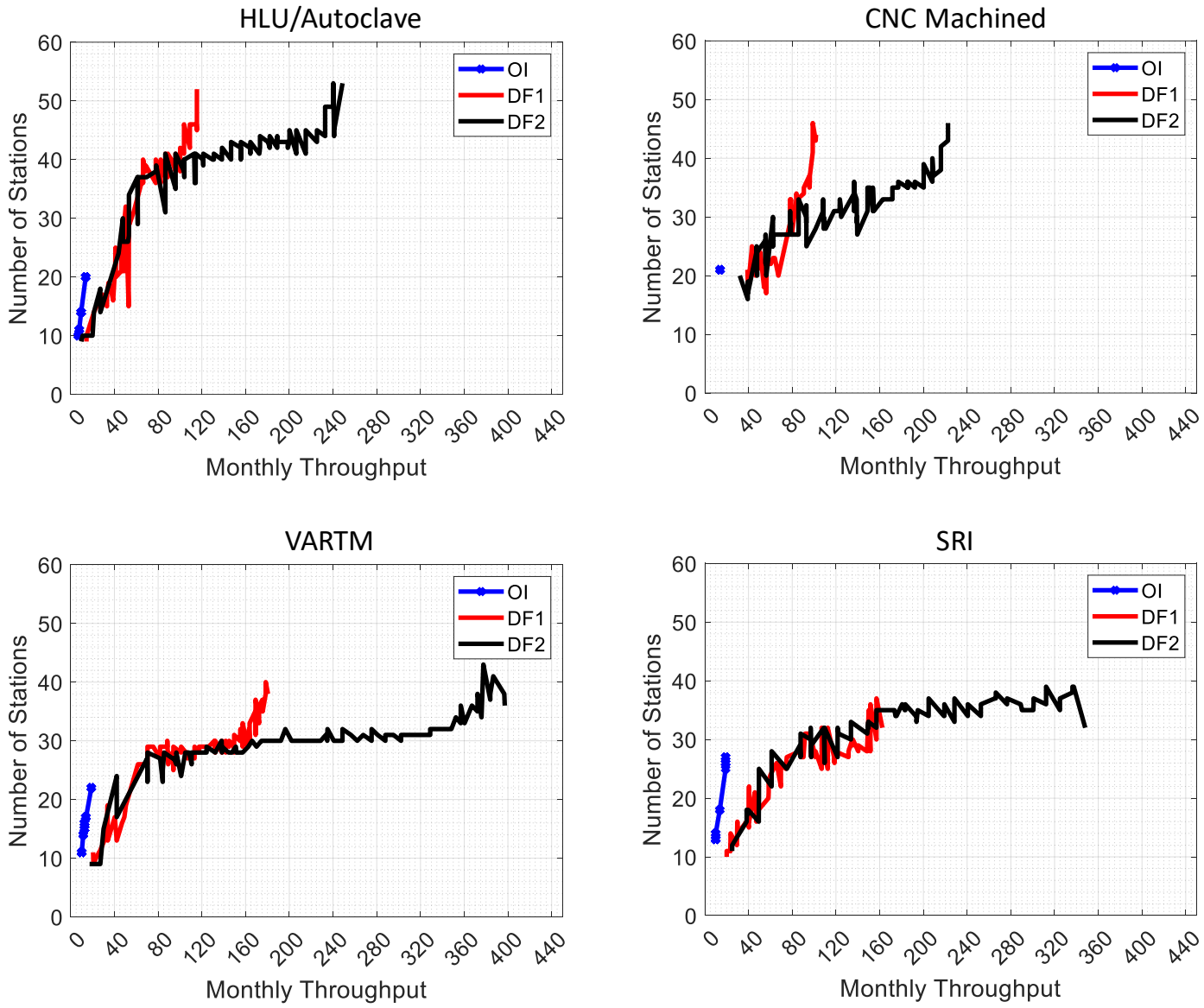


Figure 5.26: Number of Stations in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework as a Function of Throughput for All Processes, DF1 is 500k ft^2 Constraint and DF2 is 1M ft^2 Constraint

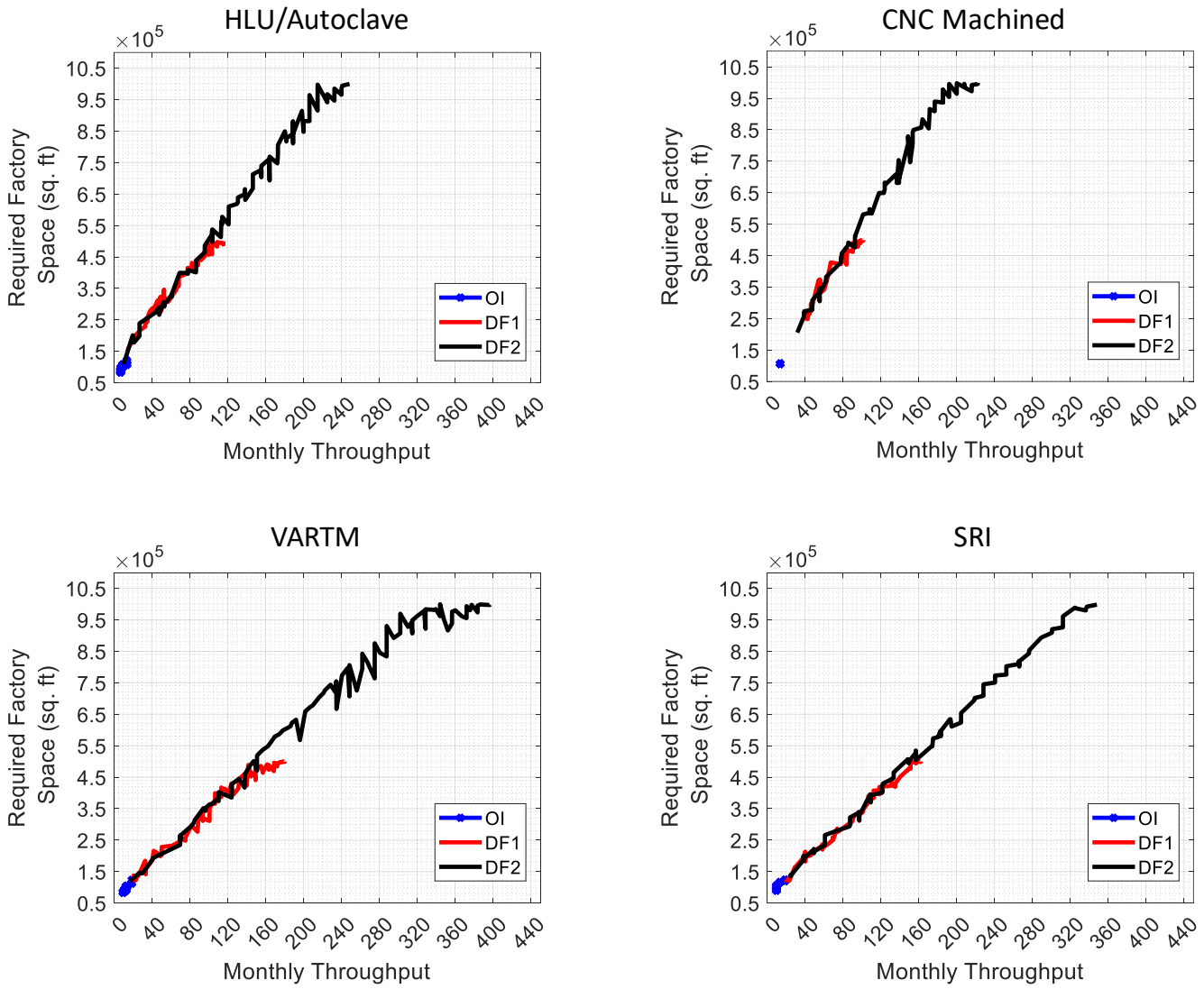


Figure 5.27: Space Used in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework as a Function of Throughput for All Processes, DF1 is 500k ft^2 Constraint and DF2 is 1M ft^2 Constraint

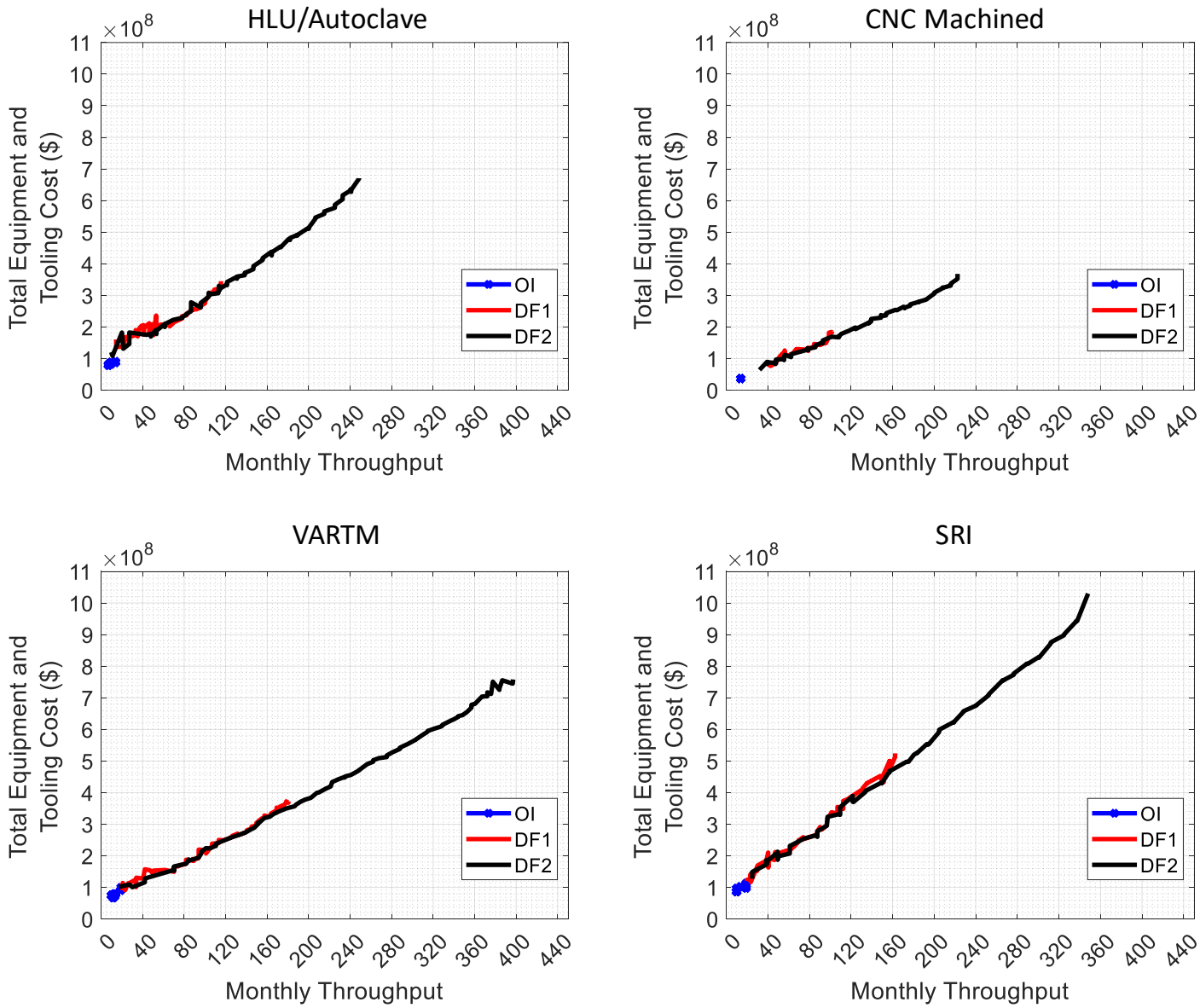


Figure 5.28: Combined Equipment and Tooling Cost in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework as a Function of Throughput for All Processes, DF1 is 500k ft^2 Constraint and DF2 is 1M ft^2 Constraint

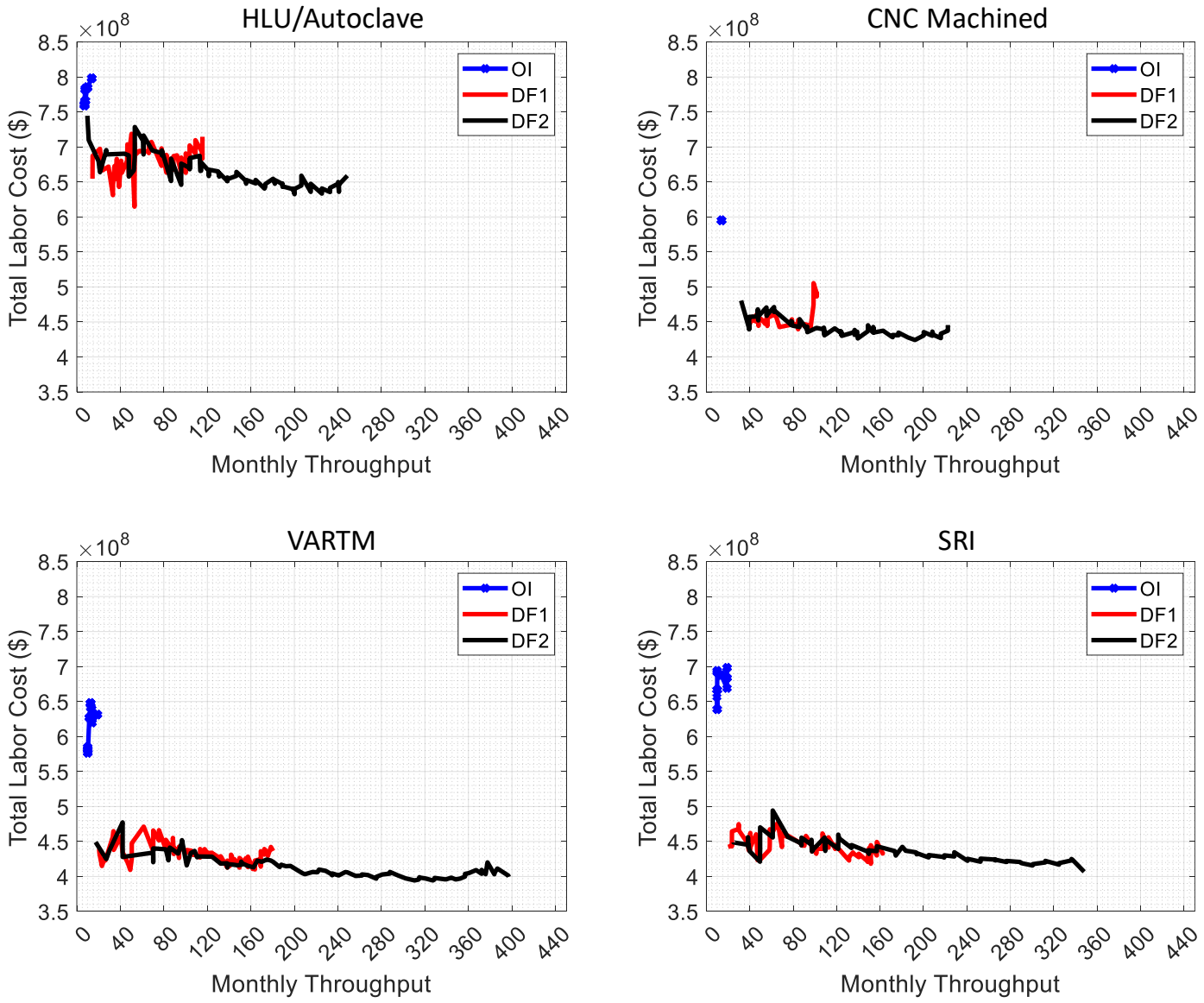


Figure 5.29: Total Labor Cost in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework as a Function of Throughput for All Processes, DF1 is 500k ft^2 Constraint and DF2 is 1M ft^2 Constraint

As can be seen in Figure 5.23 and Figure 5.27, larger amounts of available space allow for that space to be used to increase production rate, with double the available space leading to roughly double the throughput but, surprisingly, less than double the cost. This is better seen in Figure 5.29 where labor cost is shown to actually decrease as throughput is increased despite capital costs in Figure 5.28 increasing relatively linearly. This is because, despite hiring more workers to fill up the increased number of station instances, shown by Figure 5.25, the workers only work until the production run ends. With the production rate being doubled, the production run ends in half the time, leading to an otherwise net zero increase in labor cost. In other words, the product of all the variables in Equation 5.8 except the number of units in the production run stays relatively constant, causing the labor cost to not change drastically despite the number of workers going up. The labor cost ends up actually decreasing because more station duplications from further spatial availability allows for even more efficient station utilization, seen in Figure 5.24 where all processes are more efficient from having more space. That reduces overall labor cost since less labor costs are spent on idle time.

Another interesting trend that can be identified from changing the spatial constraints is that in Figure 5.25 and Figure 5.26 at the respective highest throughput for DF1 there is an abrupt jump in the number of stations and an abrupt drop in station duplications while there are no such abrupt changes for DF2 at those locations, despite the two closely tracking until that point. There is a similar jump in cost for DF1 at its highest throughput in Figure 5.23 before its summary attainment surface goes vertical as compared to DF2 at roughly the same throughput. This occurs because, at that throughput, the DF1 solutions have no more space to linearly increase both throughput and cost at the same rate; steady cost increases are sacrificed for the sake of throughput when space starts to run out, which is done by having as many separate stations in the line balance as possible. A similar trend can actually be seen in the DF2 solutions close to their max throughputs in Figure 5.25 and Figure 5.26 as well and indicates to designers that, when they are running out of space, further increases

in production rate could result in exponential cost increases. The identification of these trends would be much more difficult, if not impossible, if the ability to change spatial constraints did not exist since the trends were only possible to identify when comparing results with different constraint values. It is important to not just be able to consider spatial constraints but also to be able to change them because the former may not provide the full picture, as was just shown. The same can be said for the other material factors as well.

The set of results where changes to the assembly sequences and integral fabrication/ subassembly combinations are made are examined next. All 4 manufacturing processes are examined in all the following figures. In the legend for these results, OI still stands for Oesterle et al.'s implementation's results while DF1 refers to the developed framework's solutions that are limited to the same assembly sequences as Oesterle et al.'s implementation's assembly alternatives. DF2 refers to the developed framework's solutions that are restricted to not using any integral fabrication or major part subassemblies at all. Figure 5.30 contains the average summary attainment surfaces for both sets of spatial constraint's solutions along with Oesterle et al.'s implementation's solutions for comparison. The best summary attainment surface is not shown since it is almost identical to the average one. Plots showing the boxplot distribution of $I_\epsilon(PF_1, PF_2)$ values are in Figure 5.31. The number of parts integrally fabricated or in subassemblies is in Figure 5.33. Figure 5.32 depicts the station utilization boxplot distribution. Figure 5.36 and Figure 5.37 depict a boxplot distribution of the average number of available tasks and task times. Figure 5.34, Figure 5.35, and Figure 5.38 show how the number of station duplications, number of stations, and space used vary as a function of the throughput level. Figure 5.39 and Figure 5.40 visualize how the capital and labor costs respectively vary as a function of the production rate. The required factory space is not shown since it is almost identical to Figure 5.14. The average number of stations and station duplications do not present anything that stands out.

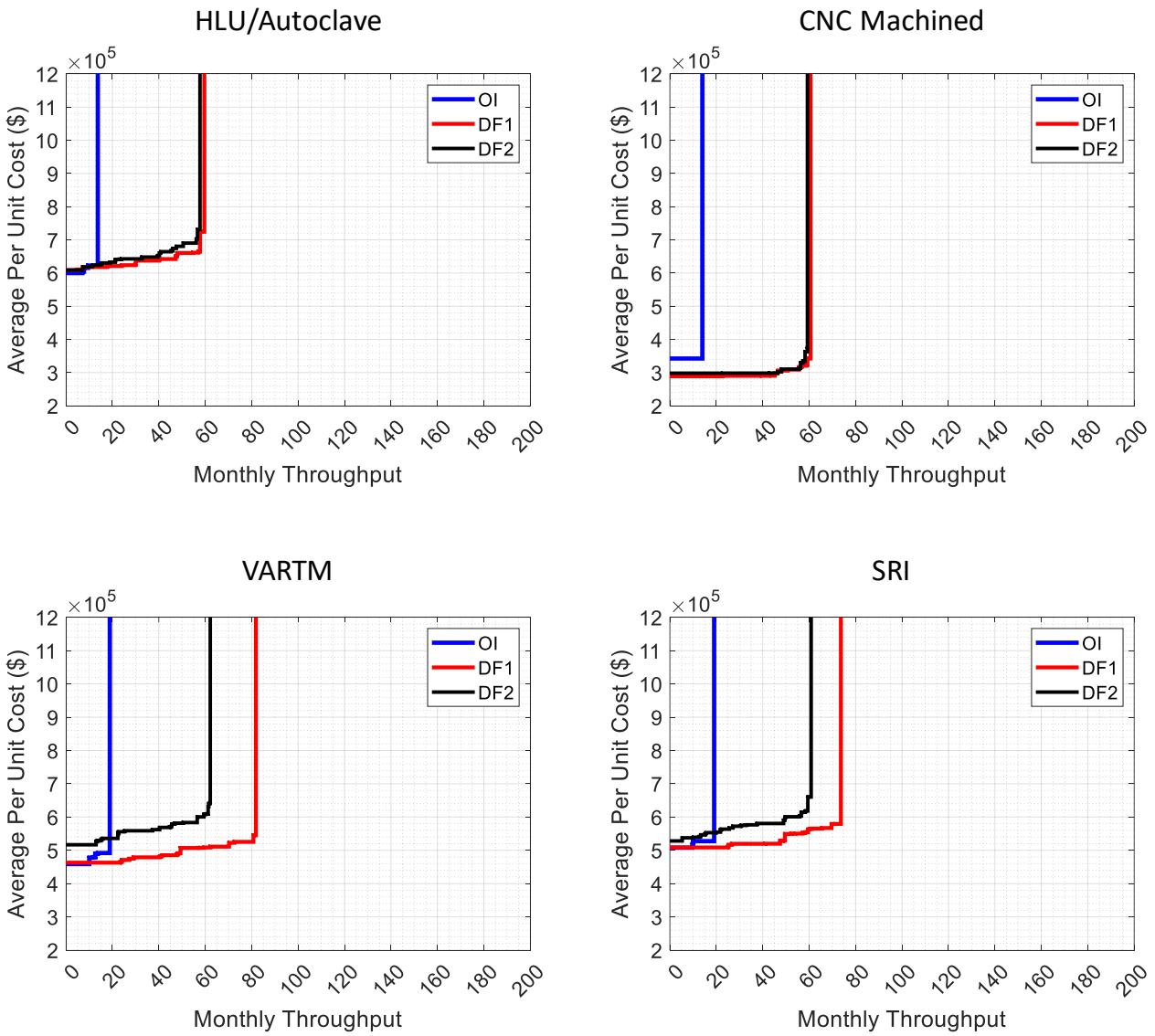


Figure 5.30: Average Summary Attainment Surfaces Obtained by Oesterle et al.'s Implementation and Developed Framework for All Processes, DF1 is Developed Framework with Limited Assembly Sequences, DF2 is No Integral Fabrication or Subassemblies

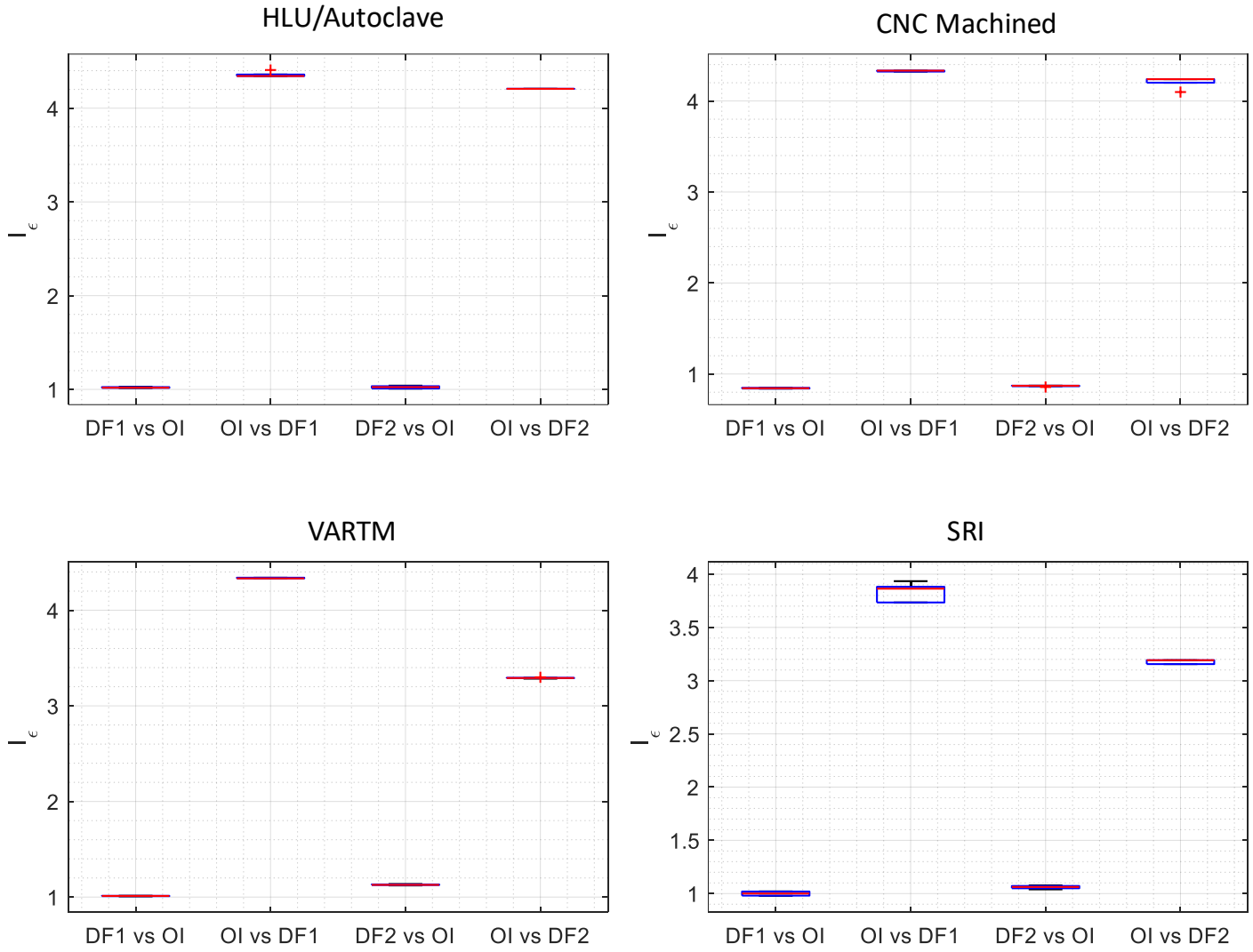


Figure 5.31: Boxplots Showing $I_\epsilon(PF_1, PF_2)$ Values Obtained by Comparing Oesterle et al.'s Implementation and Developed Framework for All Processes, DF1 is Developed Framework with Limited Assembly Sequences, DF2 is No Integral Fabrication or Subassemblies

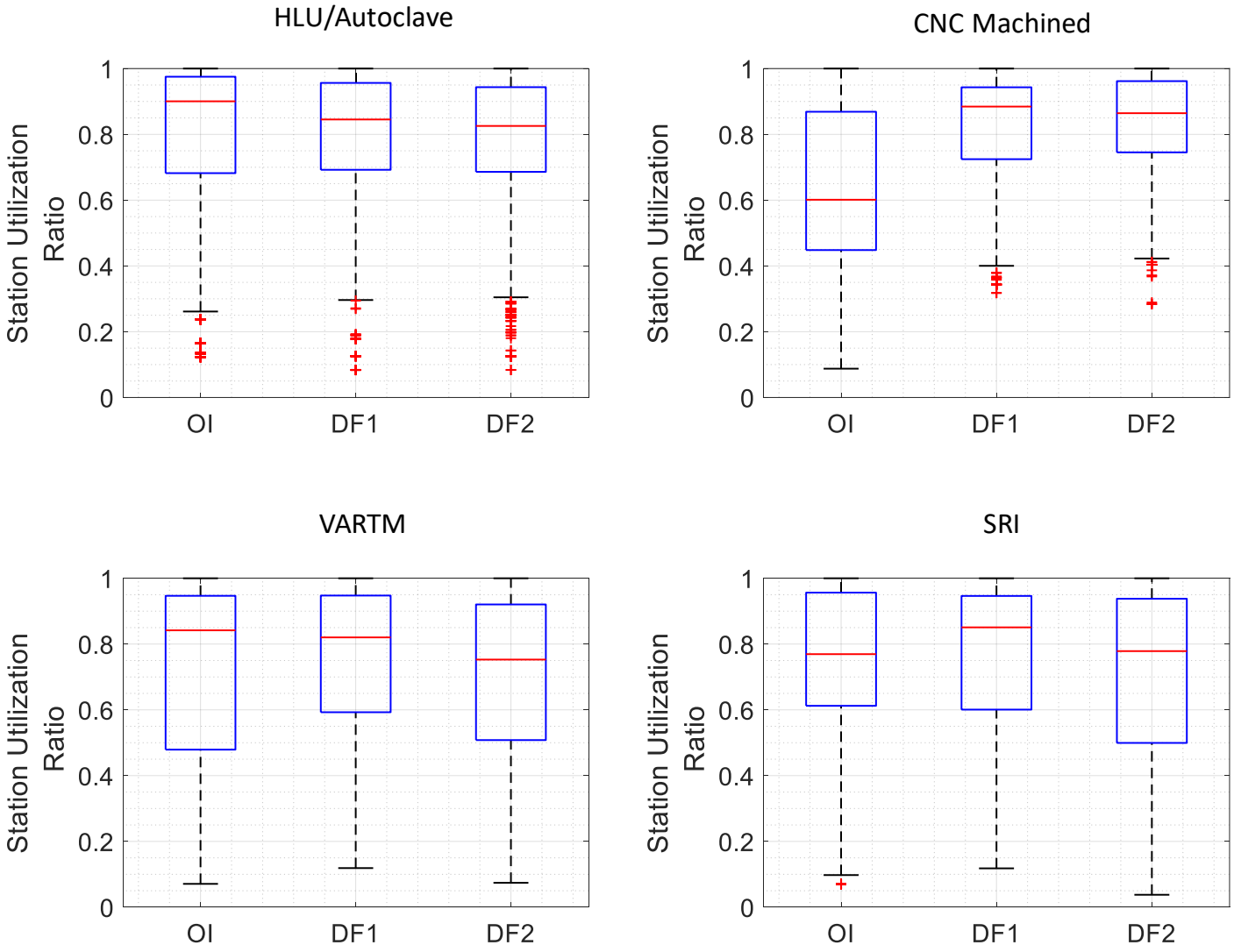


Figure 5.32: Boxplots Showing Distribution of Average Station Utilization Ratio in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework for All Processes, DF1 is Developed Framework with Limited Assembly Sequences, DF2 is No Integral Fabrication or Subassemblies

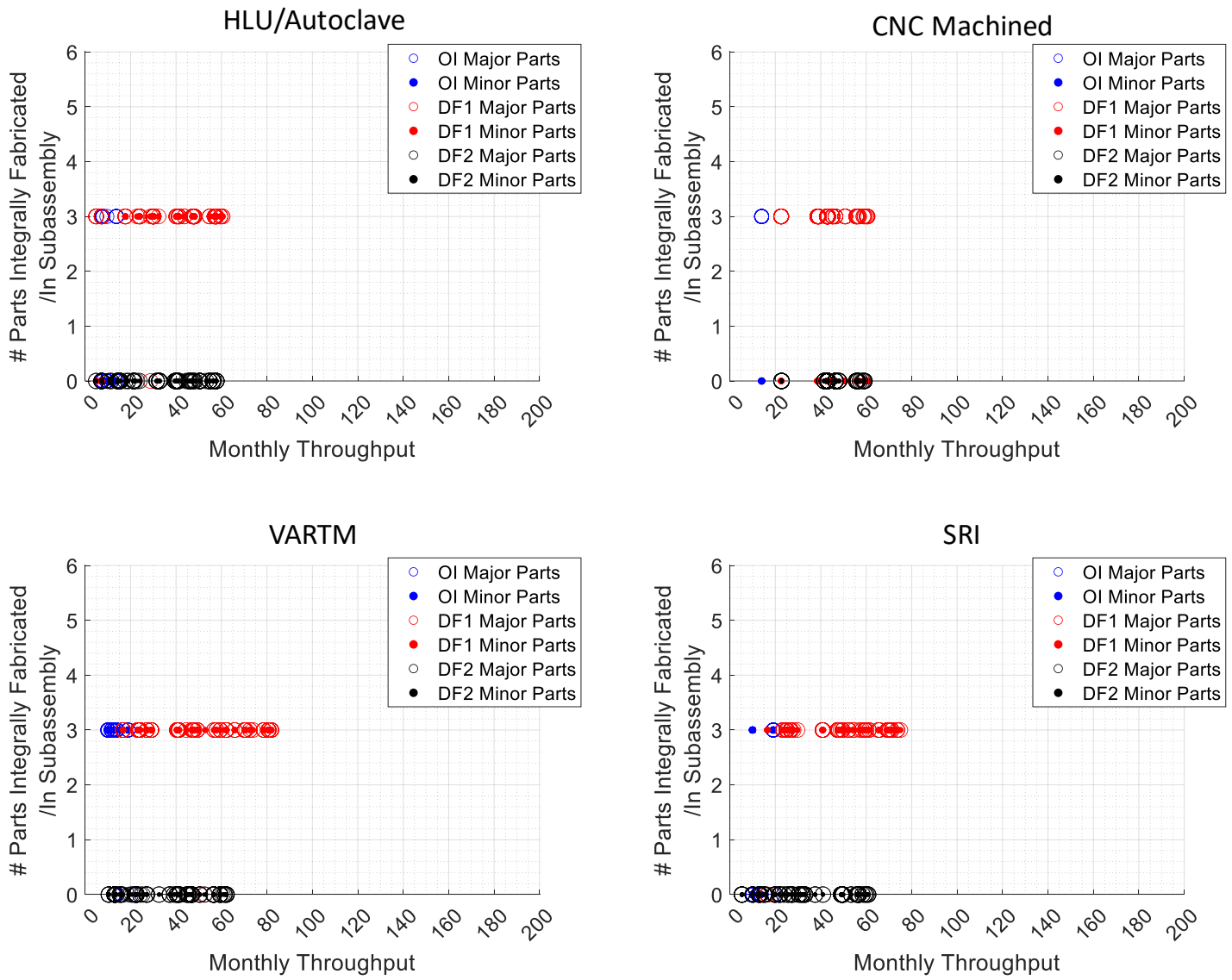


Figure 5.33: Number of Minor and Major Parts Integrally Fabricated or in a Subassembly as a Function of Throughput in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework for All Processes, DF1 is Developed Framework with Limited Assembly Sequences, DF2 is No Integral Fabrication or Subassemblies

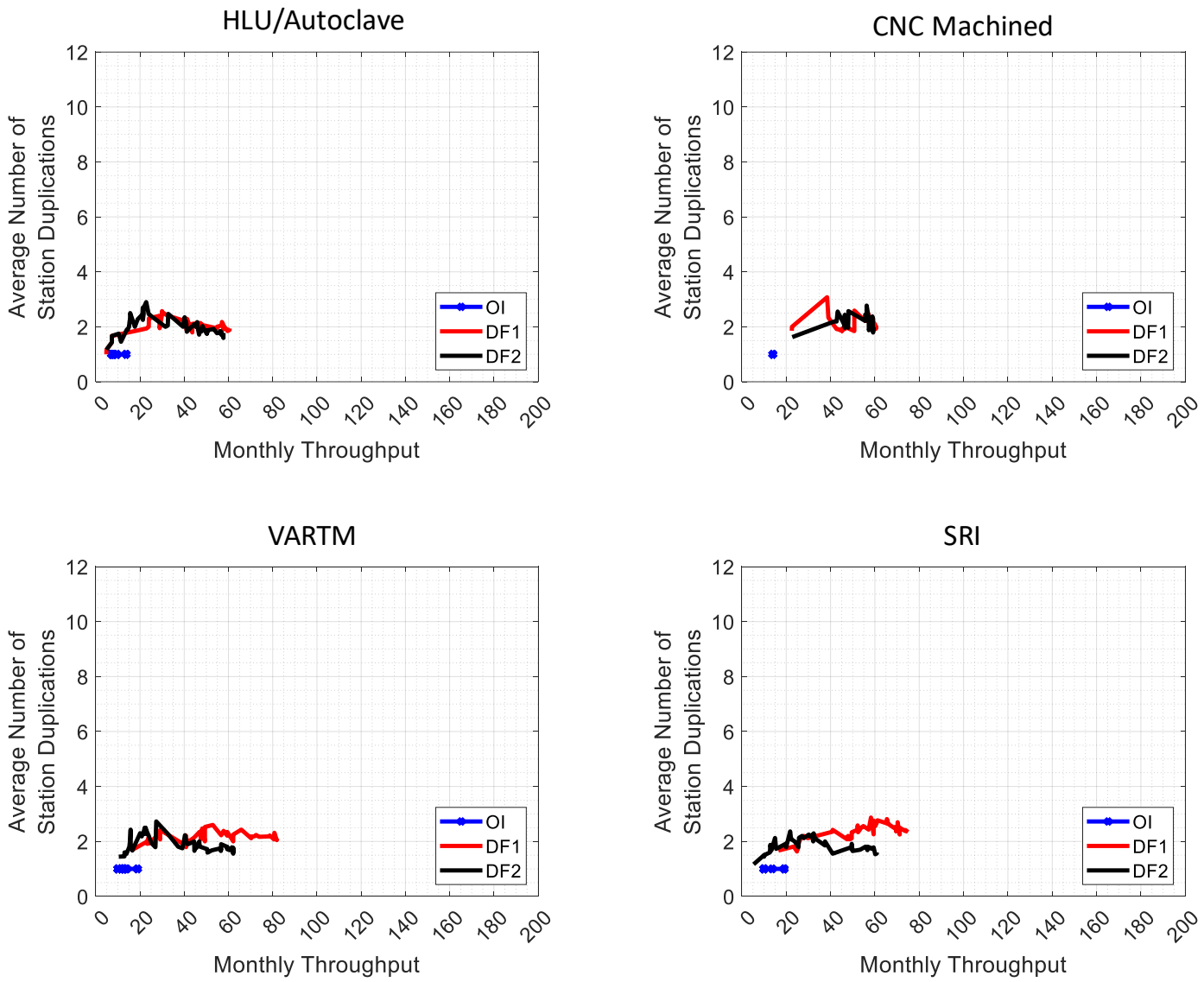


Figure 5.34: Average Number of Station Duplications as a Function of Throughput in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework for All Processes, DF1 is Developed Framework with Limited Assembly Sequences, DF2 is No Integral Fabrication or Subassemblies

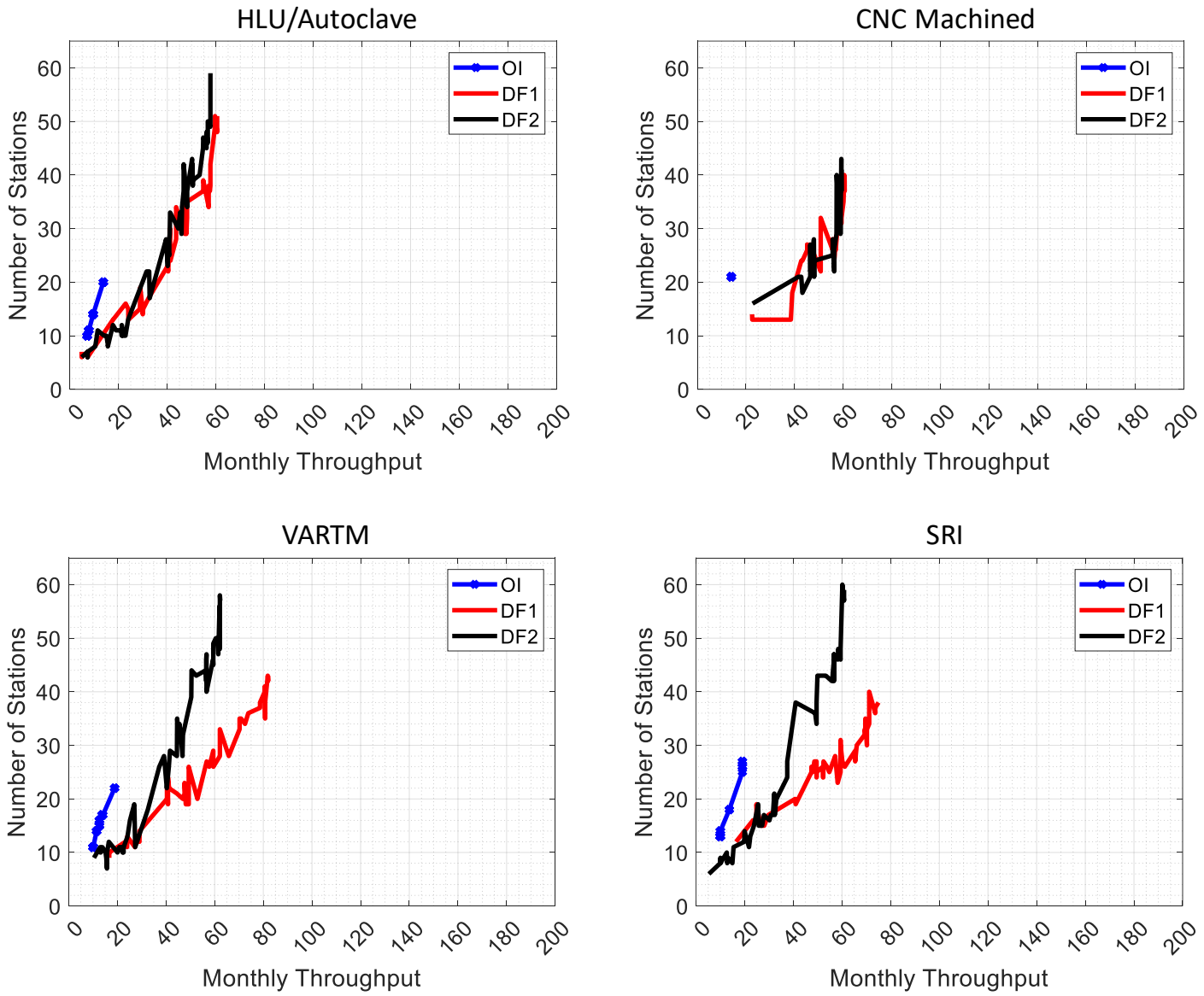


Figure 5.35: Number of Stations in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework as a Function of Throughput for All Processes, DF1 is Developed Framework with Limited Assembly Sequences, DF2 is No Integral Fabrication or Subassemblies

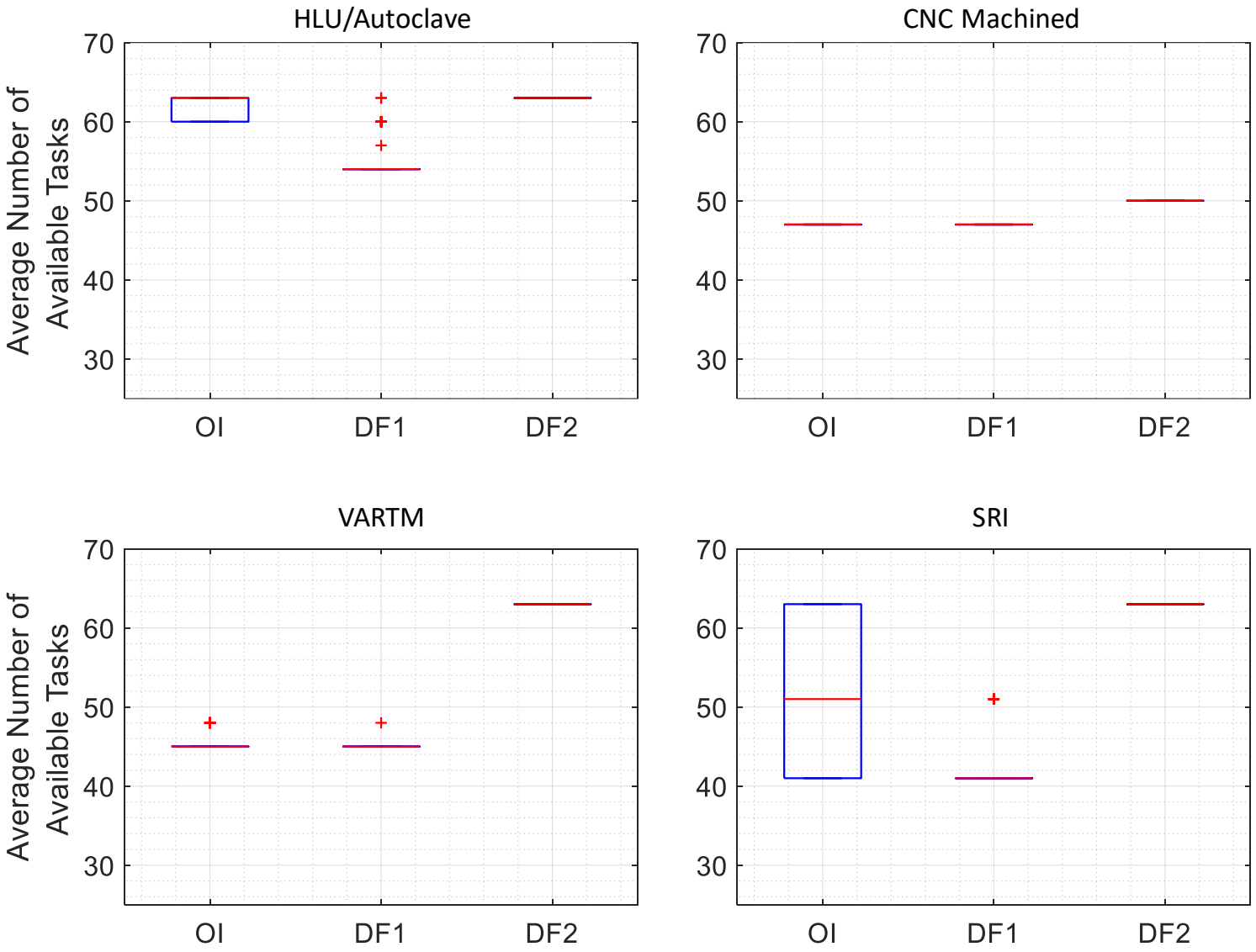


Figure 5.36: Boxplot Distribution of Number of Available Tasks in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework for All Processes, DF1 is Developed Framework with Limited Assembly Sequences, DF2 is No Integral Fabrication or Subassemblies

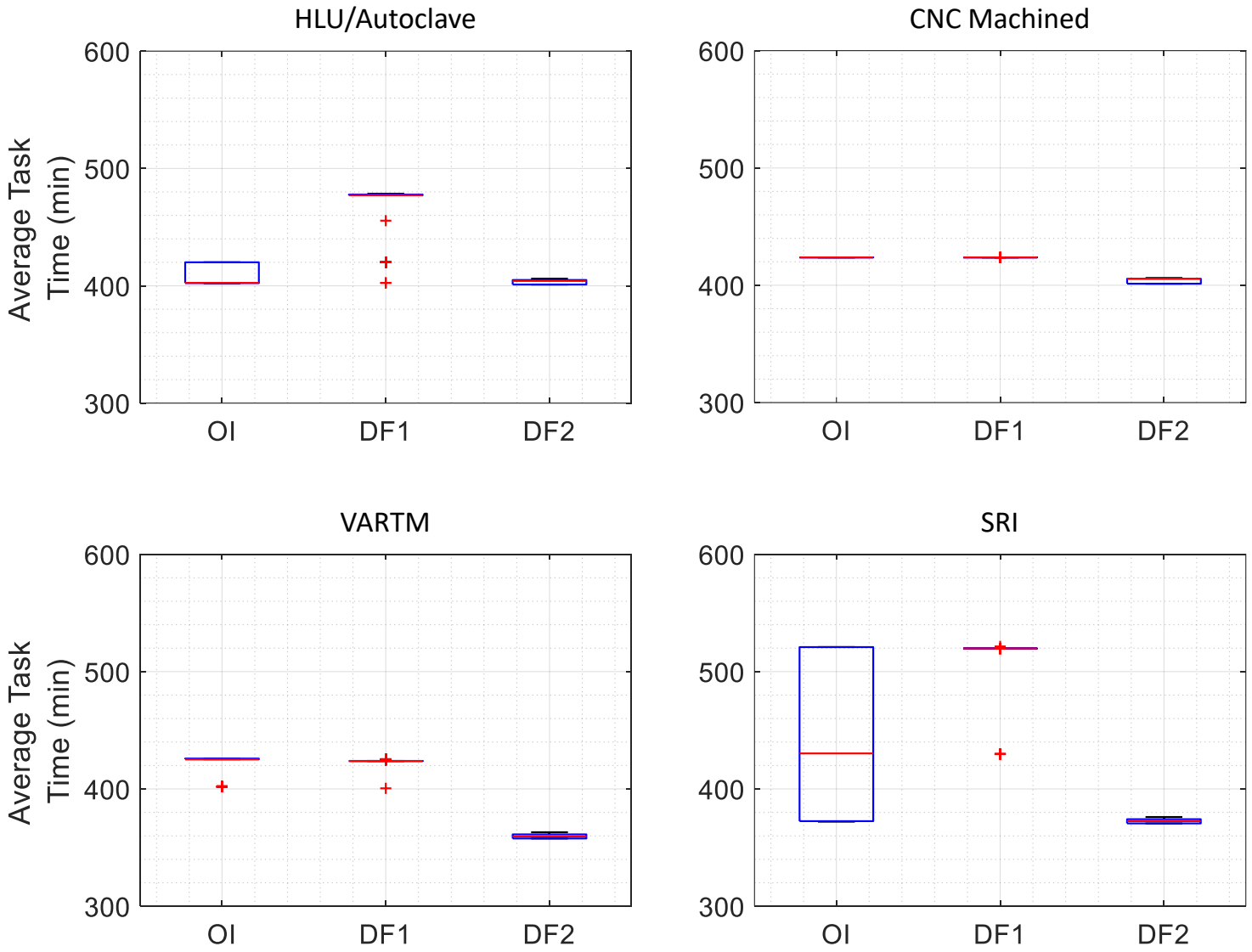


Figure 5.37: Boxplot Distribution of Average Task Times in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework for All Processes, DF1 is Developed Framework with Limited Assembly Sequences, DF2 is No Integral Fabrication or Subassemblies

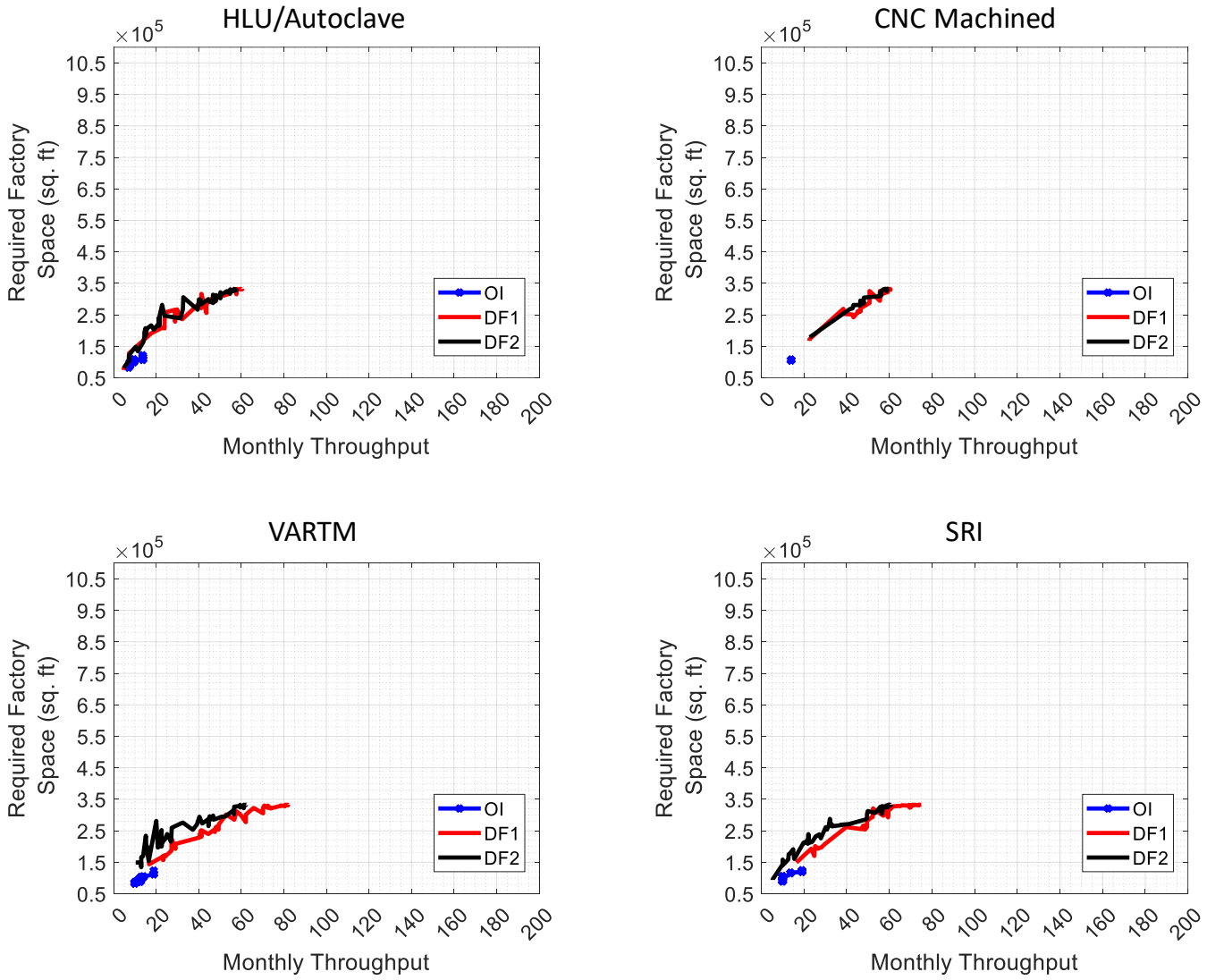


Figure 5.38: Space Used in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework as a Function of Throughput for All Processes, DF1 is Developed Framework with Limited Assembly Sequences, DF2 is No Integral Fabrication or Subassemblies

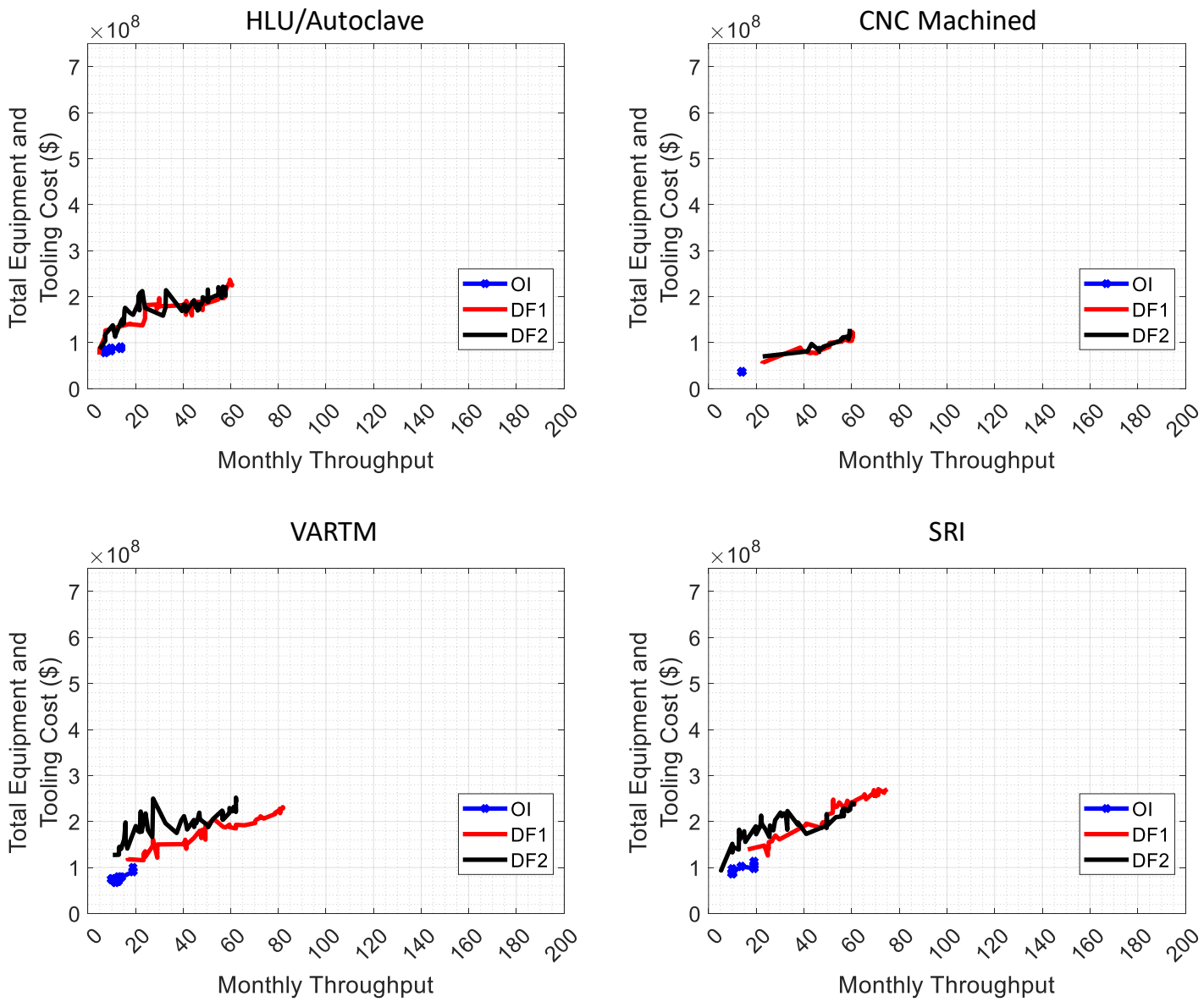


Figure 5.39: Combined Equipment and Tooling Cost in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework as a Function of Throughput for All Processes, DF1 is Developed Framework with Limited Assembly Sequences, DF2 is No Integral Fabrication or Subassemblies

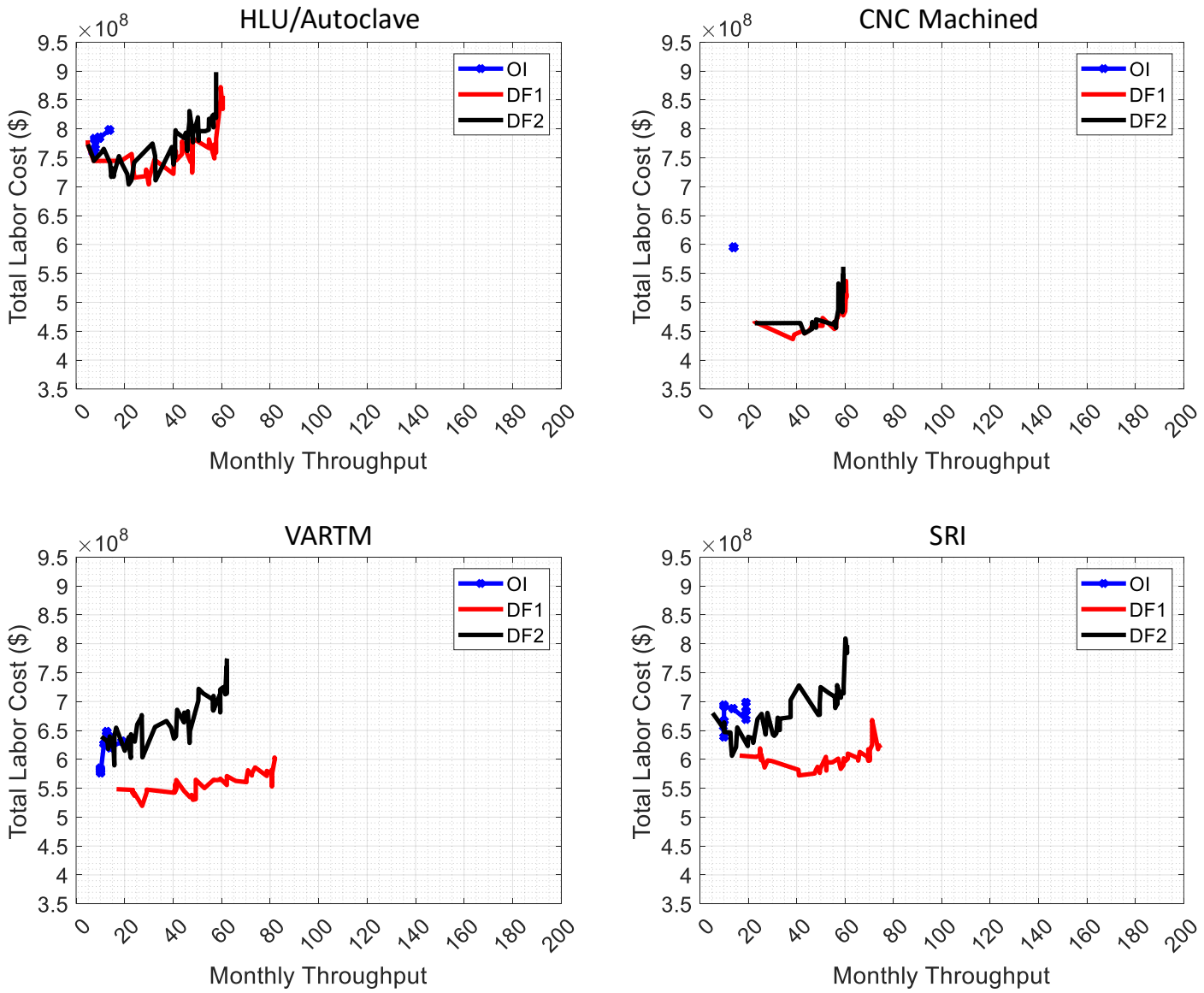


Figure 5.40: Total Labor Cost in Solutions Obtained with Oesterle et al.'s Implementation and Developed Framework as a Function of Throughput for All Processes, DF1 is Developed Framework with Limited Assembly Sequences, DF2 is No Integral Fabrication or Subassemblies

Figure 5.33 shows that the developed framework's solutions are indeed limited either via assembly sequences or in terms of integral fabrication and subassemblies. Figure 5.30 and Figure 5.31 both indicate that while the developed framework's solutions yield significantly higher throughput due to their usage of station duplication, when it is limited to the same or worse sequences or alternatives as Oesterle et al's solutions, it performs similarly, if perhaps slightly worse, in terms of minimum cost. The $I_\epsilon(DF1, OI)$ and $I_\epsilon(DF2, OI)$ values almost always being just barely above 1 in Figure 5.31 in fact means that the developed framework's solutions do not completely dominate Oesterle et al's implementation's solution. However, this is only applicable at very low throughputs that essentially involve no station duplication and only occurs when equal or inferior quality assembly sequences or part combinations are used. It means that when the developed framework cannot leverage any of its material factor considerations, the NSGA-II is likely a better optimizing method than the ε-constraint method used and is able to find a more optimal solution, which is already relatively well established given how comparatively rare the ε-constraint method is and how ubiquitous the NSGA-II is. The exception is for the CNC Machined processed where station duplication is still effective at low throughputs, giving the developed framework's solutions an edge there.

These results also emphasize the importance of selecting the correct set of integrally fabricated parts or subassemblies as the DF2 solutions consistently underperform as compared to the DF1 solutions, having higher capital and labor costs in Figure 5.39 and Figure 5.40. For the labor cost this is because the DF2 solutions consistently have much a much larger number of available tasks in Figure 5.36 while not having a proportionally smaller average task time in Figure 5.37, leading to lower overall station utilization ratios and efficiency in Figure 5.32. For the capital cost, the larger number of available tasks in the DF2 solutions allows for more stations to be made in Figure 5.35 which, when duplicated at relatively similar rates to the DF1 solutions in Figure 5.34, causes more equipment and tooling to be needed and increasing cost. To further emphasize the importance of the

correct integral fabrication and subassembly part combination, Figure 5.14 and Figure 5.38 are compared. The former figure has the DF solutions able to freely set their assembly sequences and part combinations and so have a linear increase in space used as throughput increases that directly lines up with the OI solutions. This means the space is being used as efficiently as possible. The latter figure shows the opposite, where the DF1 and DF2 solutions' trends do not line up with the OI solution's anymore and are steeper, indicating space is being used less efficiently to increase throughput. Proper selection of assembly sequences and part combinations thus have a large impact on both cost and production rate. These trends implicitly existed during the process of selecting the optimal sequences and part combinations but could only be made explicit by forcefully setting the sequence and part combinations used and thus altering the material factors explicitly. The developed framework is able to do this and has better capabilities for it while Oesterle et al.'s implementation does not.

A final, interesting observation is made by looking at Figure 5.17, Figure 5.18, Figure 5.19, Figure 5.33, Figure 5.36, and Figure 5.37. Despite all the possible assembly sequences and part combinations, those figures indicate that the sequences and part combinations yielding optimal cost and throughput do not change significantly as a function of throughput or cost. This implies that there does seem to be an underlying pattern where a bank of more-optimal assembly sequences is used and preferred. This is explored further in the next chapter.

The developed framework has shown it is able to accommodate the material factors of integral fabrication, variable manufacturing processes, spatial constraints, and variable cost and throughput while also including assembly sequencing better than the current state-of-the-art method. It does this both by generating solutions with higher throughput and lower cost as well as being able to justify why those solutions are the way they are and trace those reasons back to the very material factors it was designed to accommodate. It additionally is able to make changes to those material factors and actively alter them as

variables to see what further effects they have on the manufacturing design space and the cost and throughput. These capabilities demonstrate it was worth the time to create and use the developed framework. Based on all this, Hypothesis 2 is accepted.

This chapter was focused on finding a way to solve an integrated ASP and ALB problem that considers all the desired material factors and their resultant characteristics. This is done to eventually be able to perform assembly analysis that, through those material factors, is able to make tradeoffs with aircraft design when it considers its own material factors, with the ultimate goal being to combine the analyses of those two disciplines into one framework able to generate designs with better throughput and cost. To do this, ways to address the individual assembly analysis gaps related to those factors were first identified. They were then combined together with ASP and all the relevant ALB problem types and afterwards wrapped in a combination ε-constraint, GA wrapper to be implemented. Hypothesis 2 was made. This newly developed framework was tested against the current state-of-the-art that also is able to at least somewhat capture the material factor considerations. This test, Experiment 1, used the F-86 wingbox as its the experimental platform and subject of the test. Experiment 1 showed the developed framework does indeed have the claimed capabilities and can fulfill the desired purpose of eventually being able to link up with aircraft design. This is all summarized in Figure 5.41.

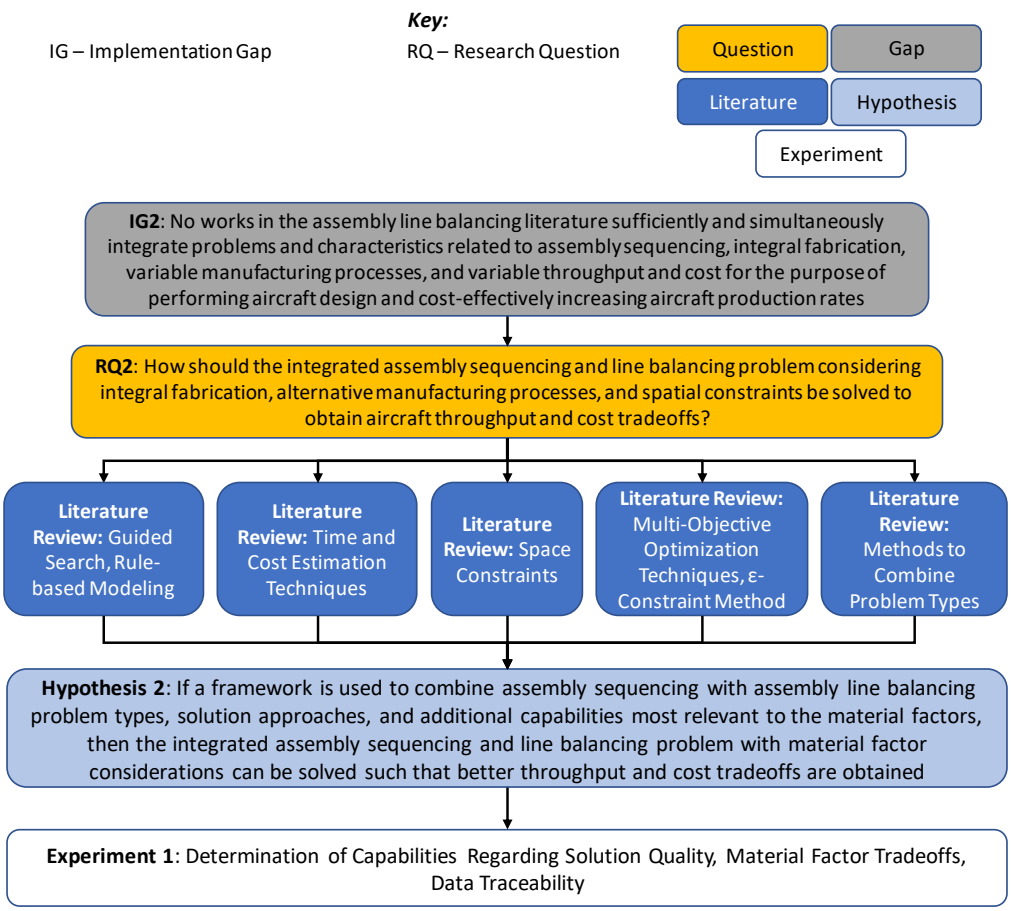


Figure 5.41: Summary Logic Diagram for Research Question 2

CHAPTER 6

INCREASED EFFICIENCY OF ASSEMBLY ANALYSIS

6.1 Research Question 3: Improving Assembly Analysis Efficiency for Aircraft Design Integration

The typical ASP and ALB problems in literature deal with only a single static product architecture with defined, overarching parameters. In this work, the ultimate goal is to have that product architecture not be static at all and to actually determine the best ones along with finding the accompanying best set of manufacturing parameters. Those parameters are the geometry and material variables that determine the product configuration on the aircraft design side and the material factors, assembly sequences, and line balances that define the factory and manufacturing system on the manufacturing side. These parameters represent the high level decisions designers can make for their entire aircraft and its means of production.

In the previous chapter, an integrated ASP and ALB framework was developed able to handle the manufacturing analysis portion of this. Due to its embedded usage of a meta-heuristic technique, the genetic algorithm, the framework is able to handle the majority of the aforementioned parameters as variables within a single problem instance that took up to half an hour to run and create a Pareto front for Experiment 1. However, this could not be done for all the parameters, with the most notable exceptions being the spatial constraints and the manufacturing processes. Those are categorical variables that, to properly examine, require a problem instance dedicated to every combination of them when creating Pareto frontiers. Due to Pareto frontier generation being stochastic, every combination of categorical variables must also be rerun multiple times to obtain a distribution of Pareto fronts and see what the true Pareto front for that combination is. This was manageable when

only the manufacturing discipline categorical variables were included: for Experiment 1 in the previous chapter, there were only 4 options for manufacturing processes, 3 for spatial constraints, and each combination was rerun 5 times to obtain the Pareto front distribution. This was all done for as many sets of situations as needed.

When the aircraft design process is added to this and all of its own variables are included, the situation becomes not so manageable. Table 6.1 depicts a small listing of potential aircraft design variables that could have to be optimized alongside the integrated ASP and ALB framework being carried out, with the continuous variables having discrete settings to remain generic and aircraft-agnostic.

Table 6.1: Table of Alternatives Showing Potential Aircraft Design Variables

Variable Type	**Variable Options**		
Range	Low	Medium	High
Wing Sweep	Low	Medium	High
Wing Taper Ratio	Low	Medium	High
Wing Aspect Ratio	Low	Medium	High
Wing Span	Low	Medium	High
Rib Spacing	Low	Medium	High
Rib Design	Separate Caps, Stiffeners, Shear Ties	Unitized Cap & Web	Undulating Web
Wing Configuration	Low Wing	Medium Wing	High Wing
Spar Shape	Outwards C	Inwards C	Z
	I	J	
Stringer Shape	Blade C	Hat	Z
	I	J	

This represents about 164,000 design alternatives by itself, on top of the fact that variables such as wingspan and aspect ratio are actually continuous in nature rather than categorical as depicted. In the worst case scenario, the integrated ASP and ALB framework would have to be carried out as many times as there are design alternatives which, even not considering the manufacturing variables, is completely infeasible. Traditionally, this issue has been addressed by leveraging meta-heuristic algorithms to filter through the less-

promising design alternatives. As shown by the works reviewed in the previous chapter as well as by the developed framework itself using a GA, meta-heuristics work well for continuous variables and even pseudo-continuous ones like line balances and assembly sequences, with the caveat that they will take longer to converge. Assuming meta-heuristic techniques can handle the continuous aircraft design variables in a somewhat efficient manner, however, still leaves over 200 design alternatives representing the various combinations of the categorical variables. It is not preferable to repeat the integrated framework developed in the previous chapter that many times along with however many times is needed to capture the different categorical manufacturing variables, on top of the necessary number of repetitions to adequately account for Pareto front generation stochasticity. This is because the current assembly analysis, in the context of needing to be performed a very large number of times to accommodate for integration with the aircraft design process, takes a relatively large amount of time to carry out. If the assembly analysis takes too long, it becomes difficult to explore the aircraft design space for the best designs early in design in a timely manner.

It is desirable, therefore, for the assembly analysis to be altered to be as efficient as possible. Currently, no works have addressed or provided guidance on how to implement or alter assembly analyses to account for the time limitation considerations involved with needing to perform them numerous times. None have needed to in the past due to not explicitly performing architecture design while performing ASP and line balancing, but it is a capability that is needed now.

Research Question 3 *How can the integrated assembly sequence planning and line balancing problem be more efficiently addressed in the context of needing to be solved many times for the aircraft design process?*

6.1.1 Potential Efficiency-Improving Methods

The most direct approach to improve on the assembly analysis's efficiency is to simply have it use techniques able to obtain solutions more quickly. This necessitates using techniques to either have the algorithms converge more quickly or have them run faster, i.e., make more function calls per unit time. There are many examples of this. Tang et al. used Arena's Principle to improve on the NSGA-II and gained a 10% computational time benefit[241]. Mishra et al. used a non-dominated solutions update approach where offspring solutions are only compared with pre-sorted non-dominated solutions until either the offspring is non-dominated or dominated by all other solutions, yielding computation times of $O(MNlogN) + O(N^2)$ [242]. Becerra and Coello combined the ε-constraint method with a cultured differential evolutionary algorithm, similar to the developed framework's combining of the ε-constraint method with a GA, and showed their method had superior convergence to the NSGA-II in the same amount of time, necessitating fewer repetitions for Pareto front stochasticity [243]. Landa et al. have the decision makers identify a region of interest on the Pareto front before splitting up the region using the ε-constraint method and solving each sub-problem with a constrained evolutionary algorithm, improving on the quality of a similar NSGA-II implementation for the same number of function calls [244]. And Fan et al. used an improved ε-constraint handling method to better deal with highly constrained multi-objective problems [245].

A commonality among all these works that either directly improve algorithm speed or improve algorithm quality, thus lowering Pareto front generation stochasticity and lowering the number of repetitious runs needed to capture their spread and indirectly improving efficiency, and among all such current improvements in meta-heuristic techniques as shown by Coello [246] is that there will always be a better algorithm. Using a specific algorithm, method, or technique will always result in a better one being made to improve on it in the near future at the cost of spending a large amount of development time tailoring the problem to each algorithm. Selecting a specific algorithm or technique improvement to

improve the analysis efficiency is not desirable because those techniques or methods will require a specific implementation strategy for the technique used, only for the technique to later be supplanted in performance anyways. It is preferable to find a more generalized way to increase the assembly analysis efficiency so that it is algorithm and technique agnostic. This way, the assembly analysis is conducive to all future improvements in algorithms and techniques as opposed to being beholden to just a single one. For this reason, technique improvements are not considered.

A possible generalizable approach is to increase the computational power the assembly analysis has access to. This is done by leveraging High Performance Computing (HPC), most often via usage of Graphics Processing Unit (GPU) acceleration. This allows algorithms to work in parallel and can massively increase their computational speed. As best demonstrated by Aguilar-Rivera, though the time complexity remains can remain roughly the same, the actual raw speed is significantly faster. Aguilar-Rivera reported on his fully vectorized, massively parallel GPU-using NSGA-II algorithm being 20-300 times faster than the baseline NSGA-II with roughly similar solution quality [247]. Despite this, the issue with relying on HPC and GPU acceleration is a fundamental one–they only focus on speed and generally ignore efficiency and require large amounts of physical resources to do so. This is akin to only increasing the amount of space available to increase throughput in the previous chapter with no regards to using the best manufacturing processes, assembly sequences, integral fabrication and subassembly part combinations, line balances, etc. More resources in the form of more space or computing power can always be used for any problem, but it does not make solving the actual problem or performing the actual analysis any more efficient. As such, all those invested resources could be getting spent on a very sub-optimal setup. Essentially, without an efficient way to take advantage of the resource, the resource will get squandered. Analyses made more efficient and then utilizing HPC will always run faster than a less efficient version of the same analysis also using HPC. As a result, relying solely on HPC, or solely on hardware in general, to increase computational

speed is not an appropriate avenue to make the assembly analysis more efficient because it completely sidesteps the problem and does not address it.

A possible alternative noted by Mavris et al. to improve analysis times and help with its efficiency is to use surrogate models [248]. Surrogate models are essentially approximation functions that can mimic high fidelity models, taking in similar inputs and outputting similar outputs, at a fraction of the computational cost. They are made via a training process where, first, their functional form is selected. This can range from response surface equations to radial basis functions to artificial neural networks. The desired input and output variables are then selected, at which point the high fidelity model is used to take in a range of the desired input variables and yields the corresponding outputs. The surrogate model is afterwards trained to be able to predict the outputs correctly and accurately given the corresponding inputs, akin to performing linear regression on a series of datapoints. The result is basically a black box approximation function that can substitute for the high fidelity model but, because it does not actually perform any of the associated complicated equations, can be run with very little computational cost. The results from surrogate models are thus relatively accurate so long as they are not given inputs outside the range of the inputs they were trained with.

Surrogate models will be heavily used in the final integrated framework of this work, when the aircraft design and assembly analyses are combined, primarily to substitute for the task time and cost estimation program and the aircraft design and structural analysis tools once the product architecture variables are actually varied. However, that still leaves the baseline assembly analysis from the previous chapter untouched. Indeed, the majority of the assembly analysis's time is spent on generating and evaluating assembly sequences, line balances, and the amount of space they take up, which does not involve any computationally expensive analysis models or programs that surrogate models can actually replace. The only way for surrogate models to be able to aid in improving the assembly analysis efficiency is if they could somehow replace the ASP, ALB, or spatial constraint process

itself. A notable issue that prevents this from being feasible is surrogate models only give approximate solutions that, while they may be very close to being exactly accurate, are still not perfectly accurate. This precise accuracy is needed because assembly sequences, line balances, and the space calculations involved with them are extremely sensitive to the ordering of the parts and tasks; any small deviation can completely change the line balance or sequence being examined or outright make it infeasible. To properly train and create an accurate enough surrogate model would require so many cases to be run that it is akin to just normally performing the assembly analysis, which defeats the entire purpose.

Additionally, surrogate models are basically one-to-one functions: they can only produce a single output given a single input. In contrast to this, a single assembly sequence can produce multiple line balances that yield wildly different costs and throughputs, and a single static architecture can produce multiple assembly sequences that themselves produce very different results. Each product architecture, assembly sequence, and line balance can produce multiple different results, and so to make surrogate models to replace those analyses requires that the single desired result from each must be specified. But the problem being tackled is multi-objective in nature, and so the desired result or result type is not known in advance, i.e., it is not known what weighting of minimizing cost versus maximizing throughput will yield a non-dominated solution. This means the surrogate models would have to somehow be able to produce multiple results per set of inputs. They cannot do this. Ultimately, trying to use surrogate models to replace ASP and ALB is impractical because a surrogate model cannot wholesale replace an entire dynamic analysis type. There are too many constraints, possible outputs, inputs, and interactions to be able to properly include everything in a surrogate model or even a set of surrogate models. Due to all these difficulties, surrogate models are not a viable option for further improving assembly analysis efficiency.

An option to explore inspired by an observation from the last chapter is to reduce the problem size. In the developed framework a candidate design is only able to end up on

the Pareto front as a non-dominated solution if it is able to have both the correct assembly sequence and the correct line balance. Determining either of those individually is already an NP-Hard problem where the simulation time and complexity increase exponentially the more parts and tasks are involved. Needing to do both simultaneously is one of the reasons why the developed framework needs such a long time to converge on solution at each ε-constraint. However, it was noted several times in Experiment 1 in the previous chapter that the assembly sequences of the non-dominated solutions tended to be very similar or the same as each other and did not change much even as the throughput ε-constraint levels were varied. The possible alternative with this is to reduce the size of the problem by identifying the most promising assembly sequences first and only performing ALB for those instead of across the entire spectrum of possible assembly sequences, which even for the F-86 experimental platform's limited number of major parts can number well beyond 10,000 unique sequences once integral fabrication and subassemblies are accounted for. In other words, a promising way to streamline the assembly analysis is to reduce its problem size by only carrying out line balancing for the best sequences instead of for every sequence generated.

The idea behind this is that a large amount of time is likely spent getting out of the local minima formed by trying to obtain the optimal line balances for sub-optimal assembly sequences. By only line balancing the most promising sequences, solutions are less likely to be trapped in these local minima in the first place, saving large amounts of time and greatly speeding up the design space exploration. Indeed, this logic can be extrapolated even further to streamline not just the assembly analysis but the entire integrated aircraft design and assembly analysis as well; in the same way that line balancing a sub-optimal assembly sequence is an inefficient use of time, determining the optimal assembly sequence for a sub-optimal aircraft design is similarly inefficient. However, whereas Experiment 1 demonstrated it could be possible to determine the best assembly sequence separately from the best line balance, there is no such guarantee between the aircraft design analysis and

ASP due to ASP's geometric reasoning intricately connecting the two. Thus, in the best case scenario, the optimal aircraft design must be determined concurrently with the optimal assembly sequence, with the optimal line balance occurring afterwards.

This method appears promising because the ASP problem is, by itself, much simpler than the integrated ASP and ALB problem. The former only has to worry about the sequence of parts in an assembly, while the latter has to deal with the sequence of parts and how the resulting tasks are distributed among workstations. It is thus highly likely that a process consisting of performing assembly sequencing analysis first followed by the solving the integrated problem on the best solutions is faster than solving the integrated problem for all design alternatives. This only rings more true with the inclusion of aircraft design, where it is almost certainly more efficient to optimize just the aircraft's design and assembly sequence together first and then its line balance afterwards as opposed to trying to optimize everything at once. Additionally, solving the integrated problem for a smaller subset of solutions already filtered by the aircraft design and sequencing analysis makes the integrated problem resemble a more typical line balancing problem. This lowers the overall problem's complexity and is likely to allow it to be solved in less time. This notion is further supported in the assembly literature by the fact that integration of multiple assembly problems is so comparatively rare. In a real production setting all the different assembly analysis types would be considered at some point for an optimal production plan. The fact that so many works look at only a single analysis type at a time when they all eventually need to be examined means the implicit assumption is that they are all addressed sequentially. If integrated problems truly are significantly faster in solving, then many more works would be focused on them than currently exist. There is thus a high likelihood that splitting up the analyses in this fashion is faster than solving a single very large integrated problem.

The goal, then, is to create a method to determine which designs and sequences are more promising without going through the effort of evaluating all combinations of all possible aircraft, assembly sequence, and assembly line balance designs. In other words, all the

designs are first judged based on some type of merit-based metric, after which only the most promising ones have their line balances generated and evaluated to be placed on the Pareto front similar to what was done in the previous chapter's experiment. The proposed method will thus require a two-step approach, where the first step is treated as a combined aircraft design and assembly sequencing problem to filter out the less promising design alternatives. The assembly sequence is the focus of the first step because the best aircraft design is one that produces the best assembly sequence. This is so because the best aircraft designs are ones that minimize cost and maximize throughput while meeting performance constraints; in the first step the only way for it to affect the cost and throughput is through the assembly sequences, ergo, the optimal aircraft designs are ones that yield the optimal sequences. The second step is just like the original integrated ASP and ALB framework developed in the previous chapter, except it is only performed on the first step's best design alternatives. The objective of the first step is thus to see which design alternatives the best sequences belong to so that they can all be passed on to the second step. This approach promises to decrease the problem size and increase the problem-solving efficiency.

However, the increased efficiency with which solutions are obtained and the design space explored using this two step approach is irrelevant if the results are not accurate or representative of the true Pareto front and design space of optimal solutions obtained by optimizing everything all at once. The two step approach is only worth considering if the best alternatives as judged by the first step do indeed produce the best solutions among all the possible design alternatives. In other words, it is only justifiable to use the two step approach if its results are approximately equivalent to solving the integrated problem for all the design alternatives. This narrows down to mean that the assembly sequences deemed to be the most promising in the first step need to actually generate the best line balances in the second step among all the sequences possible. The attention being on the assembly sequences is because using the best ones to limit the number of sequences needing to be considered is the primary draw of this approach in the first place, causing the assembly

sequences to be the primary subject of evaluation in the first step. Ways to make the results representative are covered in the next section.

6.1.2 Sequentially Solving the Integrated ASP and ALB Problem

Traditionally, the assembly literature implicitly assumes that the best assembly sequences from ASP yield the best tasks and therefore the best precedence relations for line balancing, ultimately resulting in the best line balance out of all the possible line balances for all possible sequences. The unspoken assumption is thus that optimal designs and design alternatives from ASP naturally and by default produce the best line balancing results. This is the reason why so many assembly works focus specifically only on a single type of assembly analysis such as ASP or ALB or logistics or factory layout, as opposed to looking at integrating multiple together. However, due to this assumption being implicit, there tends to be little to no means of communication between the different analysis types, ASP and ALB in this scenario, when they decide on their respective optimal solutions. As a consequence, the assembly analyses are effectively silo'ed off from each other and this assumed optimality equivalence from sequentially addressing each assembly analysis is not actually guaranteed.

In fact, Tseng et al., Lu et al., and Rashid et al. explicitly noted that this silo'ing tends to often result in sub-optimal combinations or yield outright infeasible results. The two analyses must be able to communicate in some way, as is attempted in the three mentioned works, to be better able to converge on optimal solutions [142, 149, 154]. Despite this, the approach taken by these works is similar in nature to trying to solve a single very large and complex integrated problem. This is because they simply just look at metrics pertaining to both ASP and ALB and try to simultaneously optimize both sets of metrics. This is not an appropriate path to take for the two-step approach because metrics for both assembly analyses cannot be simultaneously optimized in the first step; objectives such as cycle time and number of workstations cannot be optimized when only ASP is being carried out and no

tasks are being assigned to workstations. Although the three mentioned works do guarantee the ASP/ALB optimality equivalence to a better extent then the rest of the literature, they are inapplicable for the attempted two-step approach.

Even so, an alternative method can still be gleaned from the route that Tseng et al., Lu et al., and Rashid et al. took [142, 149, 154]. Instead of trying to optimize both sets of objectives at once, another method is to optimize only a single set of objectives that can be evaluated by ASP alone and have said single set of objectives be representative of both ASP and ALB analysis. This would enable the two assembly analyses to effectively communicate with one another and render the implicit optimality equivalence from before much more explicit. It would act as a sort of guarantee that the optimal design alternatives from assembly sequence analysis would also, to an extent determined by the objectives chosen, result in the optimal integrated ASP and ALB results.

The proposed two step procedure is thus as follows. The developed framework's procedure from the previous chapter is followed until the assembly sequences are generated. At that point, as many assembly sequences are generated as possible to provide a pool of assembly sequences to evaluate. The ASP-to-ALB precedence relations translation still occurs to obtain the task precedence relations and be able to estimate the task times and costs. The task times and costs are used to evaluate the assembly sequences according to a set of objectives representative of both the ASP and ALB analyses, yielding a ranked list of the best sequences and their design alternatives. The most promising sequences and their design alternatives then go through the rest of the process described in the previous chapter for the integrated sequencing and line balancing problem, at which point the Pareto front of the best designs is generated. This procedure reduces the extra work of having to line balance and evaluate every design alternative and assembly sequence and instead focuses only on the most promising ones, which are determined by the much simpler and less taxing first step in the two step approach. The procedure thus promises to speed up the analysis process while retaining the same final quality and accuracy as the original method.

Figure 6.1 demonstrates this comparison of the general procedure of the original versus the proposed approach.

Of course, the proposed two step procedure is fully dependent on objectives representative of both the ASP and ALB analyses being used. The trick, then, is to find these objectives. To distinguish them from the end goal objectives of cost and throughput, these objectives will now be referred to as evaluation criteria.

6.1.3 Potential Evaluation Criteria

By default, objectives from the ALB analysis cannot be used as evaluation criteria for the proposed two step approach because the evaluation criteria will be used for ASP, which does not look at workstations or task assignment and so lacks the steps needed to calculate those objectives. Possible evaluation criteria will consequently only be selected from objectives that can be evaluated solely by performing ASP analysis. The majority of the traditionally used ASP metrics were covered back in Section subsection 3.1.2. To review, the primary ones most commonly used in ASP literature are the number of assembly direction changes, tool changes, reorientations, and assembly type changes. These metrics are still useful for ASP by itself. However, it is evident that they hold no considerations at all for the ALB analysis, which is more focused on aspects like cycle time and number of workstations. In fact, as pointed out by Lu et al., the interactions between these traditional metrics and the ALB analysis could actually cause the results to be less accurate if line balancing is not explicitly performed while using them; the number of tool or type changes between the assembly of two parts is irrelevant if they are in separate workstations, for example, causing sequences taking advantage of that fact to be deemed less optimal if only the traditional metrics are considered without looking at line balancing [149]. For this reason, usage of these traditional metrics is inappropriate. Similarly, other related ASP metrics such as stability, fixture complexity, and assembly angle are considered inappropriate as well.

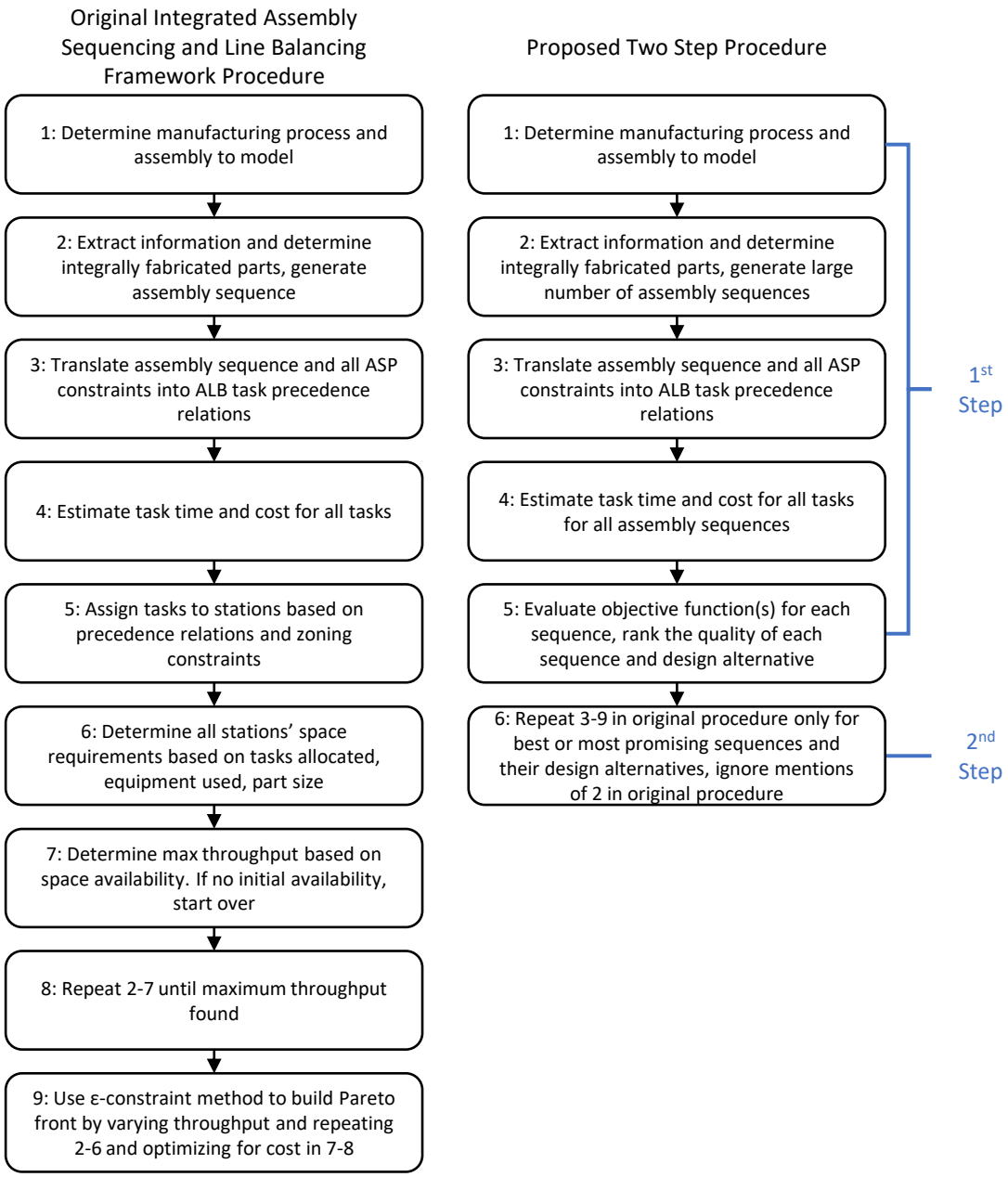


Figure 6.1: General Procedure of Original Approach Versus Proposed Two Step Approach

About the only metrics left that can be used during ASP that are not completely specific to it are production cost and production time. These are more beneficial than the other metrics since, while they are useful for ASP, they can also tie in to the ALB analysis. Assembly cost is in fact an important ALBP metric, reviewed in subsection 3.2.5, making it relevant. To differentiate between the production cost metric being proposed and the final cost obtained from solving the integrated sequencing and line balancing problem, the former will be called sequencing cost and the latter final cost. Sequencing cost is distinct from final cost because it does not include the cost of duplicating workstations or other station-related costs while the latter does. Production time, defined as the sum of all the task times and distinct from production rate, can also indirectly relate to line balancing. Sequences with shorter production times are more likely to have individual tasks that take up less time than sequences with longer production times. This makes more tasks from the former able to fit into fewer workstations to decrease cost. Similarly, given the same number of duplicated tasks and workstations, sequences and alternatives with shorter production times also yield higher production rates or, if a specific rate is trying to be achieved, have to be duplicated fewer times and result in lower final costs. Due to these better connections to the line balancing analysis and their close relationships to the final cost and throughput metrics, sequencing cost and production time are candidate metrics for the evaluation criteria in the two step approach.

To further define what they are, sequencing cost C_{seq} is the sum of the total sequencing labor cost $C_{seqlabor}$, material cost C_{seqmat}, sequencing tooling cost $C_{seqtool}$, and sequencing equipment cost $C_{seqequip}$. Similar to the final cost, the sequencing cost is done on an average per unit cost basis, so the number of units N being produced must be known beforehand. The sequencing labor cost is distinct from the integrated problem's labor cost in that the former does not include idle time inefficiencies, due to workstation assignments being absent, while the latter does. The material costs are calculated in exactly the same way as their integrated problem counterparts. The sequencing equipment costs account for only one

copy of each type of equipment needed for the fabrication and assembly processes whereas the integrated problem's equipment cost accounts for all duplicates of the stations as well. The sequencing tooling cost accounts for one set of tooling for every task, including every assembly task, since there are no workstation assignments with which different assembly tasks might share tooling like in the developed framework. The cost is represented as:

$$C_{seq} = (C_{seqlabor} + C_{mat} + C_{seqtool} + C_{seqequip})/N \tag{6.1}$$

The sequencing labor cost is the sum of all products of the individual task times T_{labor} for each task, the wage rate ω for that task, and the number of workers for that task type z, for all s number of tasks. The number of workers for each task is based on the task type and shown in section A.4. This is all multiplied by the number of units N. It is represented as:

$$C_{seqlabor} = N * \sum_{i=1}^{s} T_{labor_i} z_i \omega_i \tag{6.2}$$

The sequencing material cost is exactly the same as the integrated problem's material cost and so is not presented again. The sequencing equipment cost is the sum of the individual equipment costs C_{eqi} of all d unique types of equipment used by the design alternative. The sequencing tooling cost is the sum of the individual tooling costs C_{tooli} of all the tasks s, since every task needs its own tooling in this scenario. Unlike the integrated problem's equipment cost, there is only one copy of each equipment's cost since duplication is not involved and zoning constraints are not accounted for. These costs are represented as:

$$C_{seqequip} = \sum_{j=1}^{d} C_{eqi_j} \tag{6.3}$$

$$C_{seqtool} = \sum_{j=1}^{s} C_{tooli_j} \tag{6.4}$$

The production time $T_{production}$ is distinct from the throughput and is represented as the sum of all s number of tasks' labor times T_{labor}, represented as:

$$T_{production} = \sum_{i=1}^{s} T_{labor_i} \tag{6.5}$$

There is an interesting note to using these metrics, and metrics closely related to them, as evaluation criteria. The main idea of using evaluation criteria in the two step approach is to try to be able to capture design alternatives and assembly sequences on the Pareto front without actually having to calculate said Pareto front, at least initially. The evaluation criteria thus act as a sort of surrogate model, with the production time evaluation criteria being a surrogate for finding design alternatives close to the max throughput end of the Pareto front. Meanwhile, the sequencing cost evaluation criteria acts as a sort of surrogate for the design alternatives close to the minimum cost end of the Pareto front.

The sequencing cost and production time metrics are better evaluation criterion than the majority of the other available metrics used in the ASP literature due to being generalized enough to also be able to at least partially account for line balancing without actually balancing a line. This is on top of their ability to be calculated during the ASP analysis in the first place. Due to them being based on metrics that already exist and are regularly used in literature, they will be referred to as default evaluation criteria from now on.

While sequencing cost and production time could prove to be adequate evaluation criteria, they are likely not the best. This is because, although they do consider ALB analysis, they do so rather indirectly. The sequencing cost completely ignores the cost of inefficiency from idle time as well as the costs related to duplicating equipment. Though it is still a cost metric, it is not directed enough towards the type of cost metrics often computed and optimized for during line balancing. In other words, the fidelity of the sequencing cost evaluation criteria and the final cost objective is different enough that, even though they are both the same type of metric, there is only a relatively weak link and correlation between a low sequencing cost and a low final cost. There is not a large enough guarantee that there

is a one-to-one correlation between the lowest sequencing cost designs and the lowest final cost ones. There is a similarly weak link between production time and throughput. High throughput designs will tend to have low total production times. However, the total production time is not indicative of optimal throughput not only because it does not account for workstation assignment, but because the individual task times that make up the total also have a large effect on throughput. As an example, a sequence with a low total production time is likely to have a high throughput. However, if the majority of that production time is spent on only one task that uses a physically very large piece of equipment, it may not be able to be duplicated enough times to actually yield a high throughput. In essence, like with sequencing cost, the correlation between the lowest production time sequences and designs and the highest throughput ones is indirect, making them only somewhat correlated. The default evaluation criteria consequently do little to guide the search towards more line balancing-oriented solutions, which can result in better overall solutions being missed or more time being spent trying to find them and defeating the purpose of the two step approach.

Due to these issues with the default evaluation criteria, it is desirable to develop better ones. Despite this, it is unwise to develop completely new ones from scratch because there are no guarantees about how applicable they are to different problems or situations. As a result, it is better to modify pre-existing criteria and metrics since their performance is much better known. The default evaluation criteria already weakly consider both assembly analyses and so far are the only metrics in the assembly sequencing literature able to do so. They are thus the best metrics to use as a starting point to develop the new criteria. The reason the default evaluation criteria are only adequate instead of good is because they do not adequately incorporate key aspects of the ALB analysis. The issue with the production time metric is that it does not consider workstation assignment or how the individual task times are distributed. The issue with the sequencing cost metric is that it is oblivious to the cost of inefficiency as well as the cost of station duplication. Modifying the default

evaluation criteria to address these issues is thus likely to result in new criteria that are superior in performance.

For the production time criteria, this means it must be changed so that it more closely resembles the throughput metric. Throughput, as outputted in the integrated problem, is highly dependent on how many times the workstations are duplicated and, subsequently, on the available area of the factory as compared to the area taken up by the equipment. In particular, the maximum throughput is most easily obtained by assigning each task into its own workstation and then duplicating each workstation until no space is left, a trend noticeable in Experiment 1 where the number of stations was often equal to the number of available tasks at very high throughputs. This means the new evaluation criteria, while being based on production time, has to be able to incorporate making the individual task times distinct while also being able to look at station duplication and space availability. A possible avenue is to use assumptions related to maximum throughput and leverage key pieces of given information to make a preliminary throughput metric. First, as mentioned, since the max throughput is most easily obtained by having all tasks be in their own stations, the assumption is made that this is indeed the case with the tasks generated by ASP. This solves the issue of ASP not being able to assign tasks to workstations by simply assuming each task is in its own workstation. From this, all station times are known, with the largest station time essentially being the cycle time. Next, each task is assigned its own equipment set and the associated area requirements. This information is available due to the design alternatives having already been selected before the sequencing process was started. If this were not the case, there would not be enough precedence information to perform sequencing in the first place. Afterwards, information on the available space is used to duplicate the tasks and yield a preliminary maximum throughput estimate. This information is also available due to it being part of selecting the design alternatives. Without it, not even the full integrated problem is solvable. This process thus yields a preliminary throughput obtainable from ASP without having to actually perform a line balance. The exact procedure

for this proposed metric, which will be called preliminary throughput, is as follows:

1. Generate fabrication and assembly tasks from the sequence(s) made. All tasks are assigned to their own station.

2. Assign the relevant equipment sets to each task and their work station, calculate each station's required space.

3. Sequentially duplicate bottleneck tasks, reduce corresponding tasks' effective station times, decrement available area appropriately.

4. Once available area is insufficient for further duplication, largest effective station time is the cycle time, take its reciprocal to produce preliminary throughput metric.

The goal is to have all the effective station times be as low as possible for as high of a throughput as possible. The procedure is otherwise extremely similar to how the throughput is calculated when solving the integrated sequencing and line balancing problem, with the main difference being that the line balancing is replaced with an assumption related to trying to calculate max throughput. The resulting metric is thus a throughput-related one except produced in an assembly analysis that normally does not yield throughput-related metrics.

The preliminary throughput metric is more advantageous than the production time metric because it directly relates to line balancing while still being able to be outputted by the sequencing analysis. Due to being a throughput metric, it directly addresses all the issues with the production time metric. By assigning each task an area requirement, the tasks are all made distinct. The area requirement assignments additionally double as a means to account for station duplication and allow for actual production rate estimates. The preliminary throughput metric can thus differentiate between two sequences/design alternatives that have identical production times but with one of them using much larger equipment and having a much more skewed task time distribution. Consequently, there is a significantly

larger chance of a one-to-one optimality equivalence between design alternatives that have an optimal preliminary throughput, which doesn't use line balancing, and design alternatives that have an optimal final throughput in the integrated problem, at least in regard to alternatives with the highest production rates. As a result of being a metric able to consider both assembly analyses in more detail than production time while being able to provide the same or better indication of quality, the preliminary throughput metric is a better evaluation criteria than the production time one and is proposed to be one of the ones used for the two step approach.

In regard to the sequencing cost criteria, it must be changed so that it is able to properly encapsulate the cost of labor inefficiency and opening new and additional workstations. This necessitates changing the sequencing labor, equipment, and tooling cost equations. The final labor cost accounts for labor inefficiency by multiplying all the wage rates by the cycle time. The cycle time is generally longer than the majority of the effective station times, which is what allows the inefficiencies to be captured. To obtain the cycle time, tasks must be assigned to workstations, which the ASP analysis is unable to do. However, the act of line balancing can be replaced with the usage of some assumptions, as was done with the preliminary throughput criteria. In that case, every task was placed in its own workstation to try to minimize cycle time and maximize production rate. In this scenario, something similar must be done to try to minimize cost.

The tasks cannot be directly placed in workstations in a manner designed to minimize labor inefficiency and labor cost; to do so requires the tasks to be assigned such that all the station times are relatively uniform and adaptively so regardless of what the station times actually are. This, given zoning and precedence constraints, is much too complicated to be done via simple assumptions. In fact, trying to assign tasks to stations in this manner to minimize labor inefficiency is akin to trying to solve the bin packing problem, which is a discipline-agnostic version of the ALBP. This is not desirable because the point is to not have to perform line balancing or something as complex in the first place. Consequently,

the tasks must be placed in workstations in a manner that minimizes other types of cost that are simpler to deal with. The other types of cost that make up sequencing cost are the sequencing material cost, sequencing tooling cost, and the sequencing equipment cost. The sequencing material costs do not change regardless of how the tasks are distributed among the workstations while the fabrication tooling costs do not change because all fabrication tasks need their own set of tooling anyway, leaving trying to influence the sequencing equipment cost and the assembly tooling cost as the only option. To minimize equipment cost, as few workstations and their corresponding equipment sets should be opened up or duplicated as possible. This can be done by consolidating as many tasks as possible into as few workstations as possible. Given the presence of zoning constraints, this entails that as many parts be fabricated or assembled together in the same workstation as is allowed. Similarly, to minimize assembly task tooling costs, as many assembly tasks need to be consolidated into as few stations as possible so that the different assembly tasks can share the same set of tooling.

To do this, the same procedure used in subsection 5.4.3 to generate a line balance given an assembly sequence is used with the difference being that, during the task assignments, all tasks able to be placed in the same station are forced to be in the same station as opposed to having a random chance to do so. Additionally, no station duplication occurs to minimize not just the number of stations but their instances as well. Thus, design alternatives and sequences that allow many parts to be fabricated or assembled in one station will have fewer workstations and sets of equipment and tooling needed, resulting in less equipment and tooling costs. Using these assumptions additionally fulfills the intended purpose of assigning tasks to workstations in a non-complex way, subsequently allowing for labor costs that include inefficiencies to be captured. In this situation, since the lack of station duplication makes the station time the same as the effective station time, the longest station time becomes the cycle time and the labor cost as presented for the integrated problem can used instead of the sequencing labor cost. Consequently the resultant sequencing cost, from now

on called the revised sequencing cost to differentiate it from the original sequencing cost that did not account for task assignments, is calculated almost exactly the same way as the final cost for the integrated sequencing and line balancing problem. The only major difference between them is that formal line balancing is not performed and so labor inefficiency cannot be more directly minimized.

The primary benefits of the revised sequencing cost are that, by using basic assumptions to assign tasks to workstations, this metric is able to more appropriately consider key characteristics of the ALB analysis. In being able to look at both cost inefficiencies and the cost of opening up separate workstations, it is more appropriate to use than the default sequencing cost metric. Due to its extreme similarity to the final cost metric in the integrated problem, there is a significantly higher chance that there is a one-to-one optimality equivalence for design alternatives optimized using it. Furthermore, the revised sequencing cost is a metric that is still able to be calculated with just the ASP analysis, making it superior to the original sequencing cost in all ways. Thus, the revised sequencing cost metric will be used over the original sequencing cost as the cost-based evaluation criteria in the two step approach.

6.2 Hypothesis 3: Two Step Approach and Representative Metrics

The two step approach is pursued because it shows promise in being able to reduce the problem size of the integrated problem by filtering out the majority of the less optimal designs in the combinatorially and computationally less-expensive first step before the rest of the design alternatives are passed into the more expensive integrated problem in the second step. This can save large amounts of computational time and is a generalizable way to improve the efficiency of tackling the integrated problem while still allowing for future algorithm improvements or HPC augmentation. To be able to do this, the first step of the two step approach, which only consists of the aircraft design and ASP optimization, must be able to generate optimal assembly sequences that also generate optimal line balances

in the second step. There must be an optimality equivalence between the two analyses in the two steps: the best theoretically possible assembly line balances that generate the optimal throughput and cost solutions will have assembly sequences associated with them, and those assembly sequences must be among the assembly sequences that the first step produces so that those line balances can actually be achieved in the second step.

Previous works tried to do this by optimizing their assembly designs for metrics for both ASP and ALB. The goal of the two step approach is to do the same and optimize the assembly sequences in the first step using metrics representative of both ASP and ALB to ensure there is optimality equivalence in the first step's optimal sequences. However, most of the metrics used by previous works are not applicable to the first step because the ones that are able to be obtained during ASP do not consider ALB. The only ones that do are related to the production time and production cost as they are able to be estimated during ASP. These were deemed to be lacking due to not incorporating enough key characteristics of the ALB portion of the integrated problem and so are not representative enough. Metrics based on the production time and cost, called preliminary throughput and revised sequencing cost, were then developed.

The preliminary throughput metric takes advantage of the fact that high throughput line balances in the integrated problem tend to have stations composed of only one task to make preliminary line balances during ASP, which is ordinarily not possible. These preliminary line balances all assume that all tasks are always assigned to their own workstation. Using this preliminary line balance, stations are duplicated to fill up the factory space to produce the preliminary throughput based on the lowest effective station time. Essentially, the preliminary throughput metric is a throughput-based metric that incorporates the throughput-maximizing characteristics of ALB while being obtainable during ASP. It being more representative of ALB than the production time metric makes it superior to that metric. The revised sequencing cost likewise assumes a preliminary line balance where all tasks are always in the same station whenever possible. The existence of a line bal-

ance at all, even a preliminary one, allows inefficiency costs from the actual line balancing to be captured. Additionally, all tasks being in as few stations as possible minimizes the capital costs, which are a major cost item, making the revised sequencing cost metric a cost-oriented metric. Based on it being able to better capture the ALBP's costs than the original sequencing cost metric due to the existence of a preliminary line balance that can be done during ASP, the revised sequencing cost is more representative of ALB and is the better metric to use.

It is thus believed that these two metrics, which are representative of both ASP and ALB due to being obtainable in the former while capturing many of the latter's characteristics, are able to ensure optimality equivalence in the two step approach. Indeed, it is believed that any metrics obtainable during ASP that sufficiently capture the ALBP's key characteristics will suffice, with the two mentioned simply being the first known implementations. This leads to the hypothesis:

Hypothesis 3 *If optimal assembly sequencing design alternatives are first obtained using time and cost metrics that consider the line balancing problem's key characteristics, then the optimal results for the integrated aircraft design and assembly problem can be produced from just those alternatives, reducing the problem's size and improving the efficiency with which it can be solved.*

To test this hypothesis, an experiment must be performed where solutions from the two step approach are compared with solutions that do not use it or, in essence, solutions obtained by considering all the analyses all at once instead of in two separate steps. From here on this will be called the "full fidelity" approach. If the two step approach's solutions are roughly equivalent to or better than the full fidelity approach's solutions and they are obtained in less time, then the hypothesis is accepted. The term "time" here refers both to the physical amount of time taken as well as the computational time measured in terms of how many generations of solutions it takes to achieve the final non-dominated solutions.

The hypothesis is rejected if the two step approach's solutions perform considerably worse, meaning the selected metrics were not the proper ones to use, or if they perform roughly the same while also taking roughly the same amount of time, meaning the concept of the two step approach is likely flawed. This is all to try to demonstrate that not only is the two step approach a valid way to improve the efficiency of the integrated ASP and ALB problem, but that the type of metrics used for it are valid as well. This is all carried out in Experiment 2.

To actually perform Experiment 2, the integrated ASP and ALB framework from the previous chapter is first infused with aircraft design capabilities to create the full integrated aircraft design and assembly framework for the full fidelity approach. The full fidelity approach is then used as the baseline with which to implement the two step approach. Though the full fidelity approach's framework does incorporate aircraft design and assembly analysis, it is not necessarily the final product of this entire work. This work's final product aims to try to fill Implementation Gap 3 and be able to carry out the entire analysis in a timely manner, meaning the full fidelity approach would only be the final product if Implementation Gap 3 proved to be impossible to address. How the aircraft design capabilities are infused and how the two step approach is implemented based on the resulting full fidelity approach is described in the next section. Experiment 2 is set up and carried out in section 6.4.

6.3 Infusing Aircraft Design into the Integrated ASP and ALB Framework

The developed framework from the previous chapter encoded and operated on its design variables via a GA. Infusing aircraft design capabilities requires the same to be done, but only for the continuous variables and categorical ones that do not affect the interference matrices. This is because aircraft design has many categorical variables, such as spar shape or rib shape or stringer shape, that can dramatically change the interference matrices used to generate the assembly sequences. As was described in subsection 5.5.1, line balances can

only undergo crossover when they share the same precedence matrices via having the same assembly sequence or they must use a constructive crossover procedure like the guided search technique. Even when using the guided search technique though, the line balances still had to use a different inheritance scheme in the developed framework than is typical to make sure the new child line balances actually inherited genes. In a similar manner, assembly sequences undergoing crossover need to have the same interference matrices. However, in the case of assembly sequences, even the guided search technique is not enough when the interference matrices are different because assembly sequences do not have another inheritance scheme they can use to make sure the child sequences can actually inherit any genes when their parents' interference matrices are very different and conflicting. Indeed, the guided search technique itself was built on the assumption that the assembly sequences' precedence relations are the same during crossover. Including and encoding categorical geometric aircraft design variables in the GA is thus equivalent to randomly generating new solutions during the crossover step, which runs opposite to the purpose of a GA. As such, categorical geometric aircraft design variables that can significantly affect interference matrices are not included in the GA and every such variable or change in such a variable requires a new problem instance be run until a future work is able to address this issue.

Because the assembly sequences and line balances are all real-coded instead of binary coded, and given that all the aircraft design variables included in the GA in this work are continuous, the included aircraft design variables will also be real-coded. Given a aircraft design variables, the aircraft design chromosome is a $1 \times a$ vector. For the initial population, the aircraft design chromosome is generated by randomly picking a value within the range of the variable for each gene. For crossover, the whole arithmetic crossover operator described by Picek et al. is used due to its extreme simplicity and being one of the better real-coded operators available [249]. For mutation, if it is decided that an aircraft design chromosome will undergo mutation, every gene has a chance to mutate, with at least one gene guaranteed to undergo mutation. The genes are mutated using a real-coded Gaussian

mutation scheme described by Deb and Deb [250].

Changing the aircraft design variables as opposed to keeping them constant like in Experiment 1 in the previous chapter means that the aircraft must be resized every time such a change is made. This includes determining the aircraft's new performance characteristics as well as determining the resultant size of its parts such that it is still structurally sound. Changing the size of the parts also results in new task times and costs needing to be calculated as opposed to being able to use a single set of times in costs. This is all done by using an aircraft design program or tool suite for the performance characteristics, a structural sizing program, tool suite or framework to determine the new size of the parts, and a parametric cost and time estimation tool for the time and cost estimation. Additionally, these programs, tools, and frameworks will need to be called numerous times during the course of the optimization process. These can be computationally expensive to run and bog down the optimization process. As such, surrogate modeling, which was discussed at the beginning of this chapter, is used to replace these modules during the optimization itself so that the results of these modules can be obtained in just fractions of a second as opposed to a few hours or days.

An important aspect of aircraft design is that the aircraft must meet specific performance constraints. How these constraints are handled in the GA is that every time a new design is generated as a result of the initial population generation, crossover operator, or mutation operator, the performance characteristics of the aircraft are checked against the desired performance constraints. If the aircraft does not meet the constraints, the crossover operator or mutation operator or initial generation is repeated until a feasible aircraft design chromosome is produced. After this is done enough times without being able to obtain a feasible aircraft design, a penalty factor is applied to that design's final throughput and cost based on how many constraints were not met and how much they were not met by, with a minimum penalty factor set such that a barely infeasible design still performs worse than a design that is barely feasible but has otherwise worse throughput and cost.

Beyond including the aircraft design chromosome and its subsequent performance constraint handling in the initial population generation and crossover operation, the only major change to the GA procedure of the integrated ASP and ALB framework in the last chapter that turns it into the full fidelity approach's framework is regarding the mutation operation. For the previous chapter's framework, when an individual is selected for mutation, there were two possible mutation types, one where only the line balance was mutated and another where both the assembly sequence and line balance were mutated. When including aircraft design, there are four possible mutation types. The first two are the same as when aircraft design was not included. The third is to only mutate the aircraft design chromosome. The fourth is to mutate the aircraft design chromosome, the assembly chromosome, and the line balance chromosome all at once. These mutations are all done separately so that the perturbations caused by the mutation are not too large if the individual is close to becoming a non-dominated solution, with the chance of a large perturbation still being possible with the fourth mutation type if a large perturbation is needed to get out of a local minimum.

The full fidelity approach's framework is shown in Figure 6.2. The aircraft design initial generation, crossover, and mutation procedures were explained at the beginning of this section. The assembly sequence initial generation procedure was described in List A and the assembly sequence crossover procedure in List E in subsection 5.4.3. The assembly sequence mutation procedure is to just randomly generate another assembly sequence and so List A's procedure is used. The initial assembly line balance generation procedure was described in List B and the crossover procedure in List F in subsection 5.4.3. Similar to the assembly sequence, the assembly line balance mutation procedure is to just randomly generate another line balance and so List B's procedure is used. The station duplication steps used are described in either Figure 5.6 or Figure 5.8, depending on which objective is being pursued.

The two step approach's framework is extremely similar to the full fidelity approach's with only several notable differences. As such, to simplify its visualization, only the two

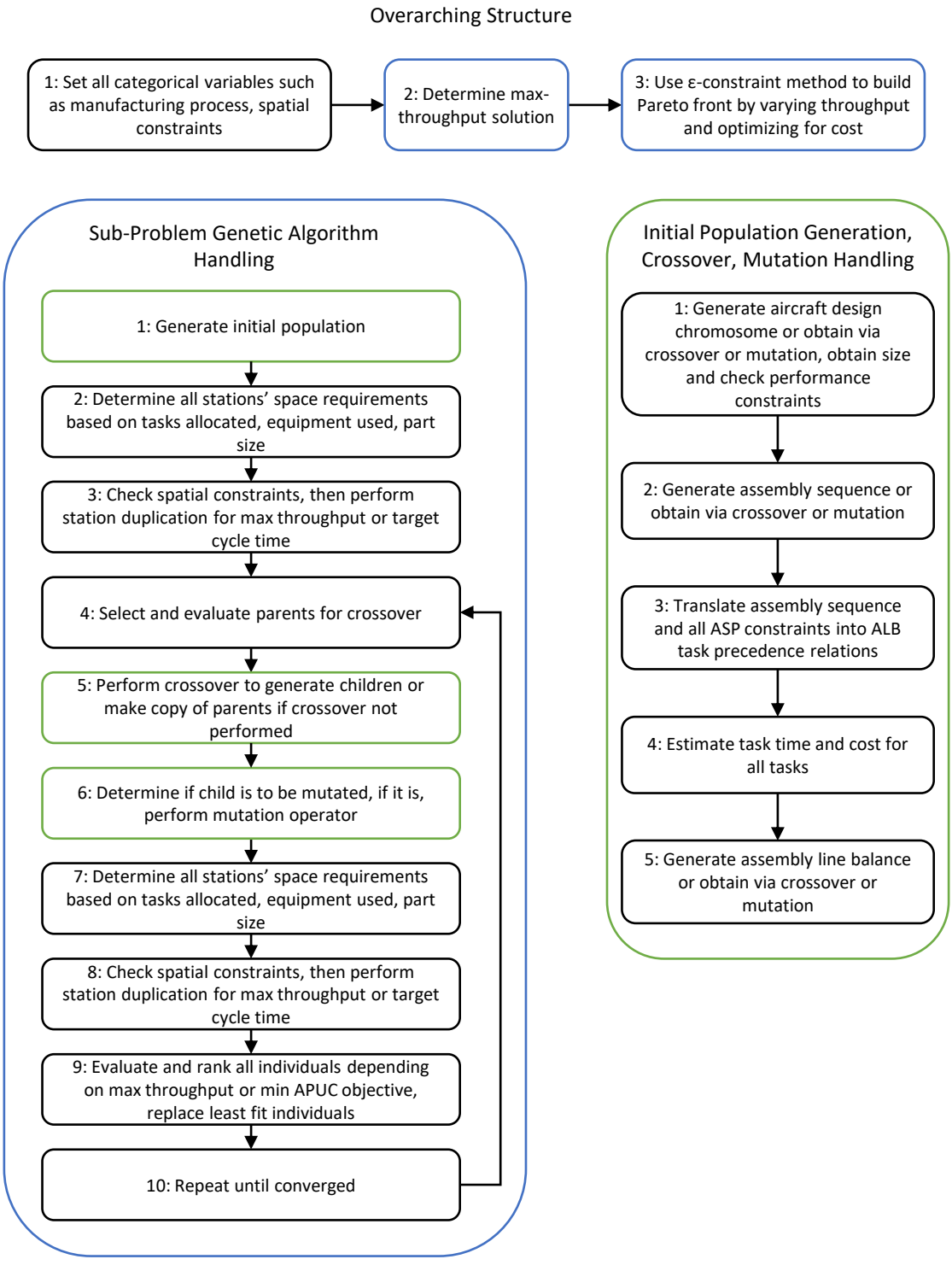


Figure 6.2: Full Fidelity Approach's Framework, Colored Boxes Indicate Different Modules

step approach's framework's overarching structure is shown in Figure 6.3, with the differences in how the handling is implemented noted below.

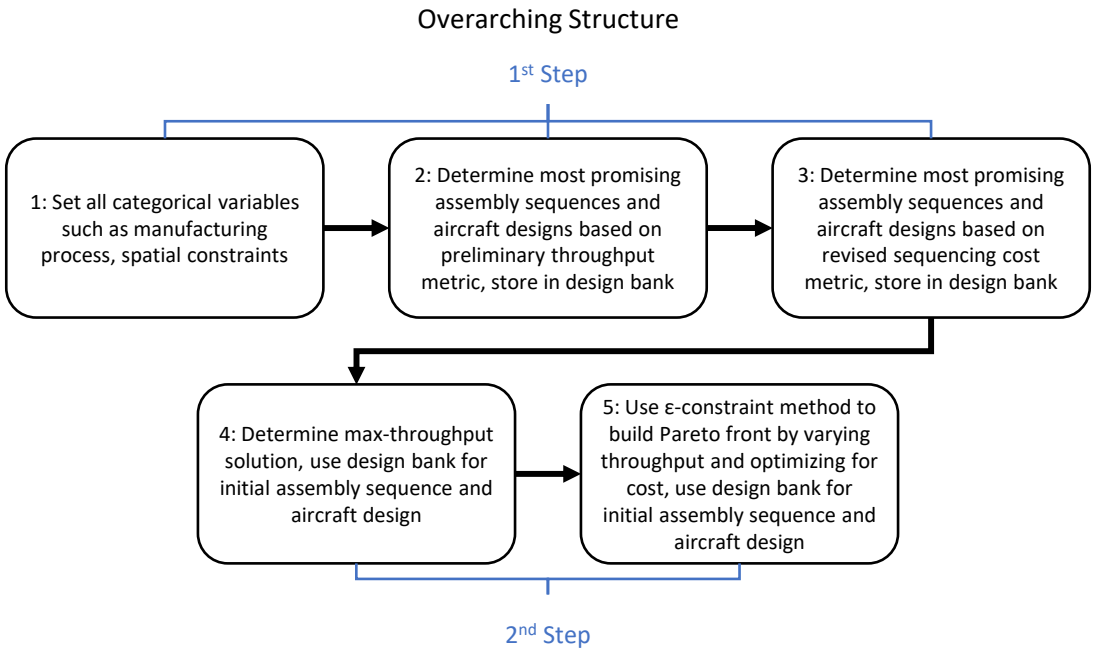


Figure 6.3: Two Step Approach's Framework, Overarching Structure

The primary difference between the full fidelity approach's framework and the two step approach's is that two subproblems are solved in the first step for the two step approach, one to obtain aircraft designs and sequences favorable in terms of the preliminary throughput metric and the other in terms of the revised sequencing cost metric. The GA handling of these subproblems is the same as that in the full fidelity approach except that the initial population is checked to make sure there are no duplicate assembly sequences. In terms of the metrics themselves, solving for the preliminary throughput is basically solving for the maximum throughput like in the full fidelity approach except that all tasks must be in their own station. And solving for the revised sequencing cost is basically the same as solving for the minimum APUC in the full fidelity approach except that no station duplication is performed and all tasks must be in the same station whenever possible. The solutions in both subproblems are ranked and the best ones in each have their aircraft design variables

and assembly sequences stored in the "design bank."

The second step of the two step approach is essentially the same as carrying out the full fidelity approach with two differences. The first difference is that during the initial population creation, individuals can only select assembly sequences and aircraft design values that are stored in the design bank. The second difference is that the stopping criteria requirement is significantly lowered; since the two step approach is supposed to have already significantly reduced the problem size, it is also supposed to converge more quickly, meaning it does not have to spend as long trying to determine if it is in a local minimum or a global one. This is where the efficiency benefits are realized if the two step approach is able to function as proposed. The mutation and crossover operators are still free to generate assembly sequences and aircraft designs not included in the design bank in case they somehow come across even better designs not obtained in the first step.

6.4 Experiment 2: Two Step Approach and Full Fidelity Approach Comparison

6.4.1 General Setup

Experiment 2 consists of comparing the two step approach and the full fidelity approach and seeing if the Pareto fronts generated by the former are of comparable quality to the latter or better while being more efficient and using less time or computational resources. Its purpose is to both show that the two step approach is a valid way to improve the efficiency of the integrated aircraft design and assembly problem and that the type of metrics being used as part of the two step approach aid with that efficiency improvement. To determine the quantitative quality of the Pareto fronts, the same quality indicators as Experiment 1 are used in the form of the $I_\epsilon(PF_1, PF_2)$ and HVR. Summary attainment surfaces are used to again visualize the average Pareto fronts attained by each approach's framework. The number of generations required to attain the Pareto fronts is used as the primary indicator of how computationally expensive each framework with, with the physical time being a secondary indicator to establish how long each framework actually takes to finish.

It is likely that the distribution of the $I_\epsilon(PF_1, PF_2)$ and $I_\epsilon(PF_2, PF_1)$ values will not be as clear cut in favor one approach or the other, as was the case in Experiment 1. As such, The Wilcoxon rank-sum test will be used to further analyze the distribution of the I_ϵ values. The Wilcoxon rank-sum test, or Mann-Whitney U test, is a non-parametric statistical test used to see if two distributions have the same median or not. Having the same median in this case means the I_ϵ values, and thus the quality, of the two approaches and frameworks are roughly the same. If the medians are not the same, two more one-sided Wilcoxon rank-sum tests are performed, one left-sided and one right-sided, to determine which way the distributions are shifted to confirm which framework has the higher quality. The Wilcoxon rank-sum test is used because it is unknown if the distributions of the I_ϵ values are normal or not, which prevents the use of a normal two-sample t-test. It has also been widely used for comparing different evolutionary algorithms and their indicators [251] and demonstrated good results in works such as Chica et al.'s [166].

The F-86 experimental platform from Experiment 1 will also be used for Experiment 2 because it is an assembly that has all the of the material factors needed for both sets of frameworks to operate on while also having aircraft design aspects that can be changed and optimized as well. No other assemblies in the literature have these unique traits to make them viable candidates as an experimental platform.

6.4.2 Implementation Details

Due to the inclusion of aircraft design, several changes are made to the F-86 experimental platform as compared to how it was used in Experiment 1. The general changes were explained in section 6.3 and so the changes as specifically pertinent to the F-86 are covered here. The F-86 already had to be sized via aircraft design and structural optimization in Experiment 1 due to available real-world F-86 data not having available data on some needed dimensions, and so the usage of the FLOPS program and RADE framework were already mentioned. The usage of the FLOPS program had to be changed slightly because

changing some of the design variables, such as its range, necessitated changing how the F-86's baseline design mission was defined. This is covered further in section A.6. Essentially, a parametric design mission is used instead of the baseline one to size the F-86, with the specific performance parameters in that parametric design mission mostly set and optimized by FLOPS itself based on the variable values inputted.

In Experiment 1, the FLOPS program and RADE framework did not have to be called numerous times whereas in Experiment 2 they will be called likely thousands of times. FLOPS can take a few seconds to size a single aircraft design, which can already be computationally prohibitive, but the RADE framework is even worse and can take upwards of 90 minutes to produce a single structurally optimized wingbox. It is impractical to call on the actual program or use the framework during the optimization process. Similarly, SEER will have to be called just as many times due to the changing wingbox dimensions necessitating the task times and cost to constantly be re-estimated. SEER is in between the FLOPS program and RADE framework in terms of computational expense, taking 40 seconds to yield the necessary outputs for a single design. Regardless, it is still untenable to directly call SEER while optimizing.

As was mentioned in section 6.3, surrogate models were created for all three programs so that their computation expense does not become problematic. The details for this are in section A.6. To summarize, a Design of Experiments was created for them using Box-Behnken and Latin Hypercube sampling to select the input datapoints. Artificial Neural Networks were then used as the surrogate models due to their ability to capture non-linear interactions. The resulting surrogate models are quite accurate, with the lowest training and validation R^2 values still being around 0.95 and the majority have values upwards of 0.99 with no noticeable patterns in the residuals. Using the surrogate models, the desired results take a maximum of 0.04 seconds to obtain as opposed to possibly upwards of 90 minutes before they were used.

Another difference that now exists is that a categorical aircraft design variable that af-

fects the interference matrices will be considered in the form of the spar's shape. The spar shape is the aircraft design variable chosen because it is a high level variable that affects both the aircraft design and assembly disciplines. Specifically, it affects the assembly discipline by changing which sequences are feasible and which are not and affects the sequences' accessibility as well. It is additionally one of the highest level variables that is able to do so. On the aircraft design side, the spar shape alters how much material is used by the spars by determining how efficiently the spar is able to handle loads during the structural sizing, which affects the aircraft's weight and therefore its performance. How this is accounted for in the structural optimization is documented in section A.6, but amounts to reorienting some of the structural elements used to represent the spar.

How the spar shape affects the ASP geometric reasoning process is also detailed in section A.6. The most notable effect is that, due to the assumption of the size of the assembly not affecting the interference matrices, it means that interference matrices between differently-sized assemblies can be shared if they have the same spar shape. The spar shape also has a significant impact on an assembly sequence's accessibility. This impact is captured in terms of task time by making the appropriate changes in SEER, which has an option to change the accessibility of a task and therefore increase its labor time if it is less accessible. This, even more so than Experiment 1, demonstrates the impact of Conjecture 1.2's implementation.

6.4.3 Additional Setup Details, Design Variables, and Procedure

Before Experiment 2 can be carried out, several additional parameters need to be defined. For the aircraft design Gaussian mutation scheme, a mutation strength of 1/30 was used like in Deb and Deb's work to ensure that, while the variables are likely to mutate only a short distance away from their original values for small perturbations, it is still not unlikely that they can make larger jumps further away if it is needed to escape a local minimum [250]. In Experiment 1, 5 Pareto fronts were generated at every categorical variable setting to obtain

the "average" Pareto front–in Experiment 2, 10 will be generated to obtain a greater spread of quality indicator values with which to compare the two frameworks. 10 was considered sufficient in Chica et al.'s work [166]. The stopping criteria remain the same for the full fidelity approach's framework. It is also the same for the two step approach's first step. For the second step, however, due to the usage of promising assembly sequences and aircraft design configurations, the problem is likely to converge faster and so the stopping criteria is the optimization is halted after only 3 generations with less than 1% improvement over the previous best individual, as opposed to 10. This is where the efficiency improvements are expected to appear, if they do at all.

When comparing Hypervolume Ratios, a reference point is needed to calculate the hypervolume. For the Hypervolume Ratio calculation, the reference point APUC is set to be 1% higher than the highest cost solution of both frameworks' solutions for each variable combination. This is roughly the same amount used by Knowles so that the Hypervolume Ratio does not behave erratically and produce unexpected results [252]. The reference monthly throughput is set to 0. Due to there not being an established true Pareto front, the best solutions from both approach's frameworks at each of the variable combination settings are treated as the true Pareto front. All other parameters are the same as they were in Experiment 1. For the Wilcoxon rank-sum test, the level of significance for all the tests is $p = 0.05$.

The design variables for Experiment 2 and their ranges are presented in Table 6.2.

Of note among these variables and their ranges is that additional ranges are available for the factory space and spar shape variables, but only the ones indicated are used because preliminary tests indicate they provided the most extreme results and so bound the integrated aircraft design and assembly problem as it pertains to the F-86's wingbox. Regarding the spar shapes themselves, a more detailed visual of what the shape name signifies is presented in section A.6. For the aircraft design variables, the specific variables and their ranges were chosen based off of what was done in Siedlak et al.'s work [29]. The wing

Table 6.2: Design Variables and Ranges for Experiment 2

Variable	**Range**
Wing Area (ft^2)	240-410
Aspect Ratio	3.84-6.24
Taper Ratio	0.36-0.66
Range (nmi)	600-1000
Factory Size (ft^2)	333k, 1M
Spar Shapes	Out-C, I
Manufacturing Process	HLU/Autoclave, ATL, VARTM, SRI, CNC Machined
Assembly Sequence	All Feasible Sequences
Assembly Line Balance	All Feasible Line Balances
Integral Fabrication & Sub-assembly Part Combinations	3 Major Parts Max

sweep is not included due to being set for aircraft design for performance reasons.

The Automated Tape Layup (ATL)/Autoclave manufacturing process, called ATL from now on in this work, is included to complete the list of processes for which there is available data on in the F-86 experimental platform. It, like the HLU/Autoclave process, is selected because it is one of the baseline composite manufacturing processes that other composite processes are compared to. In the ATL process, the plies are laid down flat via an automated tape layup machine. Because the plies are relatively flat, they then require a hot drape forming task to form them into the right shapes for all parts except for the wingskins. The plies are afterwards cured in an autoclave before being trimmed via waterjet trimming machine and then inspected via ultrasonic NDI. A more detailed description of its tasks, how it is modeled, and its modeling assumptions is in section A.3. The relevant If-Then precedence rules for the ATL process are in Appendix C while its equipment space, equipment cost, and factory space modeling details and assumptions are in Appendix E.

The procedure for Experiment 2 is relatively simple. The full fidelity approach and two step approach's frameworks are carried out for every combination of the 4 manufacturing processes, 2 spar shapes, and 2 factory sizes. 10 repetitions are done for each combination

to obtain a noticeable spread of quality indicators. Additionally, a truncated full fidelity approach is also carried out where the stopping criteria is the same as the two step approach's to see the results of halting the optimization after only 3 generations with significant improvements. This is to see how the results in the full fidelity approach fare when trying to be as computationally efficient as the two step approach but without having access to the most promising assembly sequences and aircraft design configurations. All the frameworks' results for each combination are then compared in terms of the quality indicators and computational expense metrics specified in section 6.4. Hypothesis 3 is rejected if the two step approach's results perform worse in all the combinations or perform the same with the same computational expense in all the combinations as compared to the full fidelity approach and the truncated full fidelity approach. Hypothesis 3 is accepted if the two step approach's results perform better in all the combinations or perform the same with less computational expense in all the combinations as compared to the full fidelity approach. Hypothesis 3 is partially accepted to varying degrees for all other cases, with the level of it being partially accepted depending on the exact outcomes.

All of the same assumptions used Experiment 1 are also used in Experiment 2.

6.4.4 Results and Discussion

The results for the ATL process are in Figure 6.4 and Figure 6.5, the SRI process in Figure 6.6 and Figure 6.7, the VARTM process in Figure 6.8 and Figure 6.9, the HLU/ Autoclave process in Figure 6.10 and Figure 6.11, and the CNC Machining process in Figure 6.12 and Figure 6.13. Only the average summary attainment surface is included for each set of results as opposed to the best one because it is more important to see the average Pareto front the frameworks are able to generate as opposed to the absolute best ones, which can be skewed heavily by randomly chancing upon a very optimal solution just one time among many. The HVR, computational expense values, and statistic test values are presented in Table 6.3. In the two-sided test, a p-value of less than 0.05 indicates the null

hypothesis that the two frameworks have similar median $I_\epsilon(PF_1, PF_2)$ values, and therefore similar quality, is rejected. If it is rejected, the left and right-sided tests determine which way the median is skewed. In the left-sided test, a p-value below 0.05 indicates the null hypothesis is rejected in favor of the alternative hypothesis that the two step approach's framework yields higher quality solutions. In the right-sided test, a p-value below 0.05 indicates the null hypothesis is rejected in favor of the alternative hypothesis that the full fidelity approach's framework yields higher quality solutions. A p-value above 0.05 in either one-sided test indicates that the alternative hypothesis for that test is not favored.

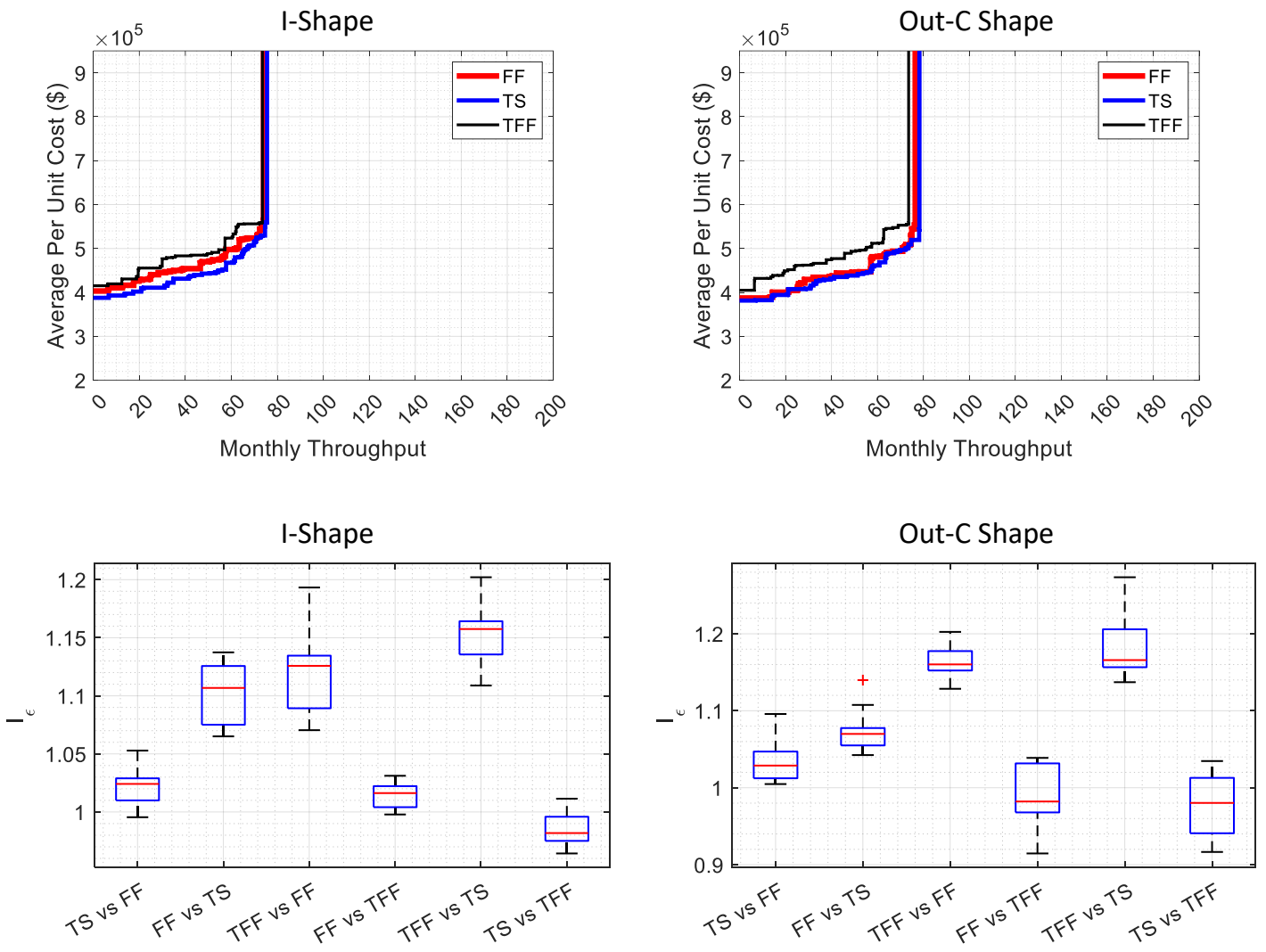


Figure 6.4: Average Summary Attainment Surface and Boxplots Showing $I_\epsilon(PF_1, PF_2)$ Distributions Comparing Two Step (TS), Full Fidelity (FF), and Truncated Full Fidelity (TFF) Approaches for ATL Process, 333k ft^2 Constraint

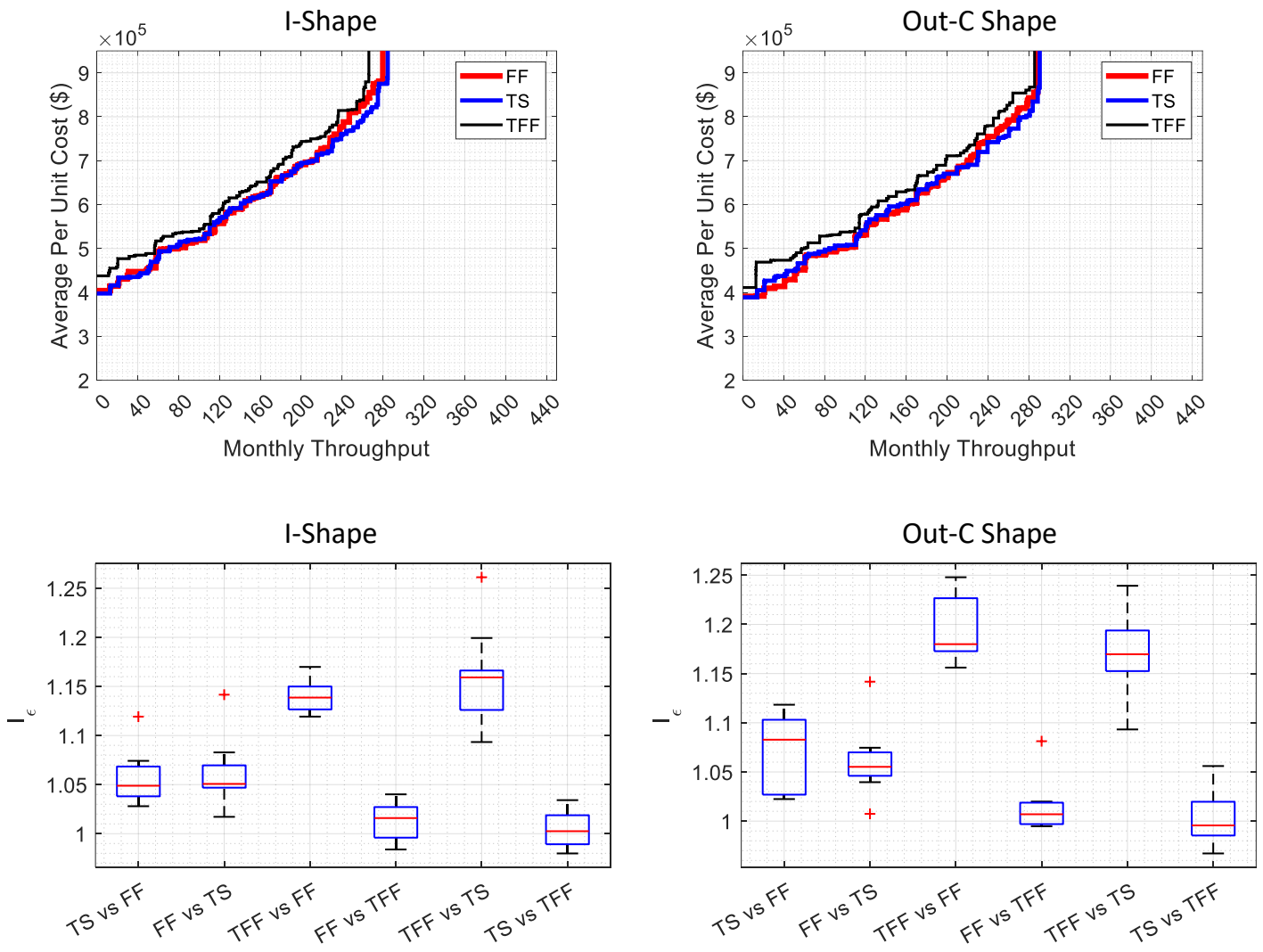


Figure 6.5: Average Summary Attainment Surface and Boxplots Showing $I_\epsilon(PF_1, PF_2)$ Distributions Comparing Two Step (TS), Full Fidelity (FF), and Truncated Full Fidelity (TFF) Approaches for ATL Process, 1M ft^2 Constraint

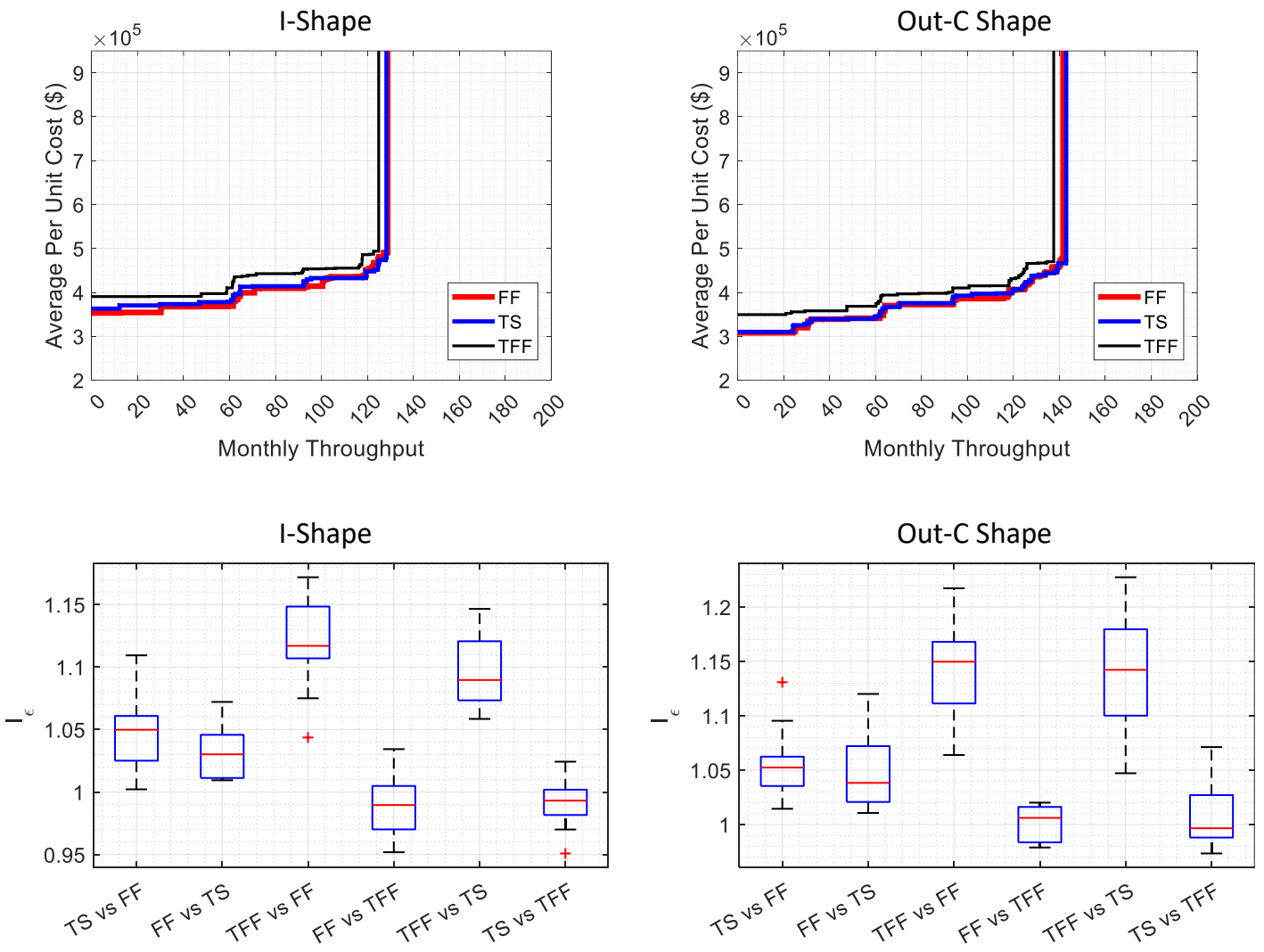


Figure 6.6: Average Summary Attainment Surface and Boxplots Showing $I_\epsilon(PF_1, PF_2)$ Distributions Comparing Two Step (TS), Full Fidelity (FF), and Truncated Full Fidelity (TFF) Approaches for SRI Process, 333k ft^2 Constraint

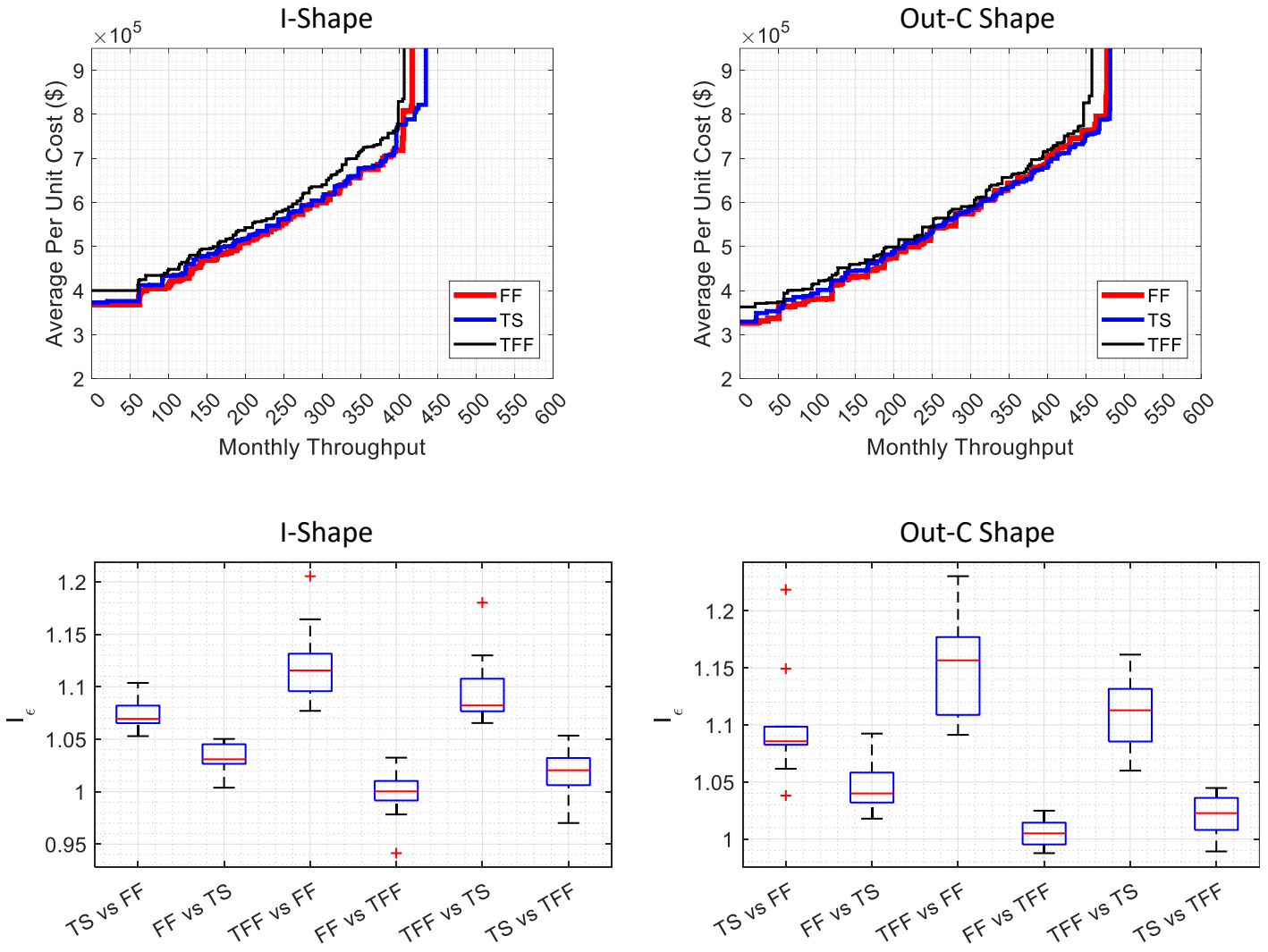


Figure 6.7: Average Summary Attainment Surface and Boxplots Showing $I_\epsilon(PF_1, PF_2)$ Distributions Comparing Two Step (TS), Full Fidelity (FF), and Truncated Full Fidelity (TFF) Approaches for SRI Process, 1M ft^2 Constraint

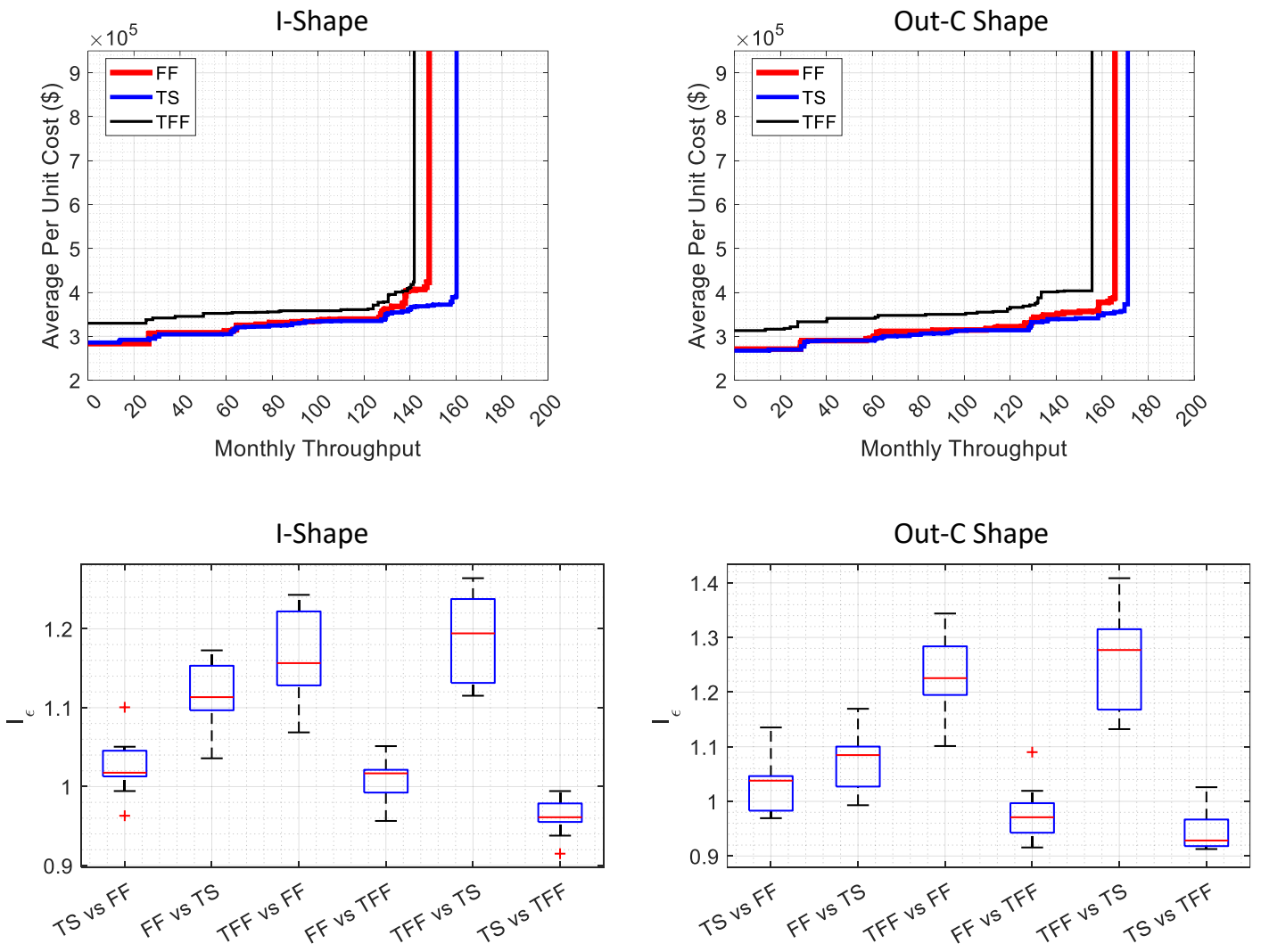


Figure 6.8: Average Summary Attainment Surface and Boxplots Showing $I_\epsilon(PF_1, PF_2)$ Distributions Comparing Two Step (TS), Full Fidelity (FF), and Truncated Full Fidelity (TFF) Approaches for VARTM Process, 333k ft^2 Constraint

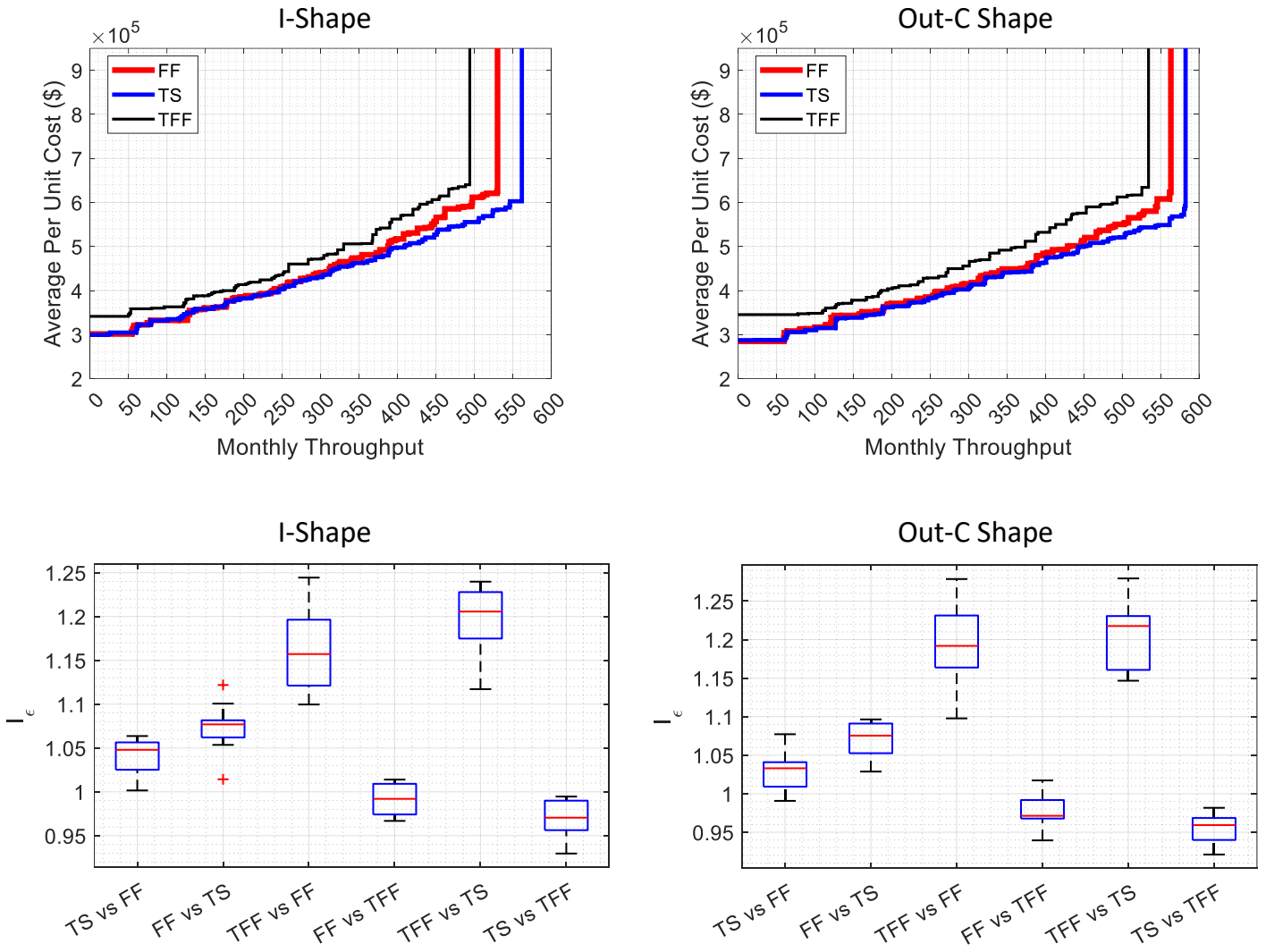


Figure 6.9: Average Summary Attainment Surface and Boxplots Showing $I_\epsilon(PF_1, PF_2)$ Distributions Comparing Two Step (TS), Full Fidelity (FF), and Truncated Full Fidelity (TFF) Approaches for VARTM Process, 1M ft^2 Constraint

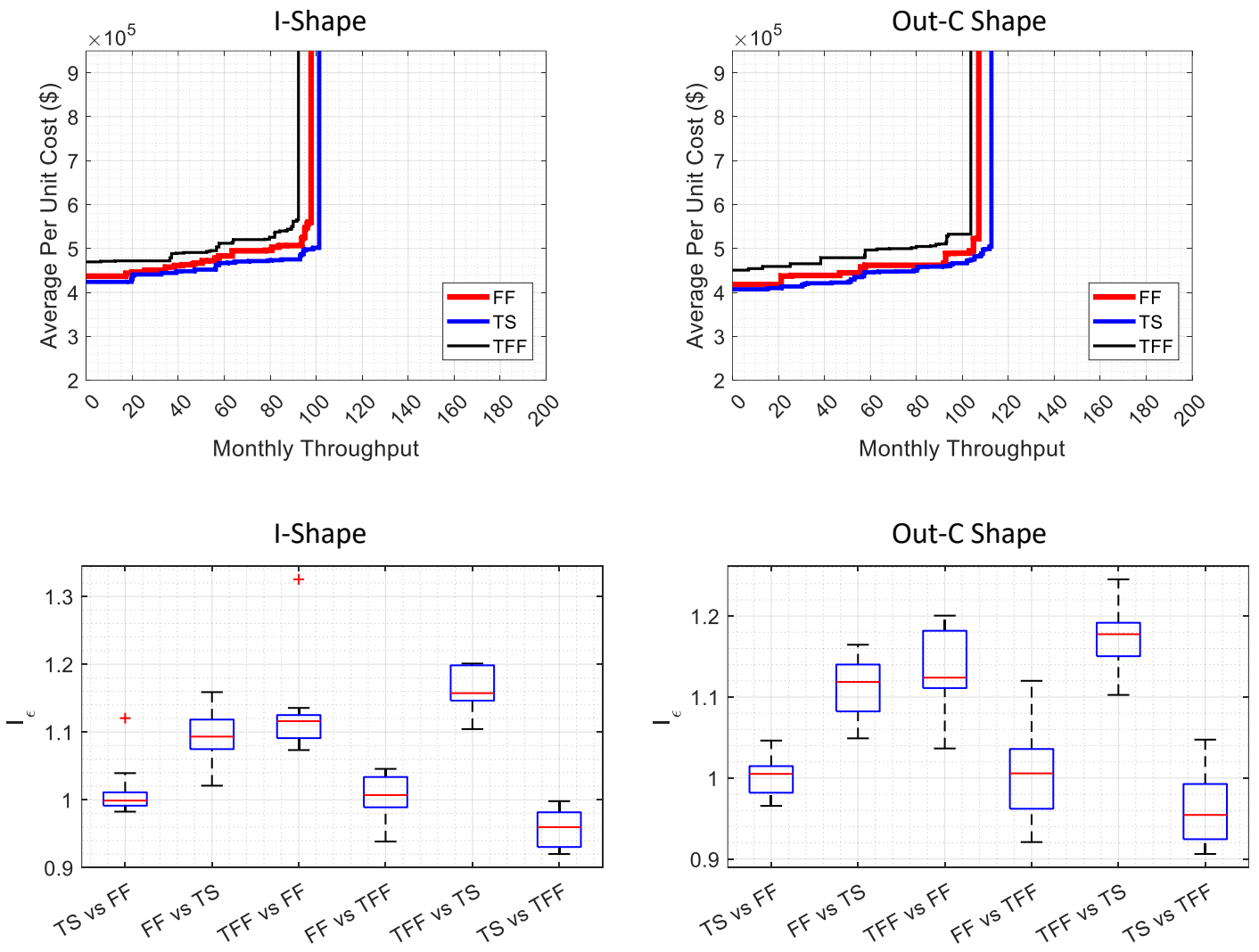


Figure 6.10: Average Summary Attainment Surface and Boxplots Showing $I_\epsilon(PF_1, PF_2)$ Distributions Comparing Two Step (TS), Full Fidelity (FF), and Truncated Full Fidelity (TFF) Approaches for HLU/Autoclave Process, 333k ft^2 Constraint

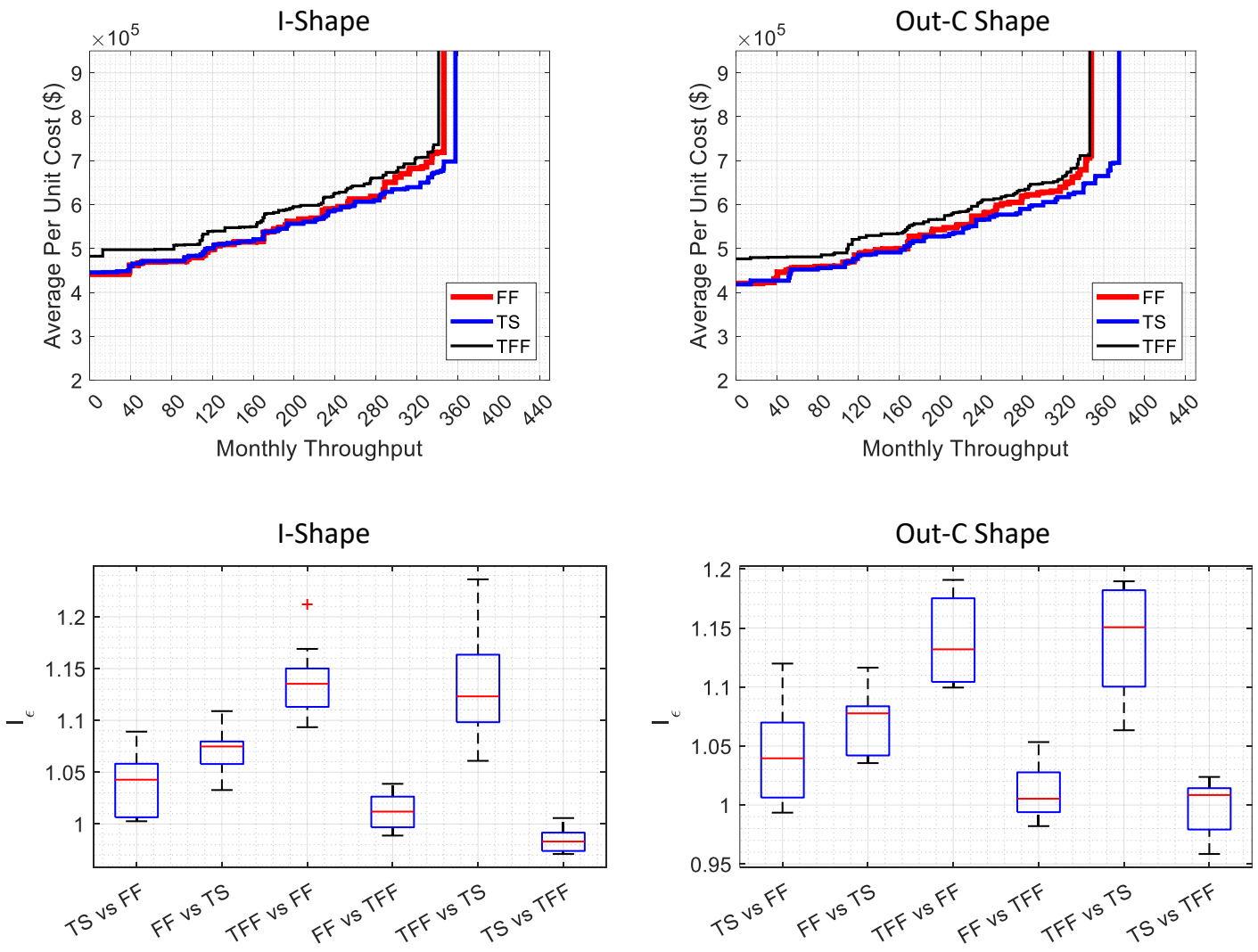


Figure 6.11: Average Summary Attainment Surface and Boxplots Showing $I_\epsilon(PF_1, PF_2)$ Distributions Comparing Two Step (TS), Full Fidelity (FF), and Truncated Full Fidelity (TFF) Approaches for HLU/Autoclave Process, 1M ft^2 Constraint

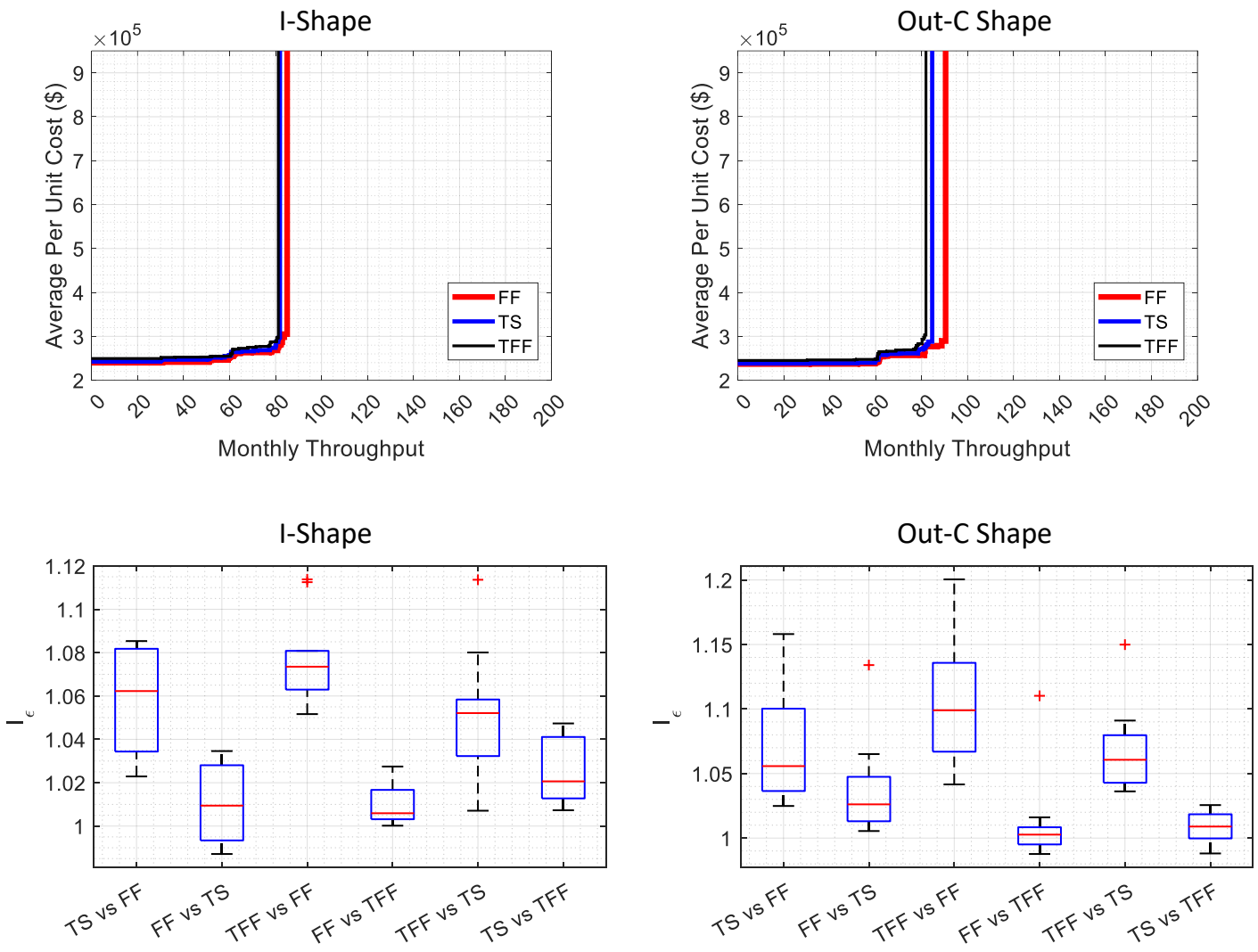


Figure 6.12: Average Summary Attainment Surface and Boxplots Showing $I_\epsilon(PF_1, PF_2)$ Distributions Comparing Two Step (TS), Full Fidelity (FF), and Truncated Full Fidelity (TFF) Approaches for CNC Machined Process, 333k ft^2 Constraint

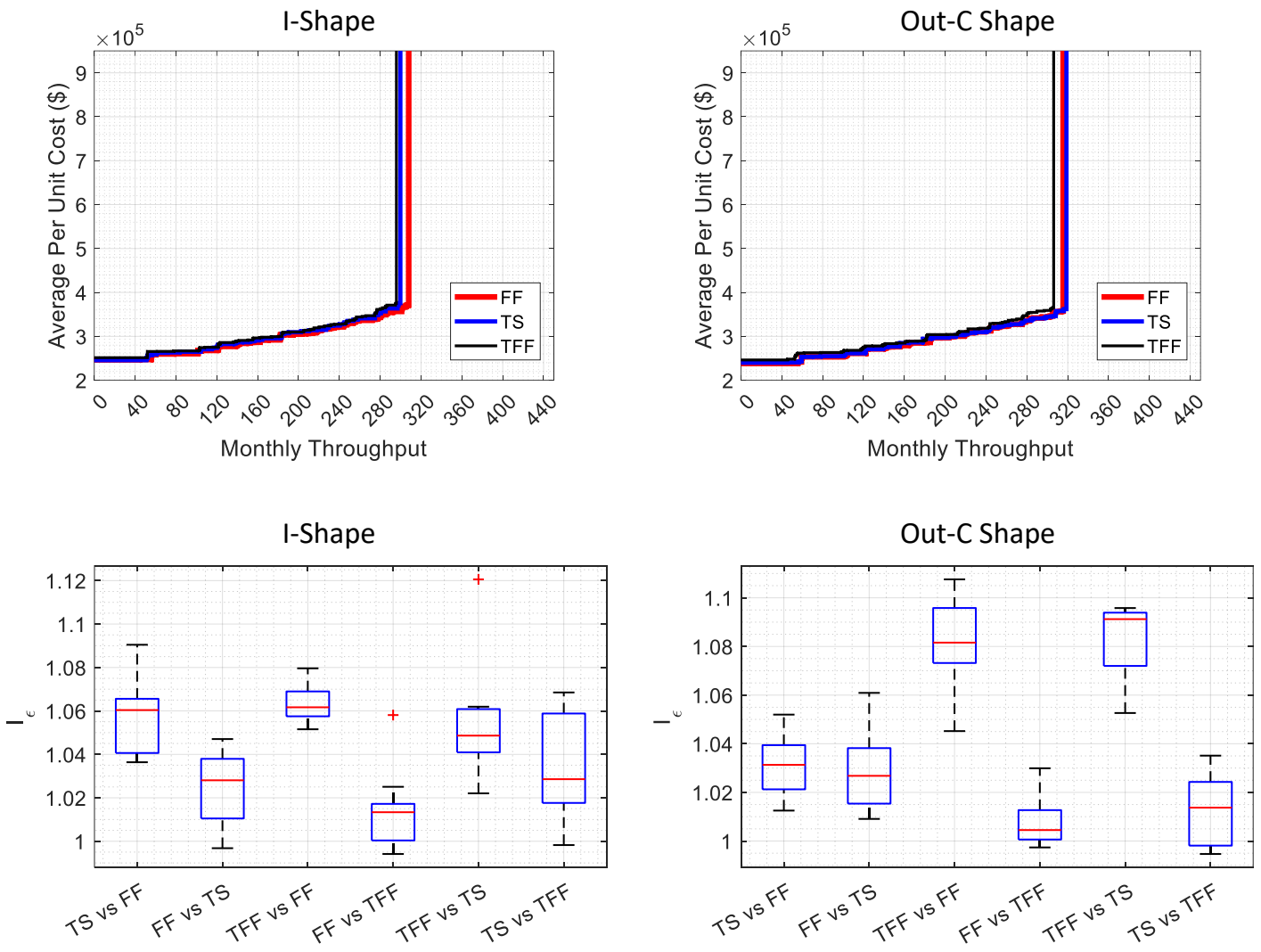


Figure 6.13: Average Summary Attainment Surface and Boxplots Showing $I_\epsilon(PF_1, PF_2)$ Distributions Comparing Two Step (TS), Full Fidelity (FF), and Truncated Full Fidelity (TFF) Approaches for CNC Machined Process, 1M ft^2 Constraint

Table 6.3: Summary of Comparison of Two Step (TS) Approach's Framework's Results with Full Fidelity (FF) Approach's Framework's Results. HLUA is HLU/Autoclave, CNCM is CNC Machined

Manufacturing Process, Shape, Space Constraint (ft^2)	**TS HVR**	**TS Average Num of Gens, Time (s)**	**FF HVR**	**FF Average Num of Gens, Time(s)**	**TFF Average Num of Gens, Time(s)**	**Two-Sided P Value**	**Left-Sided P Value**	**Right-Sided P Value**
ATL, I, 333k	0.9982	239, 1623	0.8998	487, 3025	193, 1259	0.0002	$9 * 10^{-5}$	0.9999
ATL, Out-C, 333k	0.9931	232, 1965	0.9520	502, 3875	183, 1792	0.0058	0.0029	0.9977
ATL, I, 1M	0.9955	259, 1836	0.9803	534, 3477	206, 1637	0.6776	N/A	N/A
ATL, Out-C, 1M	0.9869	244, 2206	0.9741	540, 3969	207, 1891	0.3075	N/A	N/A
SRI, I, 333k	0.9809	226, 1484	0.9895	462, 2897	180, 1398	0.2730	N/A	N/A
SRI, Out-C, 333k	0.9850	227, 2028	0.9639	490, 3664	178, 2166	0.3447	N/A	N/A
SRI, I, 1M	0.9672	242, 2021	0.9967	505, 3575	210, 1970	0.0002	0.9999	$9 * 10^{-5}$
SRI, Out-C, 1M	0.9906	243, 1880	0.9803	508, 3841	202, 2424	0.0036	0.9986	0.0018
VARTM, I, 333k	0.9958	245, 1285	0.9137	479, 2560	180, 1154	0.0008	0.0004	0.9997
VARTM, Out-C, 333k	0.9972	245, 1755	0.9387	496, 3372	182, 1726	0.1212	N/A	N/A
VARTM, I, 1M	0.9942	251, 1326	0.9662	528, 2930	201, 1394	0.0058	0.0029	0.9977
VARTM, Out-C, 1M	0.9936	263, 1778	0.9785	516, 3425	197, 1792	0.0036	0.0018	0.9986
HLUA, I, 333k	0.9958	230, 1359	0.9125	461, 2516	179, 1113	0.0022	0.0011	0.9991
HLUA, Out-C, 333k	0.9991	231, 1851	0.9249	464, 3231	181, 1531	0.0002	$9 * 10^{-5}$	0.9999
HLUA, I, 1M	0.9991	233, 1509	0.9408	481, 2858	192, 1352	0.064	N/A	N/A
HLUA, Out-C, 1M	0.9976	233, 1771	0.9525	505, 3408	188, 1653	0.1620	N/A	N/A
CNCM, I, 333k	0.9204	202, 770	0.9965	444, 1914	170, 787	0.0017	0.9993	0.0009
CNCM, Out-C, 333k	0.9250	206, 1170	1.00	438, 2255	169, 991	0.0376	0.9844	0.0188
CNCM, I, 1M	0.9720	213, 922	0.9929	459, 2051	187, 996	0.0017	0.9993	0.0009
CNCM, Out-C, 1M	0.9865	212, 1229	0.9950	459, 2306	185, 1081	0.6776	N/A	N/A

It is evident that, due to the way it was designed, the two step approach always takes less time to finish than the full fidelity approach. The amount of time saved, and therefore the efficiency obtained, is generally a factor of two: the two step approach takes half as many generations to finish and is twice as fast or twice as efficient as the full fidelity approach. The question, then, is regarding the quality of its results and whether they are worth the reduced run time. To start, it is noted that the full fidelity approach always has better results than the truncated full fidelity approach. The only difference between the two is the number of generations that must pass without significant improvement, meaning that that factor is very important in determining the optimality of the results obtained. It also indicates that the integrated aircraft design and assembly problem is not one where an optimal solution can be found immediately.

Another important observation related to this is that the two step approach also always yields better results than the truncated full fidelity approach, using roughly 33% more generations for costs and throughputs that are up to 25% better. Its average summary attainment always encapsulates the truncated full fidelity's. It at least indicates that if efficiency is critical, the two step approach should be used over simply just stopping the optimization process earlier. As the main difference between the two is the fact that the two step approach uses representative metrics while the truncated full fidelity one does not, it also indicates that the representative metrics are useful.

Turning to the comparison between the two step approach and the full fidelity one, the most immediately noticeable results are that the two step approach is generally able to create Pareto fronts with lower $I_\epsilon(PF_1, PF_2)$ values than the full fidelity approach for the HLU/Autoclave and VARTM processes. This is most evident with the VARTM process's p-values and HVR for the two-step approach, which consistently indicate that the two step approach is not only more efficient but is actually able to obtain better cost and throughput solutions. The same is true with the HLU/Autoclave process at the lower spatial constraint setting. With the higher spatial constraint setting, the p-values indicate that the null hypoth-

esis is not rejected, but examining the average summary attainment surfaces shows that the two step approach's Pareto front for the is generally at least as good if not usually better than the full fidelity approach's. Given the two step approach's added bonus of greater efficiency, it is overall beneficial to use it over the full fidelity approach at the larger factory constraint setting anyways.

The two step approach is less overtly beneficial but still useful for the ATL process. It performed better than the full fidelity approach at the lower spatial constraint setting, with the p-values indicating the null hypothesis being rejected and favoring the alternative hypothesis of the two step approach having a lower $I_\epsilon(PF_1, PF_2)$ value, signifying it is the more dominating of the two approaches. Its average summary attainment surfaces paint a similar picture. At the higher spatial constraint setting, the null hypothesis is failed to be rejected due to the two having very similar quality indicator distributions. Indeed, examining their average summary attainment surfaces, it is not clear which one is better because both dominate in some portions of the Pareto front but not in another, with the cost and throughput differences in each portion not being very large. Despite this, the two step approach still requires fewer computational resources and so is preferred regardless over the full fidelity approach for the ATL process.

The two step approach performed noticeably worse for the SRI and CNC Machining processes, never actually outperforming the full fidelity approach at all unlike with the other processes and attaining lower throughputs in three of the four settings with the CNC Machining process and generally underperforming with the SRI process at the higher spatial constraint setting. The performance difference is lower in magnitude with the SRI process where, even with the rejection of the null hypothesis and favoring the full fidelity approach's results in the higher spatial constraint setting, the average summary attainment surfaces show that the two approaches can still achieve comparable results. The difference is present, but small. With the CNC Machining process, the performance of the two step approach as compared to the full fidelity approach is almost akin to how the full fidelity

approach compares to the truncated one when looking at their average summary attainment surfaces. This means that caution must be exercised when trying to use the two step approach to improve efficiency for that process.

In general, it appears that the two step approach works best when there is a large pool of possible assembly sequences, such as with the HLU/Autoclave, ATL, and VARTM processes. The two step approach helps significantly reduce the size of this pool of possible assembly sequences to only the most promising ones and actually allowing for an improvement in results as compared to the full fidelity approach. This is not quite the case with the SRI process because co-cured major parts in that process require that the minor parts also be co-cured, which actually substantially reduces the number of possible sequences. With a lower pool of possible assembly sequences that is still rather large, the two step approach performs roughly on par with, if overall just slightly worse than, the full fidelity approach. This is not the case with the CNC Machining process where the pool of assembly sequences is very small with only 5 major parts and no minor part integral fabrication being allowed. In this case, with such a relatively small number of assembly sequences, even accounting for major part subassemblies, the required larger number of generations without significant improvement before stopping the optimization allowed for more optimal line balances to be found. The two step approach, comparatively speaking, does not have as much time to find the exact optimal line balance. This is why its performance is so similar to the truncated full fidelity approach for the CNC Machining process; the available number of assembly sequences is small enough that the optimal ones could easily be found even without the two step approach, meaning the majority of the GA generations were spent optimizing for the line balance, which the truncated full fidelity approach and the two step approach did not have the ability to take advantage of.

Looking at the metrics themselves, the two step approach appears to perform best in the throughput area, with its results having a relatively large throughput difference in comparison with the full fidelity approach's solution's throughputs whenever the two step approach

is determined to be better. This indicates that the preliminary throughput, which was obtained by considering the ALBP's high throughput characteristics, performed very well in its role. The revised sequencing cost metric did not perform as well but, as indicated by the two step approach's lowest cost solutions always being very close in cost to the full fidelity approach's and performing much better than the truncated full fidelity approach's lowest cost solutions, it is not an undesirable one either. In fact, the low cost results in all the process' average summary attainment surfaces for the two step and full fidelity approach always being extremely similar most likely means that the revised sequencing cost is perfectly adequate, being able to almost exactly capture the lowest costs while requiring a lower number of generations to do so.

Examining the average summary attainment surfaces in the variable settings when the two step approach underperforms, it can be seen that the underperformance is not due solely either to either metric; for the SRI process, the underperformance is in the cost area, while for the CNC Machining process, it is in the throughput area, despite how well the preliminary throughput metric performed elsewhere. The inconsistency in which metrics underperformed across variable settings, as well as the pattern of manufacturing processes where the underperformance did occur, indicates that the cause is not due to the metrics but due to the available assembly sequence issue explained above. This means that the portion of Hypothesis 3 regarding the metrics themselves is supported and, even if not supported, is at least not rejected.

An interesting note to all this is that, because the two step approach already performs better than the full fidelity approach in some situations despite a faster stopping criteria, and only performs worse in others because of that faster stopping criteria, it is conceivable to flip the requirements around and have the two step approach have the same stopping criteria. In that case, it is almost guaranteed to yield the same results at worst and better results in most other situations for roughly the same computational cost.

Overall, because of the possible problems when running into scenarios where the pool

of assembly sequences is not very large, the two step approach cannot always be reliably used to improve efficiency without sacrificing the quality of the resulting solutions. Despite the fact that there is nothing wrong with the metrics themselves, this possible issue means that Hypothesis 3 is only partially accepted. Its usage is recommended only in some situations.

These situations cannot be certifiably verified due to the new, experimental, and still relatively un-tested nature of the two step approach and its representative metrics. Based on the results obtained above, the different situations and subsequent recommendations include the following. The superior quality of the two step approach and its representative metrics over the truncated full fidelity approach indicates that, if greater solution quality is desired with a slight decrease in efficiency, the two step approach should be used with the alteration that its stopping criteria is the same as the full fidelity approach. If a manufacturing process is used that results in the number of possible assembly sequences being very large, numbering in the thousands at minimum, and that pool size is unlikely to be reduced due to the effects of integral fabrication, then the two step approach is recommended. This situation matched the HLU/Autoclave, VARTM, and ATL processes tested. If the pool size includes a moderate number of sequences, numbering in the mid-hundreds, then the two step approach is recommended only if efficiency is desired at the possible cost of solution quality. If the pool size only includes a relatively small number of sequences, numbering in the low hundreds, then the two step approach is not recommended. This is due to the best sequences being much more apparent in that state, making the effects of slightly more optimized line balances that the two step approach is less likely able to attain much more impactful. The reasons for the smaller pool sizes can be varied. For the SRI process, it was due to the effects of integral fabrication, most notably major part integral fabrication combinations being more optimal and defaulting minor parts to also be integrally fabricated. This reduced its pool of more optimal assembly sequences from being very large to only being moderate in size. And for the CNC process, it was due to the complete lack of

integral fabrication as well as the low number of major parts modeled, making its pool very small. These recommendations are listed out in Table 6.4. Whenever the two step approach is less than fully recommended, the alternative option of using the full fidelity approach is always available to be used instead. The full fidelity approach should be used outright if the two step approach is not recommended.

Table 6.4: Two Step Approach Situations and Tentative Recommendations Based on Experiment 2 Results

Situation	**Two Step Approach Recommendation**	**Comments**
Higher Quality Results Desired, Slight Efficiency Decrease Acceptable	Recommended	Stopping Criteria Must Be Same as Full Fidelity Approach
Pool of Assembly Sequences is Very Large in Size	Recommended	
Pool of Assembly Sequences is Moderate in Size	Possibly Recommended	Efficiency Improved, But Quality of Results Not Guaranteed
Pool of Assembly Sequences is Small in Size	Not Recommended	Efficiency Improved, But Quality of Results Likely to Decrease

A more efficient approach to solving the integrated ASP and ALB problem was desired because the problem must be solved numerous times when incorporated into the aircraft design process. This is not a problem currently addressed in the literature because the literature does not consider incorporating assembly analyses with aircraft design in such a fashion. Algorithm-improvement techniques is ruled out due to constantly better ones always being developed and it is desired to keep the efficiency improvement as generalizable as possible in light of this so it can always be applicable. HPC is not used because it does not actually address the efficiency problem and tries to brute force it via computing speed instead. Surrogate models are leveraged for other purposes but are unable to address the efficiency requirements because they cannot replace the entirety of the ASP and ALB analyses by themselves. A two step approach was developed in response to all this where

the most promising assembly sequences and aircraft design configurations are first solved for in the first step before the second step focuses on solving the integrated ASP and ALB problem using only those most promising alternatives. This is done because previous works in the literature and previous trends from Experiment 1 indicated that the best assembly sequences from ASP are also able to generate the best line balances. Narrowing down the assembly sequences and aircraft design configurations limits the number of alternatives that must be examined, reducing the problem size and increasing the efficiency with which it is solved. To ensure optimality equivalence between the most promising assembly sequence and aircraft design alternatives with the best integrated problem solutions possible, in the first step the promising alternatives are selected by optimizing for metrics that consider the key characteristics of the ALB portion of the integrated ASP and ALB problem. Experiment 2 was carried out to test this hypothesis. This hypothesis was ultimately only partially accepted because while the metrics used were adequate and the problem-solving efficiency can be improved by a rough factor of 2, the two step approach was found to produce lower quality results whenever the problem size was already somewhat small to begin with.

This is all summarized in Figure 6.14.

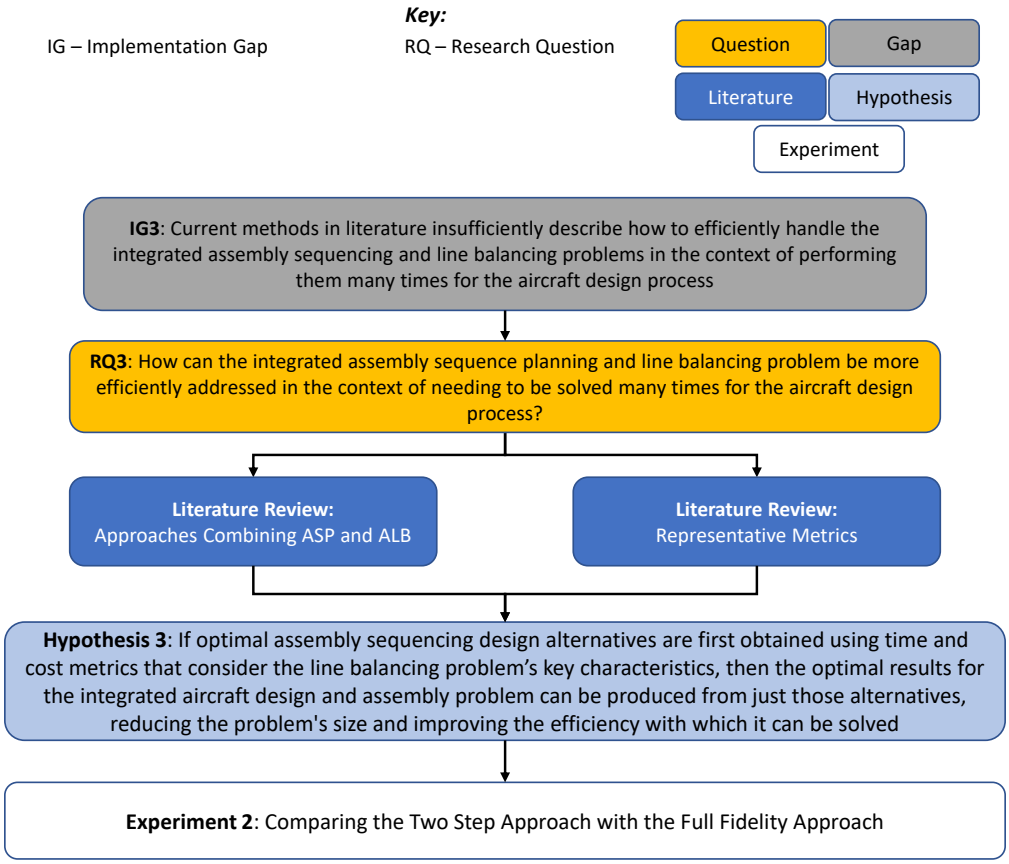


Figure 6.14: Summary Logic Diagram for Research Question 3

CHAPTER 7
AIRCRAFT CASE STUDY DEMONSTRATING INTEGRATED AIRCRAFT DESIGN AND ASSEMBLY FRAMEWORK

7.1 Integrated Framework for Aircraft Design and Assembly Tradeoffs

The aim of this work is to improve aircraft production rates and cost to meet future demand by designing the aircraft to incorporate those requirements from the start. This is done by better integrating the aircraft design and assembly disciplines. This is possible by using geometry and material-related information as the link between the disciplines' analyses to create stronger feedback loops between them. Aircraft geometry and materials are already regularly traded and changed in the aircraft design discipline. ASP and ALB are two assembly analyses also able to consider geometry and material information. The challenge has been to link the tradeoffs in one discipline with the equivalent tradeoffs in the other.

Chapter 4 focused on integrating early aircraft design and ASP using the geometric factors by enabling early aircraft design information to be passed on to and processed by ASP's geometric reasoning process. This was done by first identifying the appropriate aircraft design sizing fidelity such that the aircraft's overall configuration can be quickly sized while still being able to pass on sufficiently detailed geometric information to ASP for processing. Then, the appropriate ASP geometric reasoning fidelity was identified that allowed for as much geometric and mechanical feasibility information to be extracted as possible without hindering the required speed of the entire design process.

Chapter 5 focused on combining the ASP and ALB analyses such that they are able to incorporate all of the desired material information simultaneously in a multi-objective framework revolving around minimum cost and maximum production rate. The analyses have traditionally been able to do so but only with specific combinations of the material

information factors, not with all of them simultaneously and not while the two analyses are integrated together. ASP and ALB must be integrated together because the former depends on the latter for its precedence relations when they are both being considered and because ASP itself is helpful in being able to combine some of the material factors with the others. Additional capabilities related to the material factors that were found to be lacking in current ASP and ALB works were also developed and included, with the result being a framework able to solve the integrated ASP and ALB problem for minimum cost and maximum production rate.

Chapter 6 focused on finding ways to improve the efficiency of solving this integrated problem because it must be solved numerous times after it has been included with the aircraft design process. A two step approach was developed where the most promising aircraft designs and assembly sequences are first obtained using metrics also newly developed for this specific purpose. Those promising design alternatives are then used to seed the initial solutions for the integrated ASP and ALB problem. This is so that the line balancing efforts involved with solving the integrated problem are directed only at the most promising design alternatives as opposed to all available aircraft design configurations and assembly sequences. This reduces the size of the resulting integrated aircraft design and assembly problem, improving the efficiency with which it can be solved. However, this method can only be used if the size of the initial possible assembly sequences is large enough or else it could yield slightly less than optimal results.

With all of the setup work in place, everything can now be brought together. Aircraft design is first performed and the geometry and material system variables are varied to generate and size different designs. The geometry and material information is then brought over to the assembly discipline. The geometry information is used to determine what the available assembly sequences are and the accessibility of the resultant assemblies. The assembly sequences in turn define the precedence relations for the assembly line balancing, with the geometry having further impact in determining how long the tasks in the assembly

line balancing take to perform and how much they cost. The material system information comes into play here, with the different material systems defining the available manufacturing processes, the equipment that can be used and how much space they take up, and whether the parts can be integrally fabricated or not, all during the assembly line balancing. Line balances for the assembly sequences and geometry and material variables are obtained and the optimal ones determined, with variable settings that produce a large number of available assembly sequences using the two step approach to reduce the problem size and allowing the optimal line balances to be obtained more easily. From this, the Pareto front of the optimal production rates and costs as a function of the input aircraft design geometry and material variables is obtained. The throughputs and costs can be compared with their aircraft design output counterparts in the form of the aircraft's performance characteristics. From there, tradeoffs can be made about what can be done to maximize production rates at the appropriate cost level with the required aircraft performance characteristics and whether any requirements should be relaxed or altered.

This integrated framework is the culmination of this work's attempt to integrate the aircraft and assembly design disciplines such that more interdisciplinary feedback loops exist to have larger impacts on throughput and cost while still considering the aircraft's high level performance. This is all to better embody Schrage et al.'s [52] ideal IPPD approach where information can be freely moved between different disciplines and design phases and tradeoffs can be easily made if desired. Because of all these qualities, this final integrated framework is called the Integrated Framework for Aircraft Design and Assembly Tradeoffs, or INFRADAT for short. INFRADAT's flow is summarized in Figure 7.1, Figure 7.2, and Figure 7.3.

INFRADAT's workflow starts with step M1 in Figure 7.1 to select the manufacturing process used. This is done so that the size of the pool of possible assembly sequences can be evaluated in step M2 to see whether the two step approach should be used or not. This size is estimated based on the number of major parts being modeled along with the number

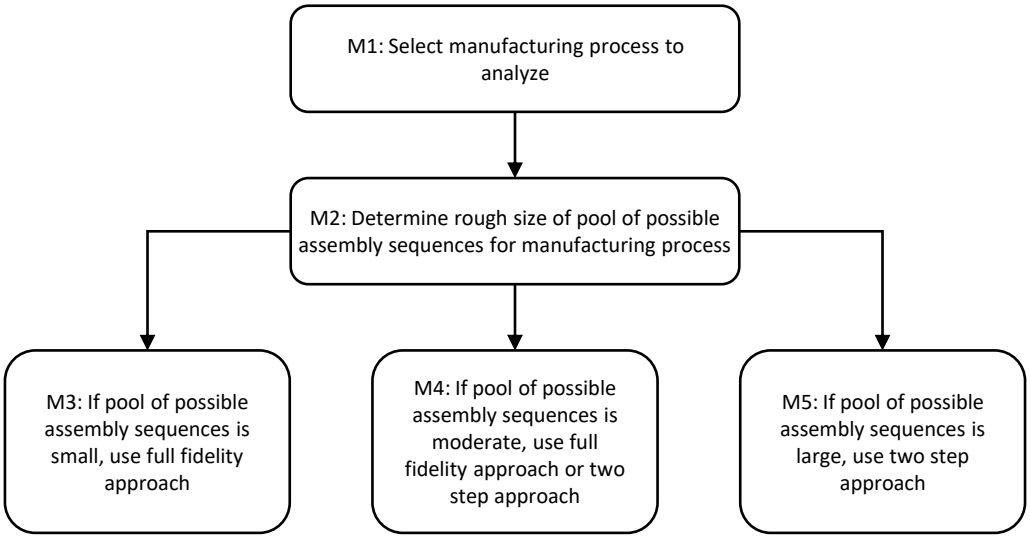


Figure 7.1: Flow of First Steps in INFRADAT

of parts that can be integrally fabricated for each particular manufacturing process. Assuming n major parts and m minor parts that can be integrally fabricated, there are at least $n! * 2^m$ possible assembly sequences as a first order approximation. Major part integral fabrication and subassemblies will significantly increase this value. "Small" in step M3 refers to possible sequences that number in the low hundreds, "moderate" in step M4 to sequences that number in the mid-hundreds, and "large" in step M5 to sequences that number in the thousands or higher. These recommendations were obtained from analyzing the results of Experiment 2 and summarized back in Table 6.4, where the two step approach was shown to be more efficient and have better quality results the larger the number of possible assembly sequences. Solution quality suffered when the number of possible sequences was too small.

The full fidelity approach is so named because it does not take any shortcuts in generating the solutions. Assuming that the full fidelity approach is used in Figure 7.2, all of the design variables are initiated in step F1. The aircraft is sized via analyses from the early preliminary aircraft design phases in step F2. This is described further at the beginning of subsection 5.5.2, in section 6.3, and in great detail in section 6.3, section A.1, and

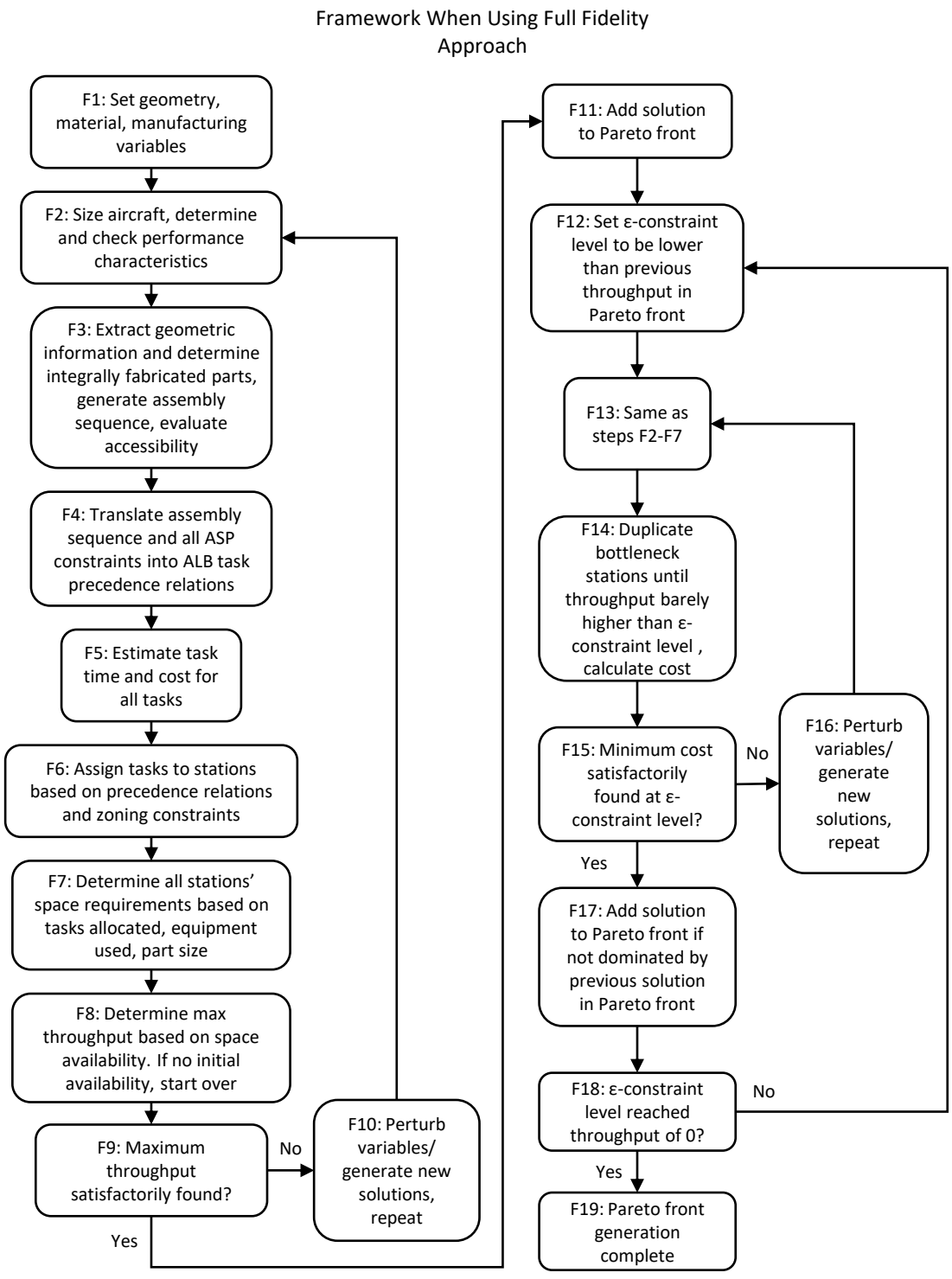


Figure 7.2: Flow of Full Fidelity Approach in INFRADAT

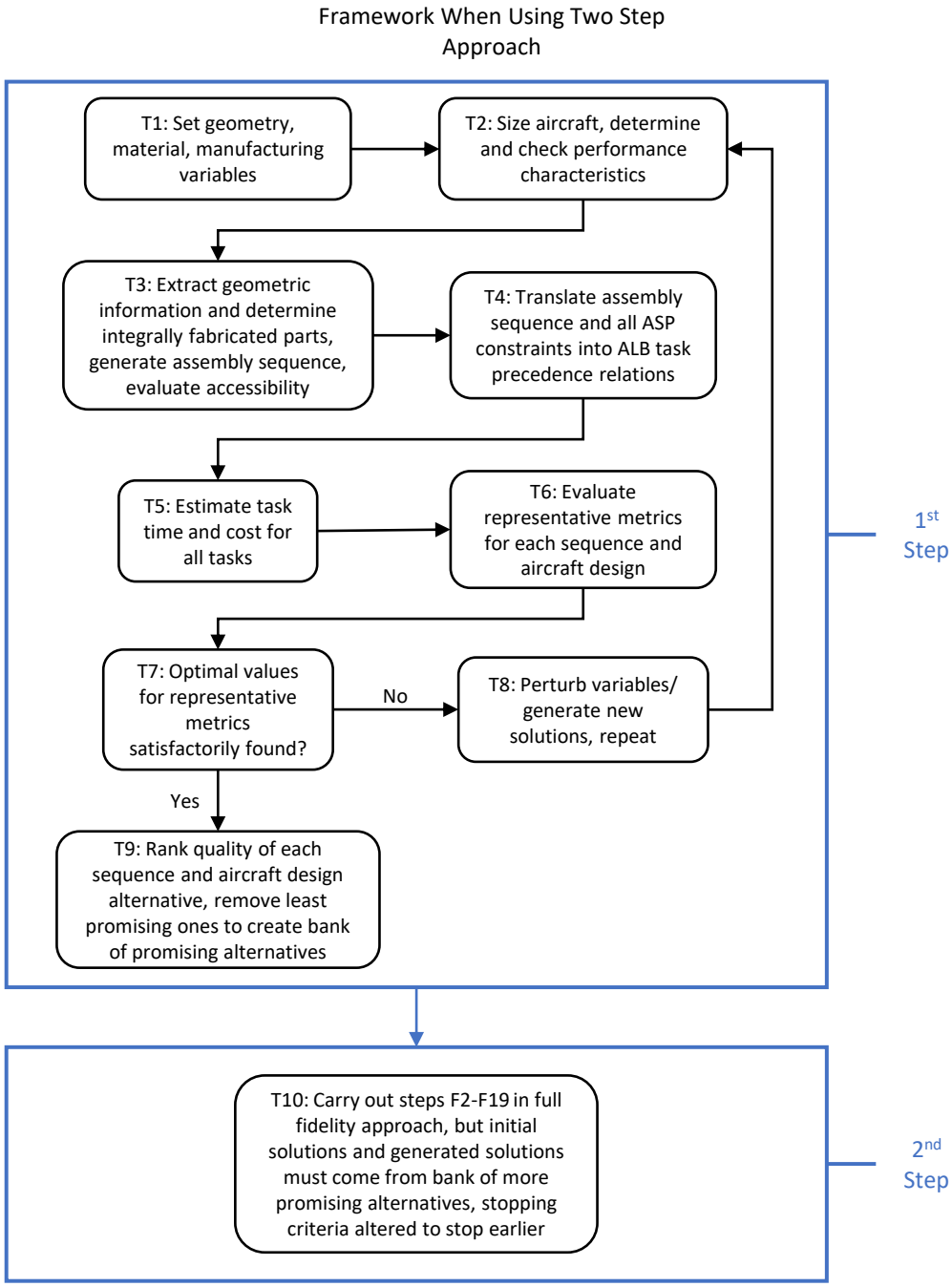


Figure 7.3: Flow of Two Step Approach in INFRADAT

section A.6 for the F-86 experimental platform. Essentially, surrogate models for various aircraft design and structural sizing programs such as FLOPS, NASTRAN, AVL, and Hypersizer are used to size the aircraft and evaluate its performance given the variables set in step F1. Aircraft designs are filtered out in step F2 as well if performance constraints are set.

In step F3, the detailed part and shape definitions from step F2 are extracted and used to perform the assembly sequence planning. The process is described in greater detail in subsection 5.4.1. To summarize, liaison graphs and interference matrices are created from the geometric definitions, with the interference matrices generated via the projection-based technique described in Figure 5.4. A combination of the liaison graphs and the If-Then precedence rules used to capture the manufacturing process's constraints are leveraged to select the parts to be integrally fabricated or part of a subassembly. The interference matrices are updated to account for the integral fabrication or subassembly usage if it is decided they will be used. Example If-Then rules for the F-86 experimental platform are in Appendix C. This information is afterwards all combined to create a feasible assembly sequence. A detailed description of a genetic algorithm implementation for the initial assembly sequence generation is provided at the beginning of subsection 5.4.3. The sequence's accessibility is then determined using a visibility-based method described towards the end of subsection 4.3.2 and subsection 5.4.1. The geometric reasoning process involving the interference matrix generation and accessibility calculations as they pertain to the F-86 experimental platform are also described in section A.2 to serve as a more definitive example.

For step F4, the assembly sequence generated in step F3 is combined with the If-Then precedence rules as part of the rule-based constraint modeling described towards the middle of subsection 5.2.1 in order to generate the precedence graphs and matrices needed for line balancing. Example If-Then rules for the F-86 experimental are again in Appendix C. Example precedence relations that make up the precedence graphs for the F-86 experimental

platform are shown in Figure A.5, Figure A.6, and Figure A.7. Example precedence matrix numberings for the F-86 experimental platform are shown in Figure A.8, Figure A.9, and Figure A.10.

In step F5, parametric time and cost estimation methods are used to calculate the task times and costs for all the tasks defined in step F4. INFRADAT in particular uses the SEER-MFG program to do this. In short, the requisite manufacturing information for every task is inputted into SEER-MFG along with the size of the part being operated on in said task. The labor times outputted by SEER are converted into task times based on the number of workers for each task type, described in Figure A.11 for the F-86 experimental platform. The tooling and material costs from SEER are used as-is, while the labor costs are a function of the task times and the line balance selected; SEER's labor cost calculation is not used. How this is performed for the F-86 experimental platform is explained in section A.4.

Step F6 is where the line balancing starts. It is described in the first half of subsection 5.4.2. The detailed genetic algorithm implementation for it is described towards the beginning of subsection 5.4.3. A list of available tasks is created by eliminating all tasks deemed unavailable based on the If-Then precedence rules, the assembly sequence, and the integral fabrication or subassembly part combinations used. Available tasks are randomly assigned to stations and done so feasibly by following the precedence relations imposed by the precedence graphs and matrices created in step F4 and also by following zoning constraints. Zoning constraints are described in more implementable detail at the end of section A.3 but basically mean that different fabrication task types use different equipment and tooling sets and so cannot be placed in the same station. On the contrary, assembly tasks are slightly more agnostic and so all assembly tasks can be placed in the same station, though they may need their own tooling sets. Whether or not different assembly tasks can share tooling sets is described towards the end of section A.5.

Once the tasks are assigned to their stations, how much space each station occupies

in the factory is calculated in step F7. The space taken up by each station is described by Equation 5.1 and includes space for the equipment or tooling, space for loading and unloading parts, aisle space, space for parts waiting in queue, and working space. The equipment, tooling, loading, unloading, and working space are defined by the manufacturing process, task type, and the largest part in the station. The queue space is a function of how many tasks are in the same station, and the aisle space is a function of the sum of all the other types of space. More detailed descriptions of how space is calculated for the F-86 experimental platform to serve as an example is in Appendix E and the latter half of section A.5.

With each station's space requirements calculated, the maximum possible throughput given the factory spatial constraint is calculated in step F8 by following the procedure shown in Figure 5.6. Bottleneck stations are continuously identified and duplicated, reducing their effective station time and increasing the amount of space used in the factory, until there is no space left in the factory. At that point, the cycle time is set to equal the bottleneck station's effective station time, shown in Equation 5.5 as the station time divided by the number of station duplicates or instances. The throughput is the inverse of the cycle time.

With the maximum production rate for a given design calculated, it must be determined in step F9 if that maximum production rate is the highest one possible among all the different designs. This is determined in INFRADAT's optimization process via the usage of its stopping criteria, explained in subsection 5.5.3 as a certain number of iterations without major improvement. If the calculated max throughput for a design is deemed to be insufficient, the design is perturbed or a new design generated in step F10. This is implemented in INFRADAT as a genetic algorithm, whose details are described in subsection 5.4.3 and the beginning of section 6.3. The primary variables changed during this process are the aircraft's design, the assembly sequence chosen, and the line balance used. After the stopping criteria has been reached, the highest throughput design forms the first solution in the

Pareto front in step F11.

After the initial solution on the Pareto front is obtained, the rest of the Pareto front is generated via the ε-constraint method described in the latter halves of subsection 5.2.3 and subsection 5.4.2 and in Figure 5.7. Essentially, ε-constraints of continuously decreasing throughput levels are set and the minimum cost solutions at each level optimized for. With the highest throughput design already supposedly identified, this sweeps through the rest of the design space. In step F12 the ε-constraint level is set to a throughput level lower than that of the previous solution added to the Pareto front. In step F13, steps F2 to F7 are repeated for a new design alternative at the current ε-constraint level. Step F14 is very similar to F8 except that, in F14, the station duplication of bottleneck stations only occurs until the throughput is barely higher than that of the ε-constraint level. The cost calculated afterwards for INFRADAT is the APUC, described in Equation 5.6. Similar to step F9, in step F15 it is determined if the APUC of the current solution is low enough to be the lowest possible of all designs at that ε-constraint level. If the optimization stopping criteria has not been reached yet with the current solution, step F16 is carried out and the design is perturbed or a new one created via a genetic algorithm, just like in step F10.

If the stopping criteria has been reached, the current lowest APUC solution is only added to the Pareto front in step F17 if it is not dominated by the previous solution on the Pareto front, in essence, the current lowest APUC solution has a lower APUC than the previously added solution on the Pareto front. Afterwards, if the ε-constraint throughput level is not 0 or sufficiently close to 0 in step F18, steps F12 to F17 are repeated until it is. This generates the Pareto front and visualizes the design space in step F19 by sweeping through all minimum cost solutions at every throughput level lower than the maximum one. All information associated with every solution on the Pareto front is stored. This includes not just the aircraft design, assembly sequence, and line balance used for every solution, but the number of station duplications for each station, all the task times, all the individual costs, and all the aircraft performance parameters as well.

The two step approach in Figure 7.3 is so named because it has two distinct steps. It is preferred over the full fidelity approach whenever appropriate due to having greater efficiency and therefore takes less time to generate similar or sometimes even better Pareto fronts than the full fidelity approach. It is described in more detail in section 6.2 and essentially improves efficiency by trying to identify more optimal aircraft designs and assembly sequences before the line balancing is done so that the computationally expensive line balancing is only performed on the best alternatives as opposed to every alternative.

Should it be selected as the approach to use in step M4 or M5 in Figure 7.1, its first several steps, T1 to T5, are identical to its full fidelity counterpart's, F1 to F5, respectively. Step T6 is where the two step approach becomes notably different from the full fidelity one. Instead of performing a formal line balance like in steps F6 to F8 where tasks are randomly assigned to stations, in step T6 preset line balances associated with the representative metrics are established instead. The two representative metrics used for INFRADAT's two step approach are the preliminary throughput and the revised sequencing cost metrics, described in more detail in the second half of subsection 6.1.3. They are calculated in the same way as the normal throughput and APUC metrics in the full fidelity approach except that their calculation can be done during ASP due to their usage of said preset line balances. The preliminary throughput metric assumes that every task is in its own station, after which the bottleneck ones are duplicated until there is no factory space left to maximize production rate. Conversely, the revised sequencing cost metric assumes that all tasks are consolidated into as few stations as possible given the precedence and zoning constraints to minimize equipment and tooling costs as well as to account for the cost of labor efficiency. In step T6, these two metrics are evaluated for the design alternative being analyzed.

Step T7 is the same as steps F9 and F18 in that it must be determined if the designs and their representative metrics have appropriately converged. If the stopping criteria has not been met, then step T8 is performed. Step T8 uses a genetic algorithm to perturb the design variables or generate new design alternatives entirely in the same way as steps F10 and F16

in the full fidelity approach. Steps T2 through T8 are iterated through until the stopping criteria for the representative metrics is met. At this point, in step T9, the quality of the representative metrics for all the design alternatives are evaluated, with the least promising alternatives for each metric eliminated. The remaining design alternatives constitute the bank of more promising designs. This concludes the first step of the two step approach, where more promising design alternatives are identified.

In the second step of the two step approach, step T10 in Figure 7.3, the same steps as the full fidelity approach's steps F2 to F19 are carried out, which constitutes performing the line balancing. However, the initial population of solutions along with all generated solutions in steps F2 through F19 must come from the bank of more promising designs. This is so that the line balancing is only done on the more promising alternatives. Additionally, the stopping criteria is altered such that the optimization converges earlier due to the assumption that more promising initial designs will cause the optimization to arrive at a converged answer sooner. For INFRADAT, this entails changing the stopping criteria for the genetic algorithm from 10 generations without significant improvement to 3 generations without significant improvement. The final output from step T10 is a Pareto front similar to or, ideally, better than the one generated for the full fidelity approach but that only needed a fraction of the time to compute.

This concludes the overview of the flow of INFRADAT.

7.2 Case Study Purpose

In the same way that the integrated ASP and ALB framework in chapter 5 had to be compared with the current state-of-the-art in the ASP and ALB literature to demonstrate its capabilities and show its improvements were worth its development in Experiment 1, so too must INFRADAT be compared with the state-of-the-art in the systems-thinking literature to showcase its capabilities improvement and demonstrate it accomplished what it set out to do and prove the Overarching Hypothesis in the Case Study, which is the overarching experiment. The purpose of the Case Study is thus to illustrate that using INFRADAT allows better throughput and cost tradeoffs to be made due to its superior leveraging of ASP and ALB to further integrate the aircraft design and assembly disciplines. The purpose is to emphasize INFRADAT is worth the trouble to develop.

To carry out the Case Study and its purpose, a state-of-the-art systems-thinking methodology, technique, or framework must be selected to compare INFRADAT against. Back in section 2.3 it was stated that systems-thinking methods incorporating aircraft design would be used as the baseline with which to develop what eventually turned into INFRADAT. As such, a systems-thinking method incorporating aircraft design will be used as the state-of-the-art comparison. The specific method to be compared against will be one of the MInD methodology's implementations since the MInD works are the most advanced ones available among the aircraft design-considering systems-thinking methods; no other works are able to size and examine aircraft designs and their requirements and performance characteristics at a high level and still be able to flow the needed information down to obtain a throughput and cost and are additionally able to do so for numerous different designs at the same time. However, MInD itself cannot be compared against since it is ultimately just a concept, similar to the IPPD approach, albeit one where aircraft are sized in the early preliminary design phase to allow many designs to be iterated through while providing enough information for parametric time and cost estimation to enable obtaining aircraft production

rates and cost. In that sense INFRADAT is technically also an implementation of MInD since it follows a very similar workflow, hence why the comparison must be with one of the MInD methodology's implementations and not the concept of the MInD methodology itself.

The most appropriate MInD methodology implementation to compare INFRADAT against is MInDPRO, which was created by Siedlak et al. in [62] and expanded on in their later work in [29]. MInDPRO follows the basic MInD methodology but also uses discrete event simulations (DES) to model the preliminary production environment and keep track of the factory's workflow and its resources, perform basic assembly line balancing, and optimize the final production rate and cost. It essentially uses DES to carry out what INFRADAT tries to achieve in its solving of the integrated ASP and ALB problem, with their overall similarity making MInDPRO the best framework to compare INFRADAT against.

To prove the Overarching Hypothesis, then, INFRADAT must show that it is able to obtain better throughput and cost tradeoffs than MInDPRO. As with Experiment 1, this is not quite so simple as just obtaining solutions with better production rate and cost values because those results need to be justifiable so the designer can be assured they are realistic. Additionally, exhibiting better justifiability brings with it better data traceability as well, which can be used to further improve on the results by making it easier to identify and therefore take advantage of notable trends. In particular, better tradeoffs imply that more tradeoffs can be made, most notably by including additional variables that were not present before but whose existence adds to the understanding of the problem, enables improved data traceability, and even possibly yield results with numerically superior throughput and cost. The meaning of "better tradeoffs" is thus diverse.

For the purposes of the Case Study, the framework able to make better tradeoffs must fulfill two conditions. The first is that the one with better tradeoffs must be able to obtain results with higher production rates and lower costs. This is because the primary motivation of this work is to cost-efficiently improve production rates and the aim is to still

be able to do so. However, the production rate and cost improvements must be done in a justifiable, traceable way that relates back to the variables and factors being examined, meaning the magnitude of the improvements is adjustable by setting those variables and factors to different values. Since MInDPRO and INFRADAT's variables and factors are all parameters that have significant real-life impact on manufacturing, being able to trace the results back to them and make tradeoffs with them means the results are grounded in reality. For INFRADAT in particular, because it was designed with ASP and ALB in mind to take advantage of additional interdisciplinary feedback loops based on geometry and material factors, if it is the framework with the better production rates and costs then the justification for those results must be traceable back to those factors. The first condition is thus related to the numerical superiority of the better framework and the designs it is able to produce.

The second condition is that the framework able to make better tradeoffs must be able to make a larger number of trades that have a noticeable impact on the production rate and cost. As the number of such trades is dependent on how many types of variables and important secondary parameters a framework is able to handle, it means the better framework has more types of variables and secondary parameters it can change to significantly influence cost and throughput. Due to MInDPRO and INFRADAT's variables and parameters being based on real, physical design aspects, more trades and thus more different types of variables and parameters means reality is better captured. Additionally, being able to change a larger number of such variables and parameters and observing the significance of their effects on the outputs results in additional insight as to how the problem behaves, leading to a greater understanding of it as well. This all leads to being able to optimize the throughputs and costs more effectively and understanding the limits of what can be physically done. The second condition is thus related to the better framework producing a greater understanding of the problem being solved by being able to make more tradeoffs with it. The framework able to make better tradeoffs must be able to meet both of these conditions,

meaning to prove the Overarching Hypothesis INFRADAT must be able to fulfill both of them.

7.3 Case Study Setup

7.3.1 MInDPRO Framework Overview and Alterations

As stated earlier, the MInDPRO framework is set up very similarly to INFRADAT and how it works was briefly described back in subsection 2.2.2. It first leverages the early preliminary aircraft design phase to size an aircraft or many aircraft able to meet the desired performance requirements. This is done using FLOPS for the aircraft sizing and the RADE framework for the structural optimization much in the same way as it is done with INFRADAT. The resulting geometric information is similarly used to provide inputs to SEER to parametrically estimate the labor times and costs. These time and cost estimates are afterwards inserted into a factory production flow model made using the DES software, Simio, to perform line balancing, track the flow of resources such as workers, parts, and sub-assemblies, and optimize the required number of lines, shifts, workers, and other manufacturing variables. The result is a production line optimized for either throughput or cost given the available design and manufacturing variables. Surrogate models of the aircraft sizing, structural optimization, and DES are then made. These are used in a GA, the NSGA-II to be exact, to obtain a Pareto front of optimal solutions and their production rates and costs.

The factory production flow model is now described in greater detail to enable comparisons with it later. In the production flow DES model, tasks in a station can only be carried out if there are sufficient resources to do so. These resources are the workers and the available tooling and they must both be available in a station for tasks in that station to be performed. This contrasts greatly with a typical ALB problem, where all stations are always fully manned with equipment and tooling always available, allowing for parts and subassemblies and subassemblies to be pulsed after an amount of time equal to the

cycle time. In a DES model, and thus in the production flow model, there is no pulsing and so there is no explicit cycle time, only an implicit one once the critical path of bottleneck stations has been identified. To affect this implicit cycle time and help determine the production rate, the number of station duplicates or instances is a design variable, as is the number of tooling sets, referred to as just toolsets from now on, and the number of workers. As all the stations are still fully equipped with the necessary equipment, the MInDPRO equipment costs are calculated the same way as the INFRADAT ones. The toolsets are assumed to be common tooling able to be used at every station, akin to the layup mandrel for composite processes in the F-86 experimental platform. However, as there is only a limited number of them, the tooling costs are not the same between MInDPRO and INFRADAT.

In MInDPRO, the total number of fabrication toolsets is based on how many toolsets there are for each fabrication line. The number of fabrication lines is dependent on how many unique stations, not station instances, are used to capture the first task in every fabrication task set. The total number of assembly toolsets is based on how many assembly stations there are, including station instances, since every assembly station requires its own toolset. The number of toolsets is multiplied by their respective tooling costs to obtain the total tooling cost for MInDPRO $C_{mindtool}$ and is represented as:

$$C_{mindtool} = \sum_{k=1}^{fn} n_i C_{mtool_i} \tag{7.1}$$

C_{mtool} is the cost of the common tooling for each fabrication line or the cost of the assembly tooling, n is the number of toolsets per fabrication line or the number of assembly station duplicates, and fn is the number of different fabrication lines and assembly stations. To round out the cost discussion, how labor costs are calculated is described later in this section while material costs are calculated in the same way as INFRADAT.

Further details on other aspects of MInDPRO can be found in Siedlak et al.'s works [62, 29].

To serve as a fair point of comparison with INFRADAT, several changes to the MInD-

PRO framework must be made as compared to how Siedlak et al. implemented it in their works. The first is regarding MInDPRO's ability to take in stochastic task times and output the possible production rates resulting from the task times being a distribution of possible times as opposed to a single, deterministic value. INFRADAT does not yet have this capability due to its primary focus being on integrating ASP and the other geometric factors with ALB in this initial iteration, whereas stochastic line balancing was determined to be an additional complexity that contributed more to reliability than pure production rate improvement and so was not included in INFRADAT's development. Consequently, to have as fair and direct of a comparison as possible, the task times used in MInDPRO for the Case Study will be deterministic as well. However, it is acknowledged that this is a trade-off that MInDPRO is currently able to make that INFRADAT is unable to. It will ideally be addressed and implemented in a future addition to INFRADAT.

On the topic of dynamic task times, another capability MInDPRO has is that it is able to reassign workers from one station to another if the station the workers are currently at is not a bottleneck station, altering a task's task time in the middle of a simulation when combined with the fact that task times can be altered based on the number of workers working on a task. This has the benefit of fewer workers possibly still being able to operate all stations at full capacity, decreasing labor cost, and also reducing the bottleneck station's station time, increasing throughput. However, this concept was only used for the hand layup tasks in Siedlak et al.'s work. In INFRADAT all tasks can have their task times reduced by having more workers, even automated tasks to a certain extent. It is not limited to just hand layup tasks. Additionally, all stations in INFRADAT have a minimum of 2 workers for safety reasons. To enable a fair comparison and not have INFRADAT be better by default due to being able to reduce task times on all other task types, MInDPRO's worker reassignment capabilities are altered.

Because MInDPRO's current worker reassignment capability and the fair comparison's requirements are so fundamentally contrasting, MInDPRO's worker reassignment capabil-

ities are actually changed entirely for an equivalent capability. Instead of allowing individual workers to move from station to station while a task is being carried out and always having all stations be manned, the change is made to have the stations and their instances not always needing to all be manned; each station and its instances will instead always have at least one set of workers among them, with the number of worker sets not having to equal the number of instances of a station and being adjustable. This is as opposed to INFRADAT, where every station and its instances are always fully manned. A worker set here refers to a group of workers whose quantity equals the number typically required for a task type, e.g., 4 workers for hand layup, 2 workers for ultrasonic NDI, etc. This change accomplishes several goals. First, it keeps the spirit of MInDPRO's resource tracking capabilities by allowing the resources, in this case worker sets, to still be allocatable to try to meet the production line's needs and reduce cost while maintaining or improving throughput. Second, it forces there to be a minimum of two workers since the minimum number of workers per worker set is 2. And third, this change allows the ability for workers being able to reduce task times to be applicable for all task types and additionally allows for different numbers of workers for each task type's worker sets. These changes are thus made because they are equivalent to MInDPRO's previous capabilities while augmenting it to allow for a fairer comparison with INFRADAT.

As a result of these changes, however, MInDPRO's labor variables and total labor costs are slightly altered from what they were in Siedlak et al.'s works. In MInDPRO, the number of initial workers per station is a variable set before the DES is performed. With worker sets replacing the number of individual workers per station, different variables are needed. There is now a "number of worker sets" variable for every fabrication task type and a number of worker sets variable for the assembly tasks. What this means is that every station of a particular fabrication task type has the specified number of worker sets unless there are more worker sets than there are instances of that station, in which case the number of worker sets is limited to be equal to the number of station instances. Each worker set has

a number of workers equal to the typical number required for that task type. This is all done to limit the number of variables while still making worker sets distinct between different task types; it is impractical to have a number of worker sets variable for every station. As an example, assume a situation where the number of hand layup worker sets is set to 2 and manual trim worker sets is set to 3, there are 5 hand layup stations, one for each major part, and 3 manual trim stations, and 3 instances of each hand layup station and 2 instances of each trim station allowed. Each hand layup worker set has 4 workers and each manual trim worker set has 3 workers. For the total number of hand layup workers, this is 4 workers per worker set, 2 worker sets per hand layup station, and 5 hand layup stations, and results in a total of 40 hand layup workers, a value which can be used to calculate the labor cost given wage rate and how long those workers work for. For the total number of manual trim workers, this is 3 workers per worker set, 2 worker sets per trim station since there are only 2 instances of each trim station allowed, and 3 trim stations, resulting in 18 manual trim workers.

This causes the MInDPRO total labor cost $C_{mindlabor}$ to take the following form:

$$C_{mindlabor} = \frac{N}{T} * CF * \sum_{k=1}^{fa} min(n_k, ws_k) m_k z_k \omega_k \tag{7.2}$$

N is the number of units in the production run, T is the production rate in units per month, CF is the conversion factor of how many working hours there are per month, fa is the total number of different fabrication and assembly task types, n is the number of instances allowed for stations of that task type, ws is the number of worker sets for stations of that task type, m is the total number of different stations of that task type not including station duplications/instances, z is the number of workers for the particular task type, and ω_k is the wage rate in dollars per hour. For CF, the conversion is that there are 8 hours per shift, 2 shifts per day, 5 days a week, and 4.33 weeks a month, similar to INFRADAT. Also similar to INFRADAT, workers are paid for being at work even if there is idle time.

Another major change to MInDPRO is related to how the design variables are sepa-

rated out. The MInDPRO design variables in Siedlak et al.'s works were set based on the foreknowledge of some of the problem's behavior, such as that the NDI stations in their works were the bottleneck stations or that rib and spar stations can be duplicated together for maximum efficiency. For the Case Study, such foreknowledge is not present and so some of the design variables are made more generic or split up to become more generic. Additionally, the number of shifts was removed as a design variable due to the desire to keep the number of shifts constant between INFRADAT and MInDPRO for a more fair comparison. As indicated earlier, the number of shifts per day is 2 for both. More details on this are shown when the design variables are discussed later.

The last major alteration to MInDPRO is not so much a change as much as it is an omission. MInDPRO has many additional capabilities beyond just optimizing production rates and cost such as predicting cumulative cash flow, determining ROI, and estimating the backlog size given demand variability. These are capabilities that INFRADAT currently lacks and that ideally could be infused into INFRADAT in the future. However, for the purposes of proving the Overarching Hypothesis, the only capabilities of interest are those related to determining the designs yielding the optimal production rates and cost. As a result, only those capabilities will be considered for MInDPRO when comparing it and INFRADAT. The other capabilities are not relevant and so are omitted.

7.3.2 F-86 Use Case and Commercial Aircraft Equivalence

To carry out the Case Study, a use case is needed to perform the Case Study on in the same way an experimental platform was needed to perform the experiments on. The motivation for this work, which focused on improving commercial aircraft production rates cost-effectively due to the large amount of future demand, means that a use case focused on an entire commercial aircraft is preferable. Despite this, the use case chosen is the same F-86 experimental platform as was used for Experiments 1 and 2 but with all of its aircraft design considerations, variables, and constraints included. This is for a variety of reasons.

In terms of practicality, this is because an extraordinarily large database of information with the appropriate level of detail is needed for any aircraft and its assembly or assemblies to be used as the use case. On the aircraft design side, this includes the ability to estimate its aerodynamic characteristics and access to the aircraft's design mission parameters to allow it to be properly sized as well as detailed knowledge of the assembly's internal geometry to allow it to be structurally optimized. Though a tall order, it is still possible to do this with an assembly such as NASA's Common Research Model, which is a wingbox for a Boeing 777-sized aircraft. However, the data requirements for the manufacturing side makes usage of any commercial aircraft use case impractical because data on the detailed manufacturing processes and all of their constituent tasks and the equipment and tooling used is almost always proprietary in nature and difficult to obtain. In comparison to this, data on the F-86 wingbox is relatively abundant with flight manuals [253], design and structural testing handbooks [254], structural repair handbooks [255], maintenance handbooks [256], manufacturing process details [257], and information on its design mission and performance specifications [225] abundantly available. This is all on top of the information gathered in the course of the development of the MInD works, which also used some of these sources. This plethora of easily obtainable information makes using the F-86 wingbox as the use case significantly more plausible than trying to do the same with a full commercial aircraft.

However, plausibility does not mean very much if the tradeoffs and capabilities demonstrated on the F-86 wingbox are not applicable to a full commercial aircraft, which is the desired target. In this sense, the F-86 wingbox is indeed not the same as a commercial aircraft, with it being part of a fighter jet instead of a commercial one and it being a wingbox as opposed to an entire aircraft. But though it is not the same as a commercial aircraft, it can be argued that the tradeoffs and capabilities demonstrated using it are transferable to a commercial aircraft, should the data for it be provided, and that it is equally appropriate to be the Case Study's use case. This means that performing tradeoffs on the F-86 wingbox may not be the same as on a commercial aircraft, but it is equivalent. This is due to several

reasons.

First, one of the main desires from examining the entire aircraft is to capture the interactions between its design and aircraft performance with its manufacturing performance, in essence to relate the aircraft's performance to its manufacturability. This is due to the purpose of this work being to take advantage of altering the aircraft's design and performance to try to improve its manufacturability. In this, the wings and its wingboxes are a microcosm of the entire aircraft and so any tradeoff capabilities performed and demonstrated on them are representative and transferable to the entire aircraft. This is from the fact that, on the design side, sizing the wing and its wingbox are the most important aspects of early aircraft design. In conceptual design, the primary parameters being solved for include the wing's aspect ratio, sweep, span, surface area, wing loading, and many other wing-related variables. They are all critical to determining the performance of the aircraft as a whole and setting its weight, which ultimately fixes the aircraft's size. On the manufacturing side, many of these variables affect the overall size of the wing, which determines its internal structural arrangement and so how much material and effort it needs in order to be fabricated and assembled. Additionally, its internal geometry affects its structural and, therefore, aircraft performance as well as having significant effects on assembly via accessibility. The wing and wingbox thus have parameters and variables that affect both its aircraft and manufacturing performance and allow the two to interact in a manner similar to parameters related to the entire aircraft. Examining the wing and wingbox allows for capturing of the same tradeoffs and interactions as examining the entire aircraft. This makes the wing and wingbox as suitable of a candidate for the use case as the entire aircraft since they can both be used to prove this work's main point–to trade the aircraft's size and performance for the ability to make it more quickly and seeing where the equilibrium point should be set.

Second, on the topic of aircraft design, it is more difficult to design and size fighter jets than it is to do the same for commercial jets. This is because fighter jets have to perform

more complex maneuvers and bear significantly higher loads due to the requirements of being able to combat effectively. As a result, fighter jet designing and sizing is much more restrictive than it is for commercial jets. Thus, displaying the ability to size a fighter jet like the F-86 is indicative of being able to do the same for a commercial jet because it is harder to do the former than the latter. This means that, for aircraft design, displaying the ability to design the F-86 signifies that the ability to design a commercial aircraft is present as well.

Third, the internal geometries of aircraft wingboxes in general are very similar. The F-86's wingbox uses a semi-monocoque structure in the same manner as commercial jet wingboxes. The F-86's wingbox is composed of the same type of components as a commercial jet's as well, with both consisting of spars, wingskins, ribs, stringers, and stiffeners, and all arranged in roughly the same way. The efficiency of this type of structure generally makes wingbox geometries of larger aircraft all look incredibly similar, which also means that they can be structurally sized and manufactured in a very similar fashion as well. Due to this, exhibiting the ability to perform analysis and tradeoffs on the F-86 wingbox means that the same can be done on a commercial jet wingbox due to the inherent similarity of all wingboxes. The capabilities shown for the F-86 wingbox are thus transferable to a commercial one as well, with the main difference between them being that one just needs to be scaled up more in size to match the other.

And lastly, in regard to assembly specifically since the Overarching Hypothesis focuses on assembly, the wingbox is a significant portion of total assembly costs and is a microcosm of the entire airplane as well in the assembly sense, possessing the same characteristics as the whole airplane and making tradeoff capabilities transferable between the two. Assembly as it is appropriate for the whole airplane is carried out in the FAL process, where the wings and the fuselage are joined. Its importance is emphasized by the majority of non-aircraft design systems-thinking methods only examining the FAL when analyzing aircraft. However, it is very difficult to see the effects of changing the configuration and the aircraft's

performance attributes at this level. Major decisions such as integral fabrication cannot be considered at all at this level since the wings and fuselage are already fully fabricated and set. This is in contrast to the fabrication and assembly of the wing and wingbox, where decisions like that can still be made. And indeed, the wingbox assembly itself makes up a large portion of all assembly performed on the aircraft and is comparable to the FAL process. As an example, Markish has a table of desensitized recurring cost data for a 777-200, shown in Table 7.1 [258].

Table 7.1: Desensitized Normalized-By-Weight 777-200 Recurring Cost Data in $/lbs., Adapted from Markish [258]

	Labor	Materials	Other	Total
Wing	$609	$204	$88	$900
Empennage	$1614	$484	$223	$2331
Fuselage	$679	$190	$98	$967
Installed Engines	$248	$91	$36	$374
Systems	$315	$91	$46	$452
Final Assembly	$58	$4	$3	$65

Using public data and Raymer's weight buildup estimation relations, the wing is, approximately, 15% of the 777-200's empty weight [30]. Labor costs tend to be calculated as a labor rate times the labor time. Assuming that assembly takes up 30% of the total labor time for the wing manufacturing; that labor rates for the FAL and the individual part assemblies are similar; and that the weight estimates are reasonable, the assembly labor time for the wing is roughly half that for the FAL. This is not accounting for the fabrication processes that are dependent on the assembly operation type and thus should reasonably be included as well. The wingbox assembly is of the same magnitude of importance as the FAL process and so looking at it is representative of looking at assembly for the entire aircraft; the FAL is simply just a scaled-up version of the wingbox assembly, whereas the FAL cannot be scaled down to represent the wingbox manufacture since it does not include any fabrication. On top of this, using the just stated estimates and Table 7.1, it can be seen

that the manufacture of the wingbox is likely one of the, if not the primary, bottlenecks in determining the entire aircraft's production rate. This is the reason why most designers when creating a design and focusing on a specific production rate tend to look at the manufacture of the wing as one of the most important limiters. Thus, not only is the wingbox's assembly representative of the whole airplane's FAL process and likely more advantageous to perform tradeoffs on to boot due to being able to make design tradeoffs at all, but it is likely that the wingbox's manufacture is the primary determinant of aircraft production rates in general.

Due to all these reasons, the F-86 wing and wingbox are just as suitable a candidate for the use case as an entire commercial jet, and any tradeoff capabilities performed and demonstrated on the former is applicable and transferable to the latter due to their equivalence. It means that the tradeoffs performed for the F-86 wingbox can be performed on a commercial jet in the same way should enough data to adequately model the latter exist and be made available. For these reasons and due to the availability of its data, the F-86 will be used as the use case in the Case Study.

The F-86 use case is very similar to the F-86 experimental platform used in Experiment 2, with the main difference being that the full range of spar shapes and spatial constraints are used instead of the limited, most extreme ones considered in Experiment 2. The spar shapes are listed out in section A.6 and will be referenced again when setting the design variables and their ranges later on while the full list of spatial constraints was described back in Experiment 1.

7.3.3 Setup and Implementation Details, Design Variables, Procedure

To determine which framework is better, the two conditions explained in section 7.2 must be fulfilled by either framework. As one of the conditions is to obtain solutions and designs with better throughputs and costs, i.e., generate higher quality Pareto fronts, the same quality indicators from Experiment 1 and Experiment 2 are used. This takes the form of

the $I_\epsilon(PF_1, PF_2)$ and HVR indicators. The procedure from Experiment 2 for determining the reference point is used when calculating the HVR indicator where the reference point APUC is set to be 1% higher than the highest cost solution of both frameworks' solutions for each variable combination. The reference point throughput is 0. Summary attainment surfaces are again leveraged to visualize the average and best Pareto fronts attained by both frameworks. The number of generations required to attain those fronts is used to see how computationally expensive each framework with, though it is not as important a metric as it was in Experiment 2. The Wilcoxon rank-sum test from Experiment 2 will be used to statistically verify the superiority of either framework if the result is not obvious from their $I_\epsilon(PF_1, PF_2)$ distributions and summary attainment surfaces. In the same way as with the other experiments, there is no "ground truth" or historical data to certifiably ascertain what the true Pareto front looks like. As such, the true Pareto front is instead created from all the non-dominated designs and solutions from both frameworks.

The same 5 manufacturing processes from Experiment 2 are used again for the Case Study. For INFRADAT, based on Experiment 2's results, the two step approach will be used for the HLU/Autoclave, VARTM, and ATL processes while the non-two step approach will be used when analyzing the SRI and CNC Machining processes. For the two step approach, the stopping criteria is kept the same as in Experiment 2, with the two step approach halting a sub-problem when 3 generations pass without a significant improvement and the non-two step approach halting a sub-problem when 10 generations pass without a significant improvement. The INFRADAT implementation is otherwise the same as in Experiment 2.

MInDPRO uses surrogate models in the same way INFRADAT does to perform its optimization. Because the aircraft design portion of MInDPRO is essentially the same as INFRADAT's, INFRADAT's surrogate models for the FLOPS program and RADE framework are also used by MInDPRO. In a similar manner, the parametric cost and time estimation tool used by both is the same in the form of SEER, and so INFRADAT's SEER surrogate models are also used by MInDPRO. MInDPRO, however, needs an additional set

of surrogate models in the form of surrogate models for its factory production flow model. This factory production flow model must thus be implemented in Simio and the results used to create the surrogate models.

Before that can be done, however, the production line for each manufacturing process must be defined. MInDPRO requires that the designer define which tasks are assigned to which stations before the model can even be created since each server in the model is a workstation, essentially requiring a preliminary line balance before the model can even be run. The line balancing the model does when it is run is merely to determine what the optimal combination of number of workers/worker sets and number of station instances should be. And since a line balance requires an assembly sequence, each process in MInDPRO requires an assembly sequence and line balance assigned to it before its DES can be performed. The assembly sequences are chosen to be as representative of the integral fabrication capabilities for each process as possible while still allowing the line balances to be the same as they are in Siedlak et al.'s works [62, 29]. The assembly line balances are subsequently chosen to be as representative of the ones used in Siedlak et al.'s works as possible as well. The assembly sequences for each process are presented in Figure 7.4 and the line balances used in Figure 7.5. The meaning of the task numbers for each process in the line balances is explained in section A.3. For the spar shapes, which determine the assembly sequence feasibility and accessibility, Siedlak et al.'s MInDPRO implementations used an Out-C spar shape and so that is used for all the MInDPRO designs.

To help clarify terms for later, in MInDPRO's factory production flow model for every manufacturing process there are three distinct subassembly lines: one for the spars and its minor parts, one for the ribs and its minor parts, and one for the wingskins and its minor parts. Each subassembly line is split up further into fabrication lines, defined earlier as every unique station used to capture the first task in every fabrication task set. As an example, the ATL process has 6 distinct fabrication lines based on its line balance: one for the lower and upper wingskins and one for the upper and lower stringers, which belong

CNC Machining, HLU/Autoclave

1	5	2	4	3
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

ATL, VARTM

1	5	2	4	3
0	0	0	0	0
0	0	0	0	0
1	1	1	1	1

SRI

1	2	4	5	3
0	0	0	0	0
1	1	1	0	0
1	1	1	1	1

Figure 7.4: Assembly Sequences Used for Each Manufacturing Process for MInDPRO

to the wingskin subassembly line, 1 for the front and rear spars and 1 for the front and rear spar stiffeners, which belong to the spar subassembly line, and 1 for the ribs and 1 for the rib stiffeners and shear ties, which belong to the rib subassembly line. Thus, spar fabrication lines refer to the front and rear spar fabrication line and the front and rear spar stiffener fabrication line for the ATL process. These concepts are used to assign toolsets, with each fabrication line being assigned a certain number of them.

Based on each process's line balance in Figure 7.5, each process has a different number of fabrication lines that each need their own toolsets, which determines the total number of toolsets needing to be procured and therefore sets the tooling cost. The list of fabrication lines for each process's subassembly lines is shown in Table 7.2. A notable difference is that in MInDPRO, front spar and rear spar stiffeners and upper and lower stringers are made separately as opposed to being in the same batch for the VARTM and SRI processes, which is done to mimic Siedlak et al.'s works.

There are a few other differences between how MInDPRO implements its factory and how INFRADAT implements its line balance for its own factory. MInDPRO has discrete

VARTM

1	5	9	13	17	21	25	29	33	37	40	43	46	49	18	22	26	30	34	19	23
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21

27	31	35	20	24	28	32	36	52	53	54	55	56	57	58	59	60	61	62	63
22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41

ATL

1	6	11	16	21	26	31	36	41	2	7	12	17	22	27	42	3	13	18	8	4
1	2	3	3	4	4	5	5	6	7	8	9	9	10	10	11	12	13	13	14	15

9	14	19	5	10	15	20	46	49	52	55	58	23	28	43	33	38	24	29	34	39
16	17	17	18	19	20	20	21	21	22	22	23	24	24	25	26	26	27	28	29	29

44	25	30	35	40	45	61	62	63	64	65	66	67	68	69	70	71	72
30	31	32	33	33	34	35	36	37	38	39	40	41	42	43	44	45	46

HLU/Autoclave

1	5	9	13	17	21	25	29	33	2	10	14	18	22	34	6	26	30	3	7	11
1	2	3	4	5	6	7	7	8	9	10	11	12	13	14	15	16	16	17	18	19

15	19	23	27	31	35	4	8	12	16	20	24	28	32	36	37	38	39	40	41	42
20	21	22	23	23	24	25	25	25	25	25	25	25	25	25	26	27	28	29	30	31

43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52

SRI

1	6	11	16	21	26	31	36	41	46	49	52	55	58	2	7	17	73	74	61	37
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21

33	38	43	34	39	44	35	40	45	64	65	66	67	68	69
22	23	24	25	26	27	28	29	30	31	32	33	34	35	36

CNC Machining

1	2	3	15	22	16	23	17	24	7	11	18	25	29	8	12	30	4	19	26	9
1	2	3	4	5	6	6	7	8	9	10	11	12	13	14	14	14	15	15	15	16

13	31	5	20	27	6	10	14	21	28	32	33	34	35	36	37	38	39	40	41	42
16	16	17	17	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33

43	44	45	46	47	48	49	50
34	35	36	37	38	39	40	41

Figure 7.5: Line Balances Used for Each Manufacturing Process for MInDPRO, First Row is Task Number, Second Row is Station Number

Table 7.2: List of Fabrication Lines in Each Subassembly Line for Each Manufacturing Process

Manufacturing Process	**Rib Subassembly Fab Lines**	**Spar Subassembly Fab Lines**	**Skin Subassembly Fab Lines**	**Total Fab Lines**
ATL	Ribs, Rib Minor Parts	Front & Rear Spars, Front & Rear Spar Stiffeners	Upper & Lower Skins, Upper & Lower Stringers	6
CNC Machining	Ribs	Front Spar, Rear Spar	Upper Skin, Lower Skin, Upper & Lower Stringers	6
HLU Autoclave	Ribs, Shear Ties, Rib Stiffeners	Front Spar, Rear Spar, Front & Rear Spar Stiffeners	Upper & Lower Skins, Upper & Lower Stringers	8
SRI	Ribs, Shear Ties, Rib Stiffeners	Front Spar, Rear Spar, Front Spar Stiffeners, Rear Spar Stiffeners	Upper Skin, Lower Skin, Upper Stringers, Lower Stringers	11
VARTM	Ribs, Shear Ties, Rib Stiffeners	Front Spar, Rear Spar, Front Spar Stiffeners, Rear Spar Stiffeners	Upper Skin, Lower Skin, Upper Stringers, Lower Stringers	11

and distinct tool cleaning and tool prep workstations to prepare and clean its tooling, whereas in INFRADAT the tool cleaning and preparation is done as a part of each task. This causes their task times to differ slightly. There is a tool clean and a tool prep station for every subassembly line, with the stations and all their instances being manned by two workers each. The number of instances of the tool clean and prep stations for each manufacturing process and subassembly line is based on the number of fabrication lines in that subassembly line as well as the relative length of the tasks in those fabrication lines. The number of instances for each subassembly line is shown in Table 7.3. The station times for the tool clean and tool prep stations in each subassembly line is equal to the total tool clean and total tool prep times belonging to the longest-duration fabrication line in that subassembly line. The tool clean and tool prep stations do not use any equipment or tooling. They each take up the same amount of space as their subassembly line's largest water jet trim or manual trim stations for the composite processes or their subassembly line's largest chromate spray station for the CNC Machining process. This is because cleaning a tooling set is similar to accommodating for a part being trimmed or sprayed and so it is assumed it takes up the same amount of space.

Table 7.3: Number of Tool Clean and Tool Prep Station Instances for Each Manufacturing Process's Subassembly Line

Manufacturing Process	**Rib Subassembly Line**	**Spar Subassembly Line**	**Skin Subassembly Line**
SRI & VARTM	3	4	4
CNC Machining	2	4	6
HLU/Autoclave	3	3	4
ATL	4	4	4

Another notable difference between MInDPRO and INFRADAT is that MInDPRO explicitly separates the bagging and debagging activities in the curing task, putting them each in separate stations from the curing activity in the curing task, while INFRADAT only does so implicitly. In addition to this, parts can be cured together despite not being bagged to-

gether to reduce the number of stations with curing activities and, therefore, the number of ovens and autoclaves. The list of which parts are cured together in the same oven or autoclave is presented in Table 7.4. The number of instances of cure stations is equal to half the number of instances of bagging and debagging stations, rounded up. This is essentially the DES version of INFRADAT duplicating the curing and bagging/debagging activities separately.

Table 7.4: Parts Cured Together in MInDPRO Factory Production Flow Model

Manufacturing Process	**Parts Cured Together**		
ATL	Rib Minor Parts, Spar Minor Parts	Spar Major Parts, Rib Major Parts	Skin Major and Minor Parts
HLU/Autoclave	Rib Major and Minor Parts, Spar Major and Minor Parts	Skin Major and Minor Parts	
VARTM	Rib Major and Minor Parts	Spar Major and Minor Parts	Skin Major and Minor Parts
SRI	Rib Major and Minor Parts	Spar Major and Minor Parts, Lower Skin and Stringers	Upper Skin and Stringers

With the factory production flow model fully described to account for the different manufacturing processes, ANNs are used to create the surrogate models for their implementation Simio in the same way the ANNs were used to create surrogate models for FLOPS, RADE, and SEER. How the factory production flow model was run to create the outputs was that the model simulates the production run for a 6 month period after the inputs are entered. During this time, how many products go through each station is recorded, with the simulation's outputs being the total number of products that left each station after the 6 month period. The number of products that leave each station are used later on to determine the station utilization for those stations. The number of products that leave the last station, which has the final major part fasten task assigned to it, represents the production rate for the input settings used. Though the task times are deterministic and the production

rate variability is low as a result, there is still some degree of stochasticity to DES models and so the simulation is run 3 times for every input variable setting. The average values of the three runs are used as the final outputs for the input settings used. The input variables and their ranges used for the surrogate models are shown in Table 7.5. These make up the majority of the MInDPRO design variables for the Case Study, which are covered later on.

Table 7.5: Variables and Ranges for Design of Experiments for Surrogate Models for Simio Factory Production Flow Model

Variable	**Range**
Wing Area (ft^2)	240-410
Aspect Ratio	3.84-6.24
Taper Ratio	0.36-0.66
Range (nmi)	600-1000
Number of Spar Fab Line Toolsets	2-5
Number of Skin Fab Line Toolsets	2-5
Number of Rib Fab Line Toolsets	2-5
Number of Assembly Station Instances	1-4
Number of Fab Task Type 1 Station Instances	1-4
Number of Fab Task Type 2 Station Instances	1-4
Number of Fab Task Type 3 Station Instances	1-4
Number of Fab Task Type 4 Station Instances	1-4
Number of Fab Task Type 5 Station Instances	1-4
Number of Assembly Station Worker Sets	1-4
Number of Fab Task Type 1 Worker Sets	1-4
Number of Fab Task Type 2 Worker Sets	1-4
Number of Fab Task Type 3 Worker Sets	1-4
Number of Fab Task Type 4 Worker Sets	1-4
Number of Fab Task Type 5 Worker Sets	1-4

The fabrication task type numbers are a generic way to represent the number of station instances and worker sets for stations of each fabrication task type across all five manufacturing processes. As an example, fabrication task type 1 for the ATL process is the automated tape layup task and for the HLU/Autoclave process is the hand layup task. And if the number of fabrication task type 1 station instances is set to 3 and worker sets set to 2 for the ATL process, it means that all automated tape layup stations will have 3 instances

and each of those stations will have 2 worker sets to man their 3 instances. Figure 7.6 shows how the fabrication task type numbers translate to the actual tasks for each manufacturing process.

Manufacturing Process	Fab Task Type 1	Fab Task Type 2	Fab Task Type 3	Fab Task Type 4	Fab Task Type 5
HLU/Autoclave	Hand Layup		Autoclave Cure, Bag/Debag	Manual Trim	Ultrasonic NDI
ATL	Automated Tape Layup	Hot Drape Form	Autoclave Cure, Bag/Debag	Water Jet Trim	Ultrasonic NDI
VARTM	Hand Layup		Oven Cure, Bag/Debag	Water Jet Trim	Ultrasonic NDI
SRI	Minor & Major Assembly	Stitch	Oven Cure, Bag/Debag	Water Jet Trim	Ultrasonic NDI
CNC Machined	Roll Form, Stretch Form, Skin Pre/After Form, Stringer Pre/After Form	CNC Mill	Dye Penetrant	Anodize	Chromate Spray

Figure 7.6: How Fabrication Task Type Numbers Translate to Each Manufacturing Process's Task Types

The number of assembly toolsets is not listed as a variable because, based on Siedlak et al.'s MInDPRO implementations, all the assembly stations and their instances each have their own sets of tooling, making the number of assembly toolsets dependent on the number of assembly stations and their allowed number of instances. The toolset variables in Table 7.5 are defined the way they are so that there is an adequate number of tools and tooling for the fabrication lines in each subassembly line. The toolset variables and the fabrication task type number variables, both the ones for the station instances and the number of worker sets, are all defined in broad, general terms because to do otherwise results in there being too many variables to optimize. Ideally, each fabrication line would have a dedicated variable for its number of toolsets and every single station would have variables that set their individual number of instances and number of worker sets. However, the former would double or triple the number of toolset variables and the latter would add, on average, 80 new variables since there are approximately 40 stations for every manufacturing process. It is nigh impossible to create adequate surrogate models with that many

variables and similarly would take far too long to explore that design space manually and conventionally. And indeed, this is one of the weaknesses of MInDPRO implementing its factory production flow model in Simio: it is very manual in nature and requires a significant number of inputs from the user despite its capability. Those variables are the way they are in Table 7.5 because they are a compromise between having too many variables and not having enough to obtain sufficient granularity when optimizing the production system.

With the Design of Experiments variables explained, a combination Latin Hypercube, Box Behnken, and fractional-factorial face-centered central composite Design of Experiments was created. 5025 cases were created to explore the interior of the factory production flow design space and as much of the edges as practically possible for its optimization down the line. The data resulting from the Design of Experiments is afterwards used to create the ANN surrogate models in the JMP software. The surrogate models for the Simio factory production flow model are made in the same way as the ones for FLOPS, SEER, and RADE, using the same activation functions, number of neurons, layers of neurons, learning rates, etc. The resulting surrogate models have very high accuracy with training and validation R^2 values of at least 0.97 and the majority tending more towards 0.99 with no noticeable patterns in the residuals.

MInDPRO uses the NSGA-II to create its production rate and cost Pareto fronts. MInDPRO's variables are the same ones listed in Table 7.5. Because its variables are primarily continuous ones and it is desired to keep them real-coded like with INFRADAT, its chromosomes are $1 \times a$ vectors, with a being the number of design variables it has and includes its aircraft and manufacturing design variables and is, in this case, equal to 19. For the initial population, the aircraft design portion of the chromosome is generated by randomly picking a value within the range of the variable for each aircraft design gene. The manufacturing design portion of the chromosome is generated the same way but the value of each gene is rounded to the nearest whole number since those variables must be whole numbers. For crossover, the same whole arithmetic crossover operator used for INFRADAT's

aircraft design chromosome is used. For mutation, one of three different mutation schemes is selected should the individual be selected for mutation: only the aircraft design portion of the chromosome is mutated, only the manufacturing design portion of the chromosome is mutated, or everything is mutated. For the aircraft design portion's mutation, the same real-coded Gaussian mutation operation used by INFRADAT for its aircraft design chromosome is utilized. And for the manufacturing design portion's mutation, random integer values are selected for all the manufacturing design genes. However, the number of worker sets for a fabrication task type is never allowed to exceed the number of station instances for that task type, which also applies to crossover and the initial population generation. An initial population of 200 is used, just like with INFRADAT. The crossover rate and the mutation rate are 80% and 10% respectively, also like with INFRADAT. The generation limit is set at 250, just like with Oesterle et al.'s NSGA-II implementation in Experiment 1. MInDPRO handles its aircraft constraints in the same way INFRADAT does: it regenerates aircraft designs to try to meet the constraints and, after a certain number of retries, applies a penalty to the individual's throughput and cost based on how many constraints were violated and to what degree they were violated.

The number of wingboxes to be produced for the production run remains at 2000 for both MInDPRO and INFRADAT, just like in Experiments 1 and 2. The wage rate for the labor costs in MInDPRO is the same as it is for INFRADAT at $200 an hour for each worker. The equipment costs for all of the stations that use equipment are calculated in the same way as INFRADAT. The total tooling cost was discussed in subsection 7.3.1. The individual cost of the tooling sets for each fabrication line is equal to the cost of the cure task tooling used for that fabrication line's parts for the composite processes and the cost of the NDI task tooling for that fabrication line's parts for the CNC Machining process. This is akin to INFRADAT using the cost of its cure task or NDI tooling for all of its individual tooling costs and it is because both frameworks assume the usage of common tooling that is transferable and transportable between stations, which tends to be a mandrel of some sort

or tooling that it large enough to accommodate the parts everywhere.

The design variables for MInDPRO and their ranges are the same as those used for the creation of its factory production flow surrogate models in Table 7.5 but with the addition of including the five manufacturing processes examined by INFRADAT in Experiment 2. They are relisted in Table 7.6 with the noted addition for completeness's sake.

Table 7.6: Design Variables and Ranges for MInDPRO in Case Study

Variable	**Range**
Manufacturing Process	HLU/Autoclave, ATL, SRI, VARTM, CNC Machining
Wing Area (ft^2)	240-410
Aspect Ratio	3.84-6.24
Taper Ratio	0.36-0.66
Range (nmi)	600-1000
Number of Spar Fab Line Toolsets	2-5
Number of Skin Fab Line Toolsets	2-5
Number of Rib Fab Line Toolsets	2-5
Number of Assembly Station Instances	1-4
Number of Fab Task Type 1 Station Instances	1-4
Number of Fab Task Type 2 Station Instances	1-4
Number of Fab Task Type 3 Station Instances	1-4
Number of Fab Task Type 4 Station Instances	1-4
Number of Fab Task Type 5 Station Instances	1-4
Number of Assembly Station Worker Sets	1-4
Number of Fab Task Type 1 Worker Sets	1-4
Number of Fab Task Type 2 Worker Sets	1-4
Number of Fab Task Type 3 Worker Sets	1-4
Number of Fab Task Type 4 Worker Sets	1-4
Number of Fab Task Type 5 Worker Sets	1-4

The design variables for INFRADAT are very similar to the ones used in Experiment 2 but with the possible options for the spar shapes and factory sizes opened up. They are listed in Table 7.6. Of note is that it is desired to observe what other tradeoffs can be obtained via integral fabrication and, by proxy, subassemblies. Eliminating part combinations altogether was already explored in Experiment 1 and so it desired to see the effects of expanding the number of possible part combinations. As a result, the possibility of having combinations

of 4 major parts in a subassembly or fabricated integrally is added to see what impact this would have assuming future technology makes it more feasible than it currently is. To balance this out, the tooling costs are increased substantially for all major parts and their tasks involved in any 4-major-part integral fabrication combinations.

Table 7.7: Design Variables and Ranges for INFRADAT in Case Study

Variable	**Range**
Manufacturing Process	HLU/Autoclave, ATL, SRI, VARTM, CNC Machining
Wing Area (ft^2)	240-410
Aspect Ratio	3.84-6.24
Taper Ratio	0.36-0.66
Range (nmi)	600-1000
Spar Shapes	In-C, Out-C, Z, J, I
Factory Size Spatial Constraints (ft^2)	333k, 500k, 1M
Assembly Sequence	All Feasible Sequences
Assembly Line Balance	All Feasible Line Balances
Integral Fabrication & Subassembly Part Combinations	3 Major Parts Max, 4 Major Parts Max

Aircraft design performance constraints can force the aircraft's geometry to look a certain way in order to meet these constraints, which can run counter to the geometric needs for a high throughput and low cost design. Their incorporation thus brings forth a slew of different possible tradeoffs. They have been mentioned but not actually examined up until this point and so are included in the Case Study to see what kinds of tradeoffs the two frameworks are able to make with them. 3 different constraint cases will be looked at. The first is the baseline case, where no constraints are imposed; this was the case used by default for Experiments 1 and 2. The second case is focused on improving the F-86's combat capabilities. The F-86 has a baseline turning radius of about 15,000 feet at its combat altitude of 35,000 feet while flying at Mach 0.8, a rate of climb of approximately 1000 feet per second at that altitude as well, and a range of 804 nautical miles [225]. Constraint case 2 bumps those capabilities up such that the minimum turning radius is reduced to 13,500

feet, the rate of climb is increased to 1500 feet per second, and the range increased to 900 nautical miles to make it a more effective fighter. It is desired to affect different aircraft design variables for constraint case 3 to see what different kinds of tradeoffs are imposed and so the F-86's takeoff and landing capabilities are focused on for that one. The F-86's baseline approach velocity V_{app} for landing is roughly 130 knots, its Takeoff Field Length (TOFL) to clear a 50 foot obstacle is 3770 feet, and its Landing Field Length (LFL) from 50 feet is 3450 feet. For constraint case 3, to improve its takeoff and landing qualities, the maximum V_{app} is set to 120 knots, the TOFL is reduced to a maximum of 3350 feet, and the LFL is reduced to a maximum of 3250 feet. This would allow the F-86 to take off from and land on shorter runways. To maintain its combat capabilities, constraints are also set on its turning radius and rate of climb to be equal to the default case, while the minimum range is set to 650 nautical miles. A minimum range of 804 nautical miles with all these other constraints yielded an empty design space. These constraints are summarized in Table 7.8.

Table 7.8: Aircraft Design Performance Constraint Cases and Their Constraints in Case Study

Constraint Case Number and Description	**Performance Constraints**
1: Baseline	No Constraints
2: Improved Combat	1500 ft/s Max Rate of Climb at Combat Altitude, 13.5k ft Turn Radius at Combat Altitude at Mach 0.8, 900 nmi Range
3: Improved Takeoff and Landing w/ Baseline Combat	120 kt V_{app}, 3350 ft TOFL, 3250 ft LFL, 1000 ft/s Max Rate of Climb at Combat Altitude, 15k ft Turn Radius at Combat Altitude at Mach 0.8, 650 nmi Range

For the procedure, it is first desired to check which framework is able to meet the first condition laid out in section 7.2. To do this, MInDPRO is used to generate Pareto fronts for all 5 manufacturing processes and INFRADAT is used to do the same. For this initial comparison, since MInDPRO does not consider spatial constraints or different spar shapes,

INFRADAT will only consider the 333k ft^2 spatial constraint so as to not have too high of a throughput and possibly bias the $I_\epsilon(PF_1, PF_2)$ values. It will also only use the Out-C spar shape since that is the one MInDPRO uses. To further analyze the effects of assembly sequencing and line balancing, 2 more sets of INFRADAT Pareto fronts are generated: one where INFRADAT's assembly sequences are set to be the same as MInDPRO's, one where both its assembly sequences and line balances are the same as MInDPRO's. All of these INFRADAT Pareto fronts are compared with MInDPRO's.

The second half of the procedure pertains to fully demonstrating both framework's capabilities and all their possible tradeoffs to see which is able to meet the second condition laid out in section 7.2. MInDPRO is used to generate two additional sets of Pareto fronts, one for each of the constraint cases. INFRADAT is used to generate enough additional sets of Pareto fronts to account for all of its possible spatial constraints, spar shapes, constraint cases, and number of major parts in its integral fabrication/subassembly part combinations. The tradeoffs possible from generating all of this for both frameworks are compared. A summary of the procedure for the Case Study is as follows:

1. Use MInDPRO to generate Pareto fronts for all 5 manufacturing processes, which represent its categorical variables

2. Use INFRADAT to generate Pareto fronts for all 5 manufacturing processes with Out-C spar shape, 333k ft^2 spatial constraint, 3-major-parts integral fabrication combinations for baseline comparison with MInDPRO

3. Repeat Step 2 but with identical assembly sequences as MInDPRO for comparison when assembly sequences are the same

4. Repeat Step 2 but with identical assembly sequences and line balances as MInDPRO for comparison when both assembly sequences and line balances are the same

5. Use MInDPRO to generate Pareto fronts for all 5 manufacturing processes and all 3 constraint cases to show MInDPRO's full capabilities

6. Use INFRADAT to generate Pareto fronts for all 5 manufacturing processes for all 5 spar shapes, 3 spatial constraints, 3 and 4 major parts integral fabrication part combinations, and 3 constraint cases to show full capability of INFRADAT

7. Compare full tradeoff capabilities of both frameworks

The primary assumption for the Case Study is the same as for Experiments 1 and 2, which is that the assembly line is completely pulsed with all parts being at the next station after an amount of time equal to the cycle time has passed. The automated ground vehicles that transport the parts are not modeled and their costs not accounted for either.

7.4 Results and Discussion

In all the following results, HLU/A is short for HLU/Autoclave and CNC is short for CNC Machined. An "M" in front of any set of results is short for results generated via MInDPRO and an "I" in front is short for results generated via INFRADAT. Many of the results and their trends are jagged because the nature of genetic algorithms is that they select the best set of designs at every variable or parameter setting without significant regard to the designs that came before it or after it, causing there to sometimes be large jumps in the values used. The issue is compounded due to many of the variables being discrete instead of continuous even though they are seemingly continuous, such as the number of stations or station duplications. These jagged trends are also caused by the fact that the true Pareto front is not known and can only be approximated. What is most significant is therefore the trends indicated by the distribution or layout of the results as opposed to their exact shape.

7.4.1 First Condition: Better, Justifiable Throughputs and Costs

Figure 7.7 shows the average summary attainment surfaces and $I_\epsilon(PF_1, PF_2)$ distributions for both frameworks while Table 7.9 presents the HVR attained by both frameworks as well as roughly how computationally expensive they both are.

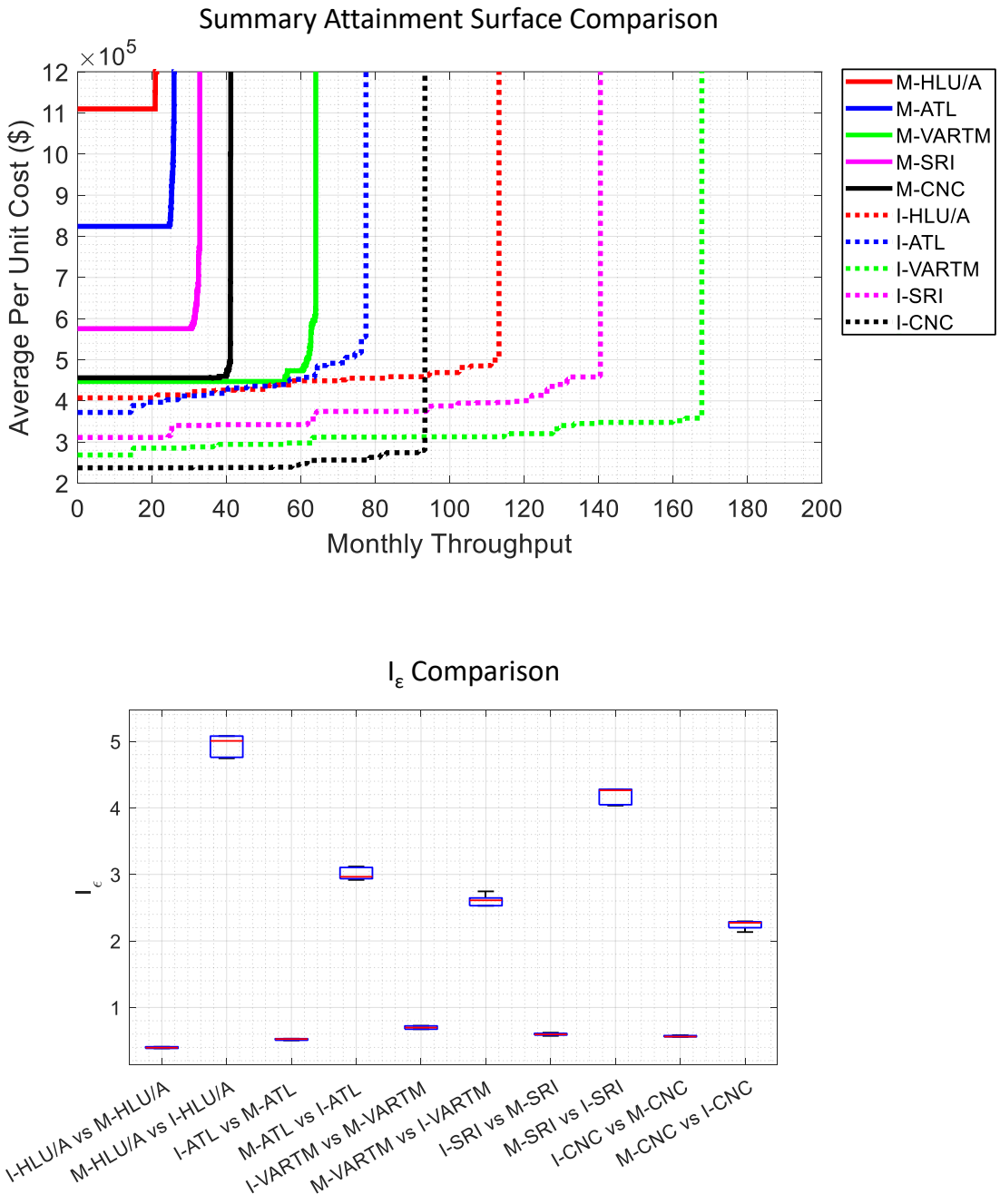


Figure 7.7: Average Summary Attainment Surfaces and $I_\epsilon(PF_1, PF_2)$ Boxplot Distributions Comparing INFRADAT (I) and MInDPRO (M) for All Processes

Table 7.9: Hypervolume Ratio and Average Number of Generations for MInDPRO and INFRADAT Results for All Processes

Manufacturing Process	**MInDPRO HVR**	**MInDPRO Average Number of Generations**	**INFRADAT HVR**	**INFRADAT Average Number of Generations**
HLU/ Autoclave	0.1216	250	1.00	245
ATL	0.1989	250	1.00	254
VARTM	0.2120	250	1.00	263
SRI	0.1679	250	1.00	489
CNC Machined	0.3234	250	1.00	442

Several trends are immediately noticeable. Given the distribution of $I_\epsilon(PF_1, PF_2)$ values as well as the HVR values of unity for the INFRADAT results, INFRADAT is shown to be able to produce designs with much lower cost and higher throughput than MInDPRO. Even the results for the worst performing manufacturing process, the ATL process, are able to completely dominate and encapsulate MInDPRO's best results. In terms of computational expense, the two frameworks have roughly the same expense in terms of number of generations required when INFRADAT uses the two step approach, which was done for the HLU/Autoclave, ATL, and VARTM processes. When the two step approach was not used, the INFRADAT results took longer to obtain. In terms of the trends with the manufacturing processes themselves, VARTM is the best process overall based on MInDPRO's results while HLU/Autoclave performs the worst for MInDPRO and ATL is the worst overall for INFRADAT. With INFRADAT, the CNC Machining process yields the lowest cost designs while the VARTM process yields the highest production rates, with the highest ones yielding 2.5 times more wingboxes than MInDPRO while also costing noticeably less.

INFRADAT is thus able to meet the first part of the first condition explained in section 7.2. To meet the second part and fulfill the entire first condition, it must be able to justify its results. A large number of diagnostic results are subsequently generated. The first set generated are related to how the workstations are laid out and how efficiently they

are used. These are shown in Figure 7.8, Figure 7.9, Figure 7.10, Figure 7.11, and Figure 7.12.

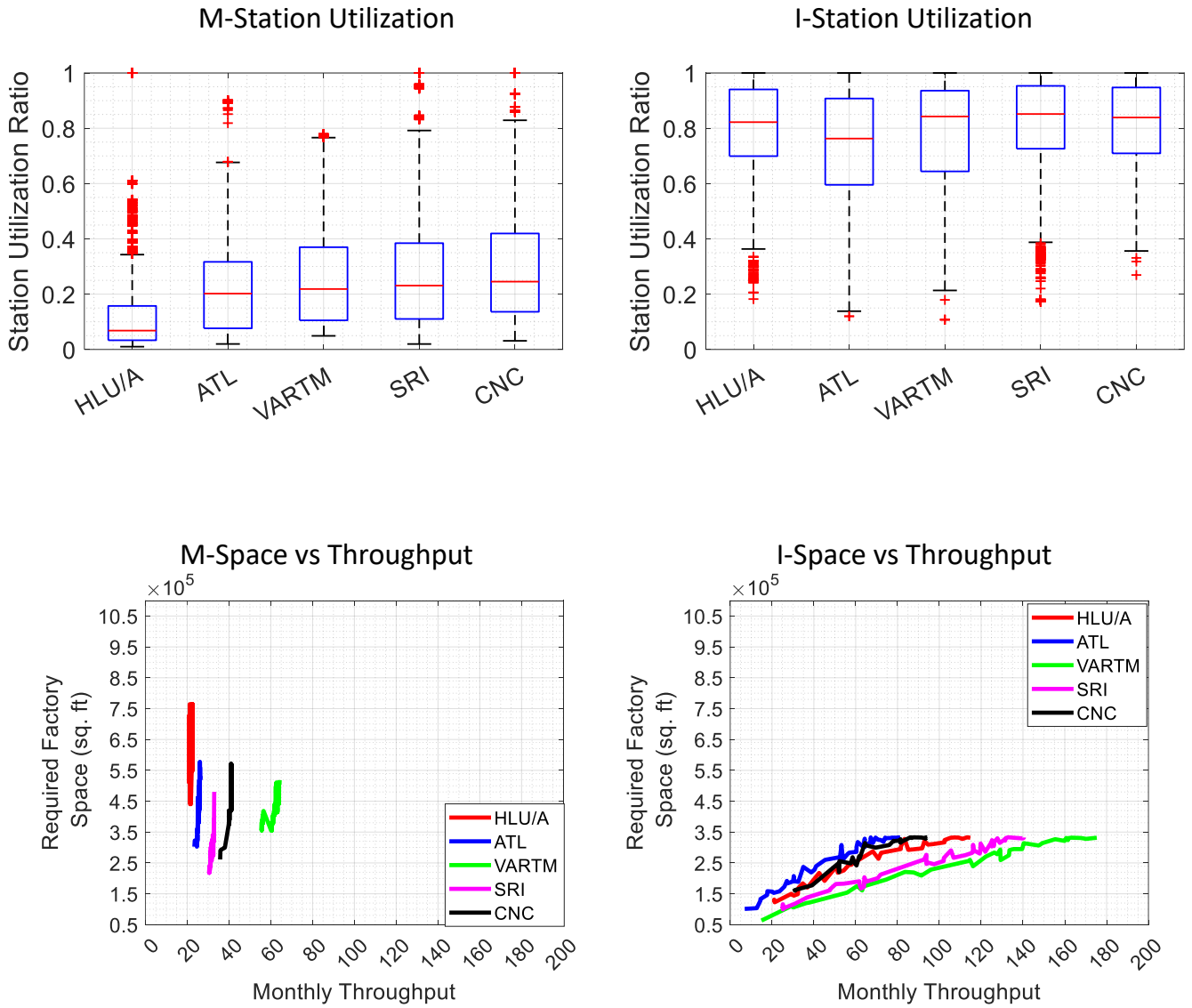


Figure 7.8: Boxplots Showing Station Utilization Ratio and Graph Showing Space Used as a Function of Throughput for MInDPRO and INFRADAT Results for All Processes

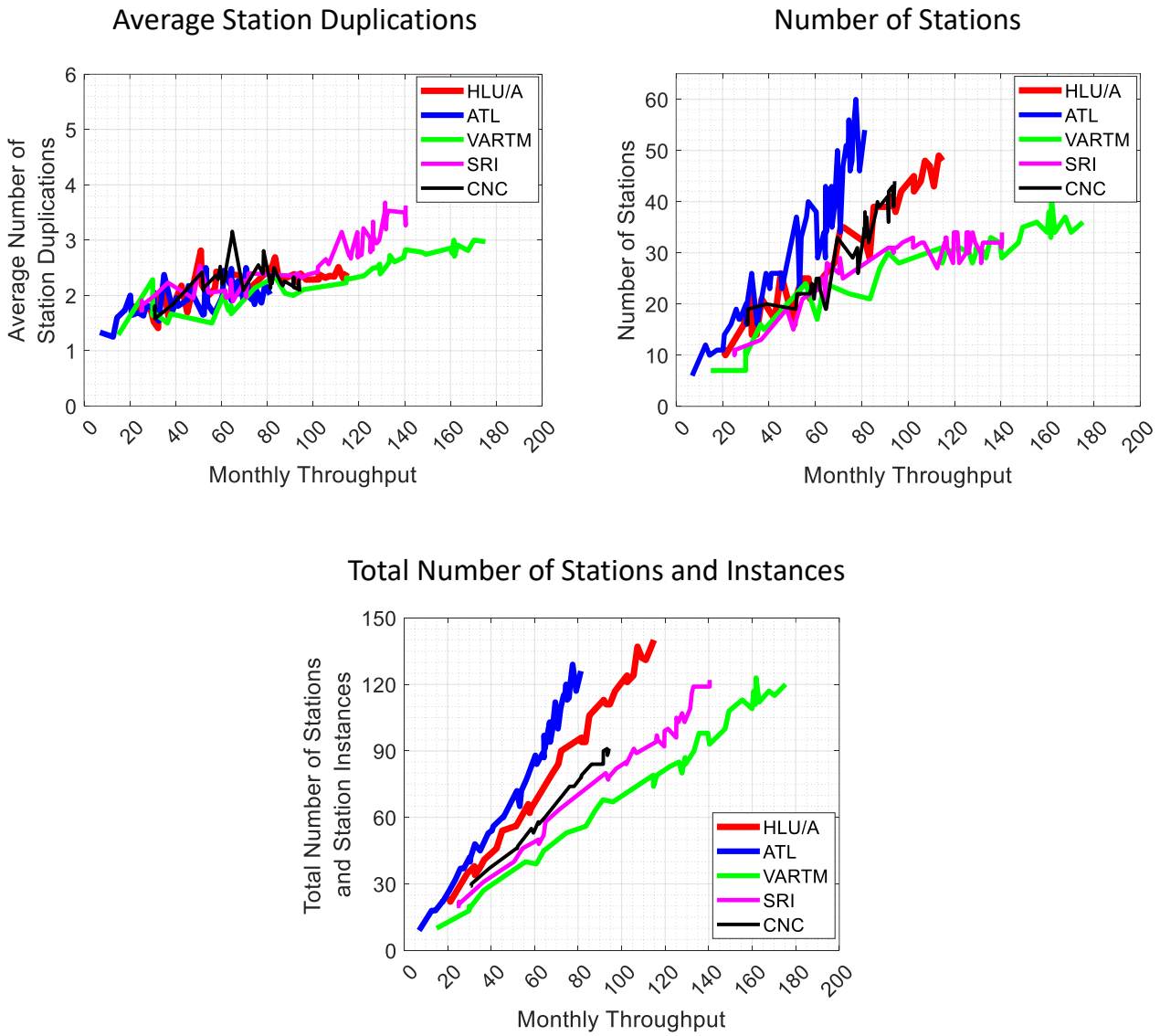


Figure 7.9: Average Number of Station Duplications, Number of Stations, and Total Number of Stations and Station Instances as a Function of Throughput for INFRADAT Results for All Processes

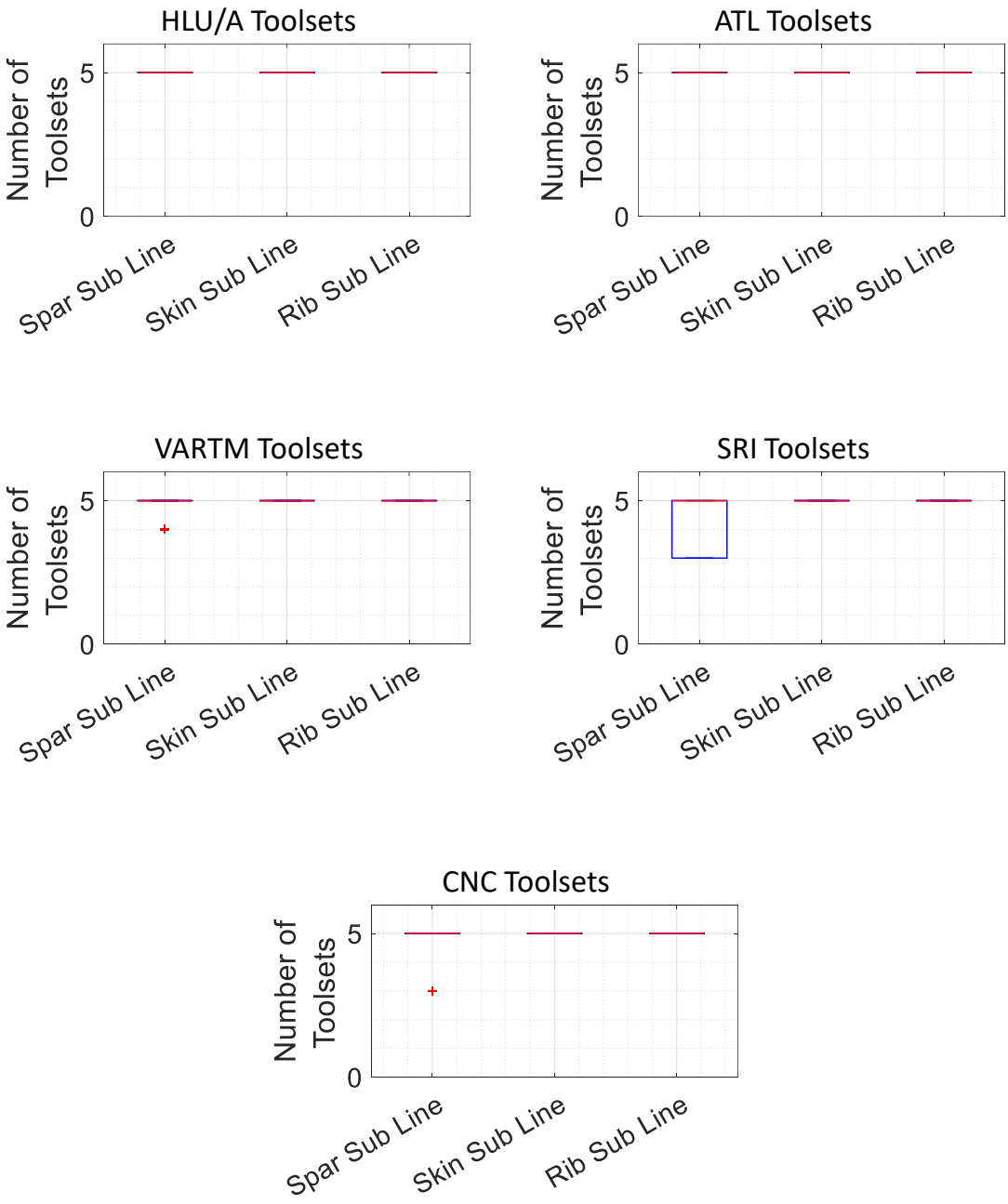


Figure 7.10: Boxplot Distribution of Number of Toolsets for MInDPRO Results for All Processes

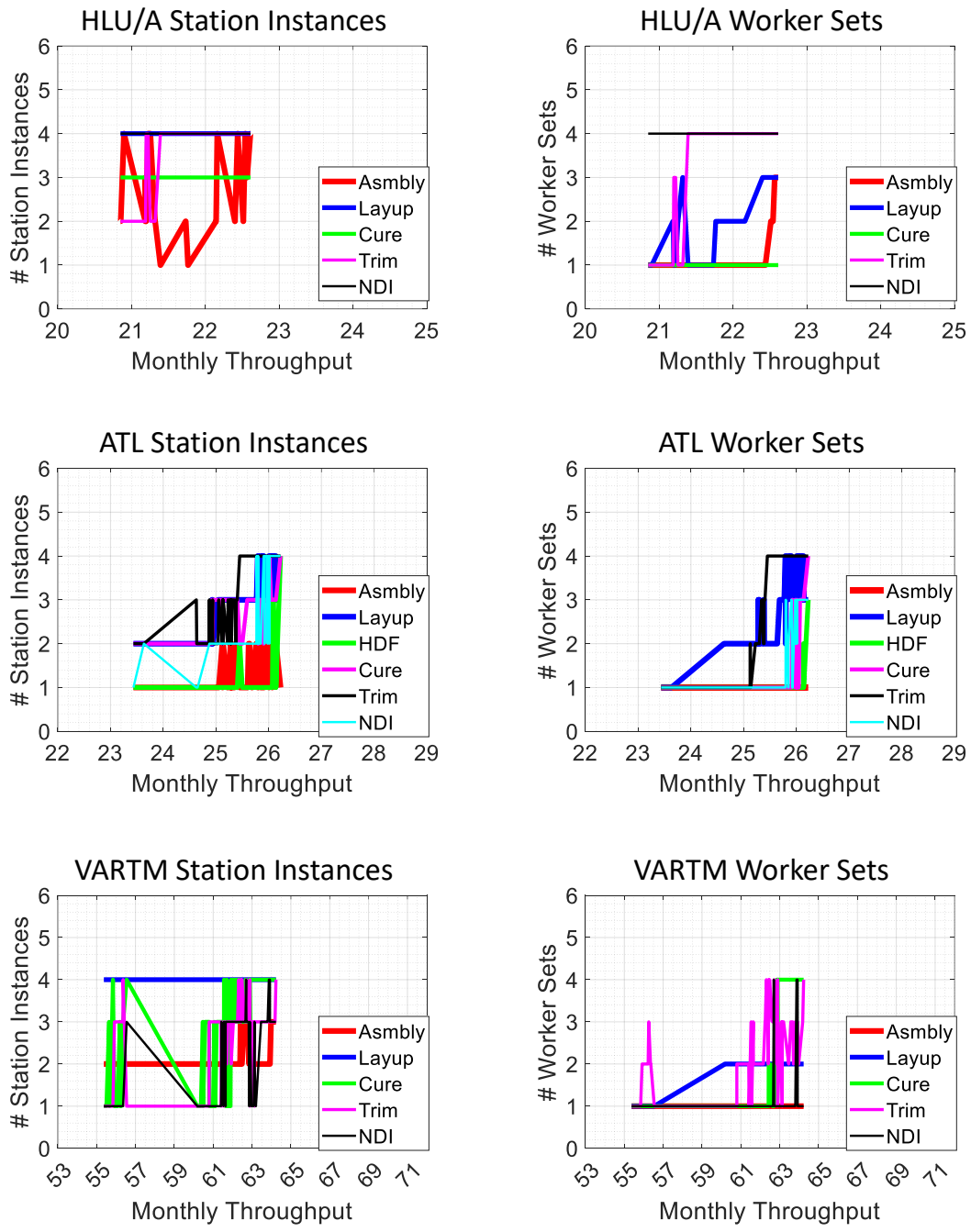


Figure 7.11: Number of Station Instances and Worker Sets as a Function of Throughput for MInDPRO Results for HLU/Autoclave, ATL, VARTM Processes

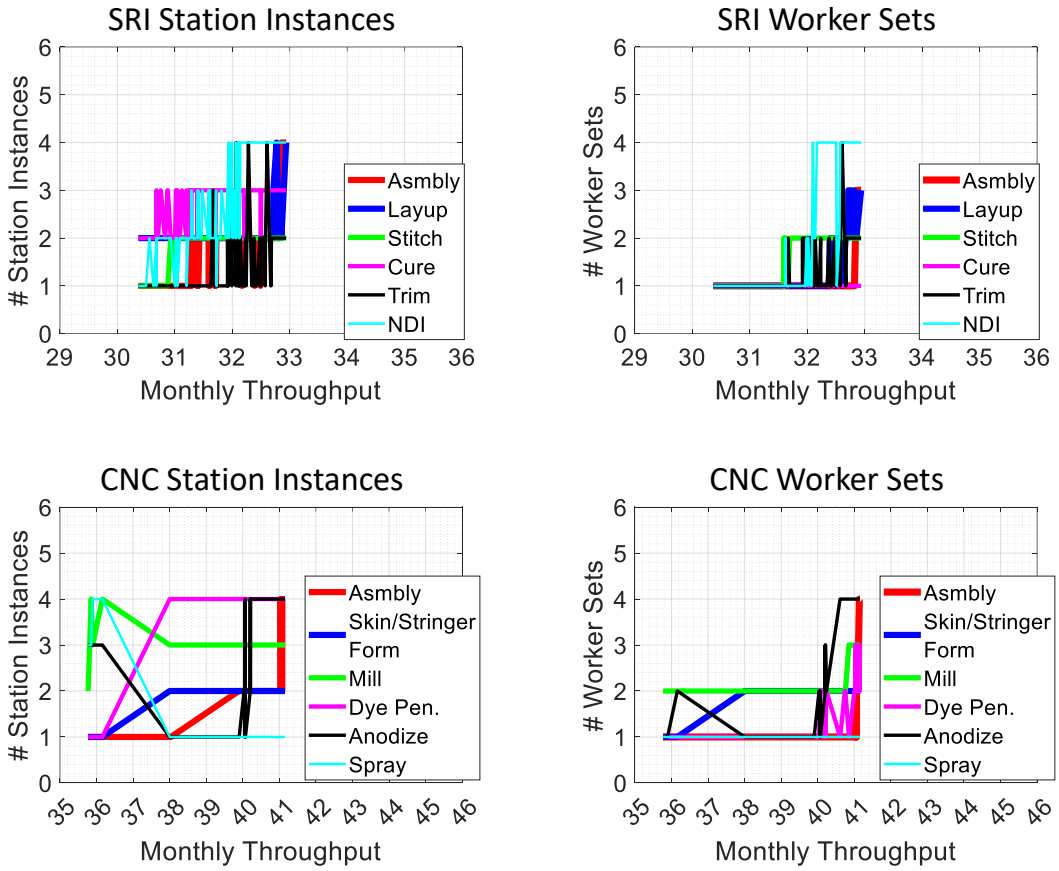


Figure 7.12: Number of Station Instances and Worker Sets as a Function of Throughput for MInDPRO Results for SRI, CNC Machined Processes

One of the most apparent reasons why INFRADAT's results perform better than MInDPRO's is in Figure 7.8, which shows that INFRADAT has significantly higher station utilization and therefore labor efficiency. There is less idle time at each station, with workers being more productive overall and so lowering cost significantly. This is further emphasized by examining Figure 7.11 and Figure 7.12. One of MInDPRO's advantages is its resource tracking, in this case worker tracking, allowing fewer workers to work on a higher number of stations, as shown by the number of station instances being generally higher than the number of worker sets for those stations in those figures. However, what this actually means is that a station instance is taking up valuable space and incurring an equipment cost while not being operated on for the full time. Every station instance must essentially be able to pull its own weight and full contribute to the production, which not all of MInDPRO's station instances are able to do, resulting in very low station and labor efficiencies that not only increase the cost but reduce production rate as well.

The reason this occurs can be seen in Figure 7.10, which indicates that the number of toolsets needed to maximize production rate and minimize cost for MInDPRO almost always involves the maximum number available. In a DES, a station needs resources in the form of workers and toolsets for its tasks to be carried out. The number of worker sets is not equal to the number of station instances to fully man all the instances at all times because there are not enough toolsets available for them to do so. The argument can be made to simply increase the available number of toolsets to solve this problem. However, the weakness with this is that, before the simulations are run, it is unknown what the appropriate variable limits are. Being forced to do multiple run-throughs to see what the variable ranges should be involves being forced to create multiple sets of surrogate models, one for each run-through, which is a time-consuming process. This is a general weakness of the MInDPRO framework, that many of the important variables and their ranges must be declared by the designer or subject matter expert who may turn out to not be correct.

This is as opposed to the INFRADAT framework, which has all of the stations and

their instances modeled as being fully equipped with the appropriate equipment, tooling, and workers. The number of stations and their instances can be dynamically varied and calculated, as shown in Figure 7.9, and does not rely on outside input at all. It is able to progressively determine that tasks should be split up into separate stations, increasing the number of stations, to reduce station times while incrementally duplicating bottleneck ones to decrease their effective station times, trading space for increased production rate. The extent of the duplication increases more and more once tasks can no longer be split up into additional stations. This is one of the many advantages with internally performing the assembly sequence planning and line balancing as opposed to having it be required to define them explicitly before the modeling can be done like with MInDPRO. In particular, because it is done implicitly as opposed to explicitly, INFRADAT is able to specify the exact stations that need to be duplicated to improve its production rate as opposed to being limited in terms of pre-defined variables and forced to duplicate stations in groups of stations for MInDPRO.

This forced blanket increase of an entire group of station's instances is why, for the majority of MInDPRO's different station types, there are less worker sets than station instances; only one of the stations in the station type is a bottleneck and needs to have its number of instances increased, but being grouped means all the other stations that do not need to be duplicated must be duplicated. And this cannot be made more granular with the way MInDPRO is currently implemented because it would require each of the, at minimum, 36 different stations from Figure 7.5 have their own variable and a separate variable for their number of worker sets. As mentioned earlier, this is intractable due to it being extremely difficult to make accurate surrogate models in a timely manner with such a large number of design variables. As an example, even with just a Box Behnken Design of Experiments, which requires one of the fewest cases to be run, 72 variables need over 10,000 cases to be run, which is more than double the number of cases used to make MInDPRO's factory production flow surrogate models that utilized a Box Behnken, fractional factorial

face-centered central composite, and Latin Hypercube Design of Experiments. And within any further cases, the design space for as complicated of a problem as the one being tackled will not be adequately mapped out for the surrogate models to be accurate.

MInDPRO was not built to be able to accommodate spatial constraints and Figure 7.8 fully demonstrates this. In a real factory with limited space, the majority of its designs as well as several entire manufacturing processes become completely infeasible given its pre-defined assembly sequences and line balances. This is in contrast to INFRADAT which never exceeds the given space and has many different designs available that use progressively less space in exchange for lower throughput. All these factors discussed above result in the labor efficiency being much higher for INFRADAT, with its workers and stations and their productivity being fully utilized. This gives the INFRADAT results higher production rates and lower costs, but is not the entire story.

Additional diagnostic results more related to the assembly sequence and tasks themselves are shown in Figure 7.13, Figure 7.14, Figure 7.15, and Figure 7.16.

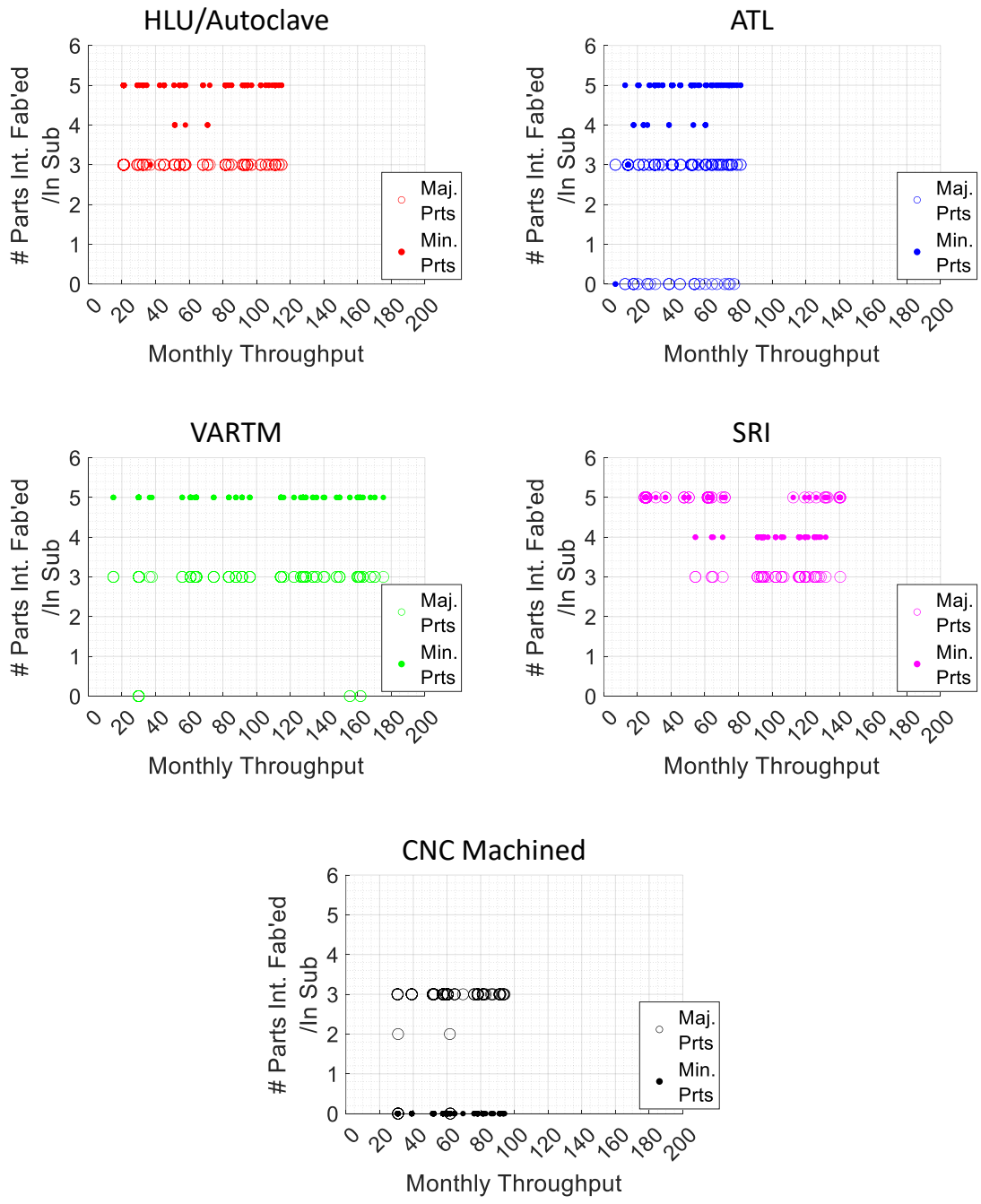


Figure 7.13: Number of Minor and Major Parts Integrally Fabricated or in a Subassembly as a Function of Throughput for INFRADAT Results for All Processes

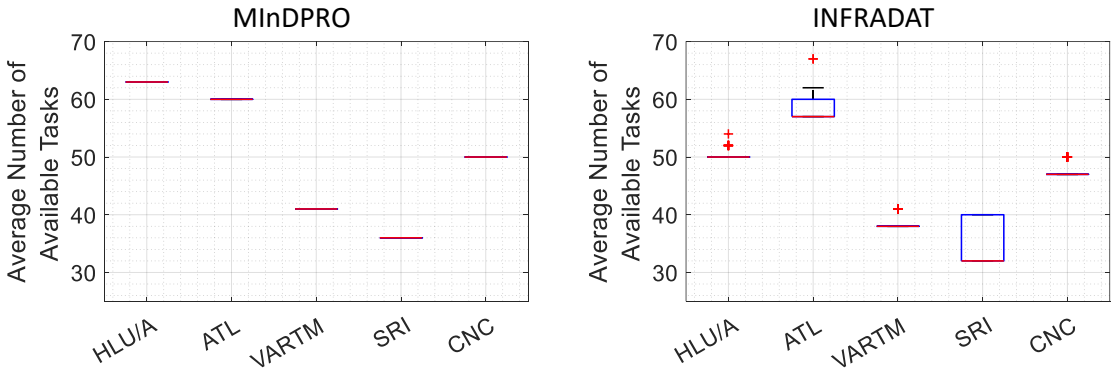


Figure 7.14: Boxplot Distribution Comparison of Average Number of Available Tasks for MInDPRO and INFRADAT Results for All Processes

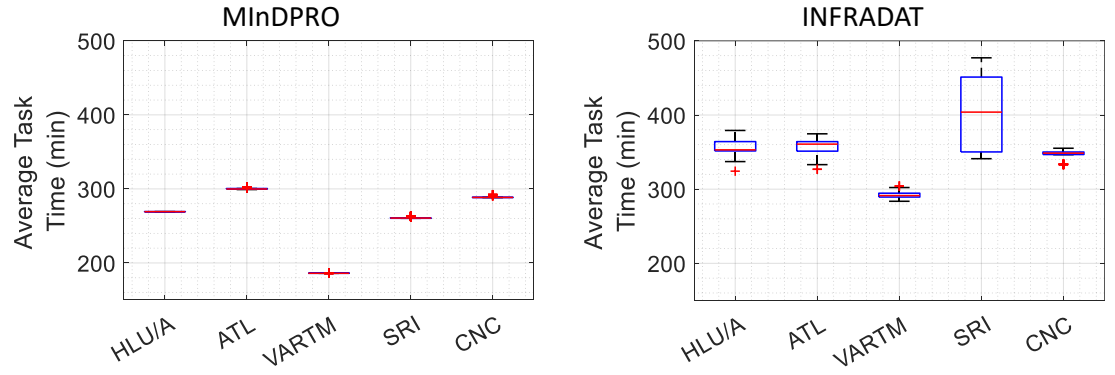


Figure 7.15: Boxplot Distribution Comparison of Average Task Times for MInDPRO and INFRADAT Results for All Processes

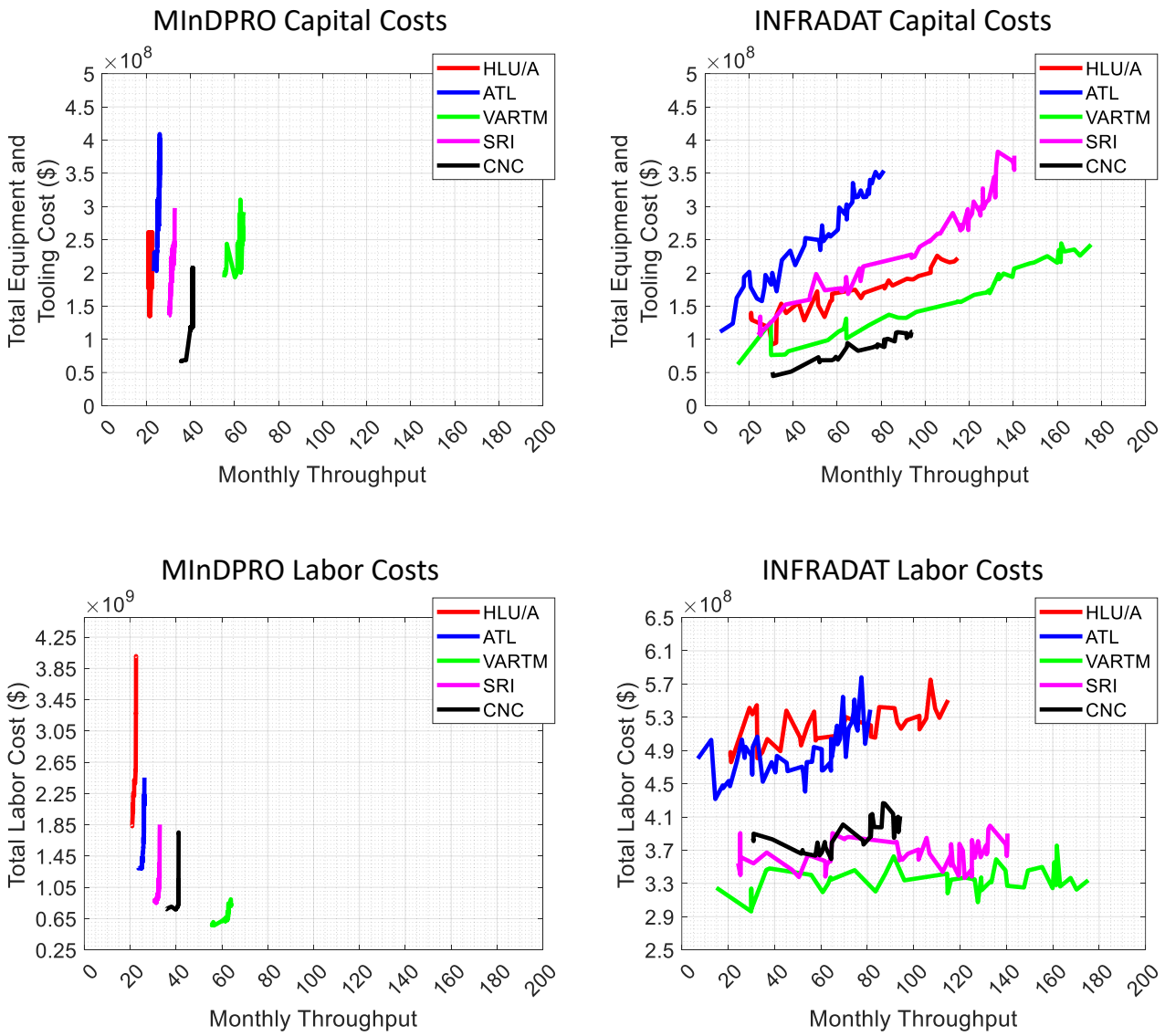


Figure 7.16: Capital and Labor Costs as a Function of Throughput for MInDPRO and INFRADAT Results for All Processes

MInDPRO's assembly sequences are static and defined before the DES for its factory production flow model were ever carried out while INFRADAT's are dynamic and set during the course of the optimization itself. This leads to different assembly sequences and combinations of integrally fabricated parts and subassemblies, as shown in Figure 7.13 by the number of minor and major parts being integrally fabricated or in a subassembly not being completely static while throughput is varied. In fact, based on that figure, it appears to always be advantageous for the majority of the minor parts to be integrally fabricated. While it is possible for the designer or subject matter expert to select a close-to-optimal assembly sequence from the outset, there is no guarantee that they are able to do so. Indeed, for the SRI process, the ideal assembly sequences and part combinations for production rates between 80 100 wingboxes a month has fewer integrally fabricated parts and subassemblies than desired rates that are higher or lower than that, which is an unexpected result and likely marks the shift between from the designs being more throughput oriented to being more cost oriented. Should those throughputs be desired, a designer would not be able to determine a priori what the best assembly sequence would be.

The effects of more dynamically-determined assembly sequences and integral fabrication/subassembly part combinations are more apparent in Figure 7.14. From previous experiments, a lower number of available tasks indicates that fewer stations are needed, leading to lower overall costs and higher throughputs. Due to the ability to select the best assembly sequences and integral fabrication and subassembly part combinations on-the-go, INFRADAT's results on the whole tend to have a lower number of available tasks, making its line balancing subsequently easier as well. This helps to further cement INFRADAT's advantages over MInDPRO and why its results are superior. Though it may seem that MInDPRO's average task times in Figure 7.15 should balance its high number of tasks, this is merely an artifact from removing the tool clean and tool prep times from MInDPRO's tasks and placing those times in separate tasks and stations as well as splitting up the bag and debag times into actual bag and debag tasks, further decreasing the average times. The

task times between the two frameworks thus cannot be directly compared due to some differences in how task times are accounted for, which is a result of differences between DES modeling and more traditional ALB.

Looking at the manufacturing processes specifically, based on Figure 7.7 the worst INFRADAT process is ATL, which has even lower throughputs than the HLU/Autoclave process despite automated tape laying being faster than performing manual hand layups. There are several reasons for this. The first is that the minimum number of available tasks for ATL is higher than HLU/Autoclave, necessitating more stations. Its average task times in Figure 7.15 are roughly the same as the HLU/Autoclave process's average task times, meaning the task lengths do not balance out. The task lengths are not significantly shorter despite automated tape laying being faster because hand layup, despite being slower, is also more manual, meaning having more workers can speed up the task. Automated tape laying is an automated process, meaning it is initially faster but cannot get any faster from dedicating more worker resources towards it. The hand layup task types having 4 workers at their stations versus only 2 at the automated tape layup stations generally means that the entirety of the hand layup tasks, including the setup and cleanup, takes roughly as long as the automated tape layup tasks. The ATL process generally has lower labor costs to reflect this and make up for the relatively similar task times, as can be seen in Figure 7.16. And on a more external level, the ATL automated tape laying task modeled is relatively conservative and likely outdated by modern standards, given that the available data for it is from previous MInD works when the technology was still relatively new. This is evident by its assumed laydown rate of only 50 pounds per hour and it having large creel change times in SEER. Combined with the number of workers factor, this is also likely to cause its task times to not be as low as expected.

Based on examination of Figure 7.9, the total number of stations and their instances for the ATL process is roughly on par with the SRI and VARTM processes and only slightly lower than the HLU/Autoclave process, indicating that the ATL process does not perform

worse because its equipment and stations take up more space. Due to this, the primary reason the ATL process performs worse than the HLU/Autoclave one is that it needs additional tasks, mainly in the form of needing hot drape forming, for its fabrication process to be completed, with its tasks having lower task times but not low enough to counter the number of tasks needed. This consequently causes its throughput to be lower. It additionally has the highest equipment costs in Figure 7.16, causing it to have higher costs overall at high throughputs with its lower labor costs only benefiting it at lower production rates. The HLU/Autoclave task has a higher total number of stations due to its bottleneck stations primarily being the hand layup ones, which are physically smaller than the assembly stations that are the bottleneck for the ATL, SRI, and VARTM processes.

The VARTM process performs better than the SRI one despite the latter being able to use major part integral fabrication to eliminate major part assembly tasks and having a lower number of tasks in general, as shown by Figure 7.13 and Figure 7.14. This is because the SRI process has a significantly higher average task time that outweighs its benefits of having a lower number of available tasks, as seen in Figure 7.15, and in terms of costs has higher capital costs as well in Figure 7.16 due to needing stitching machines that the VARTM process does not. The CNC Machined process at lower throughputs is cheaper than the VARTM one because of its lower capital costs and because of its lower material costs, which were both discussed in Experiment 1.

For MInDPRO's results, the ATL process has lower costs than HLU/Autoclave because its higher capital costs are more than made up for by its significantly lower labor costs in Figure 7.16. It is additionally able to attain slightly higher production rates because, although the number of available tasks is roughly the same and its average task time is slightly higher, based on the distribution of station instances in Figure 7.11 the HLU/Autoclave process has significantly more imbalanced task times than the ATL process, thus necessitating the maximum number of instances more often and making it harder to subsequently line balance. VARTM outperforms CNC Machining and SRI because of its very low average

task time while CNC Machining costs less than SRI due to its lower capital and labor costs. It is able to outproduce SRI despite a slightly higher average task time because the different subassembly lines for the CNC Machining process are mostly parallel in nature while the SRI process's high usage of integral fabrication causes delays at one station for one subassembly line to propagate downstream, decreasing throughput despite other factors suggesting otherwise.

For both frameworks, the autoclave-curing processes perform worse than their oven-curing counterparts because of the latter's lower number of available tasks due to the usage of co-cure integral fabrication as opposed to co-bond, which requires more tasks.

Attention is now turned towards the aircraft design variables and how they are distributed among the two frameworks. One of the primary purposes of INFRADAT is to connect the aircraft design variables with the manufacturing ones and this is how they are manifested. These are presented in Figure 7.17 and Figure 7.18. It is to be noted that no constraints of any kind were placed on the aircraft for this scenario barring the fact that the design variables needed to stay within their respective ranges.

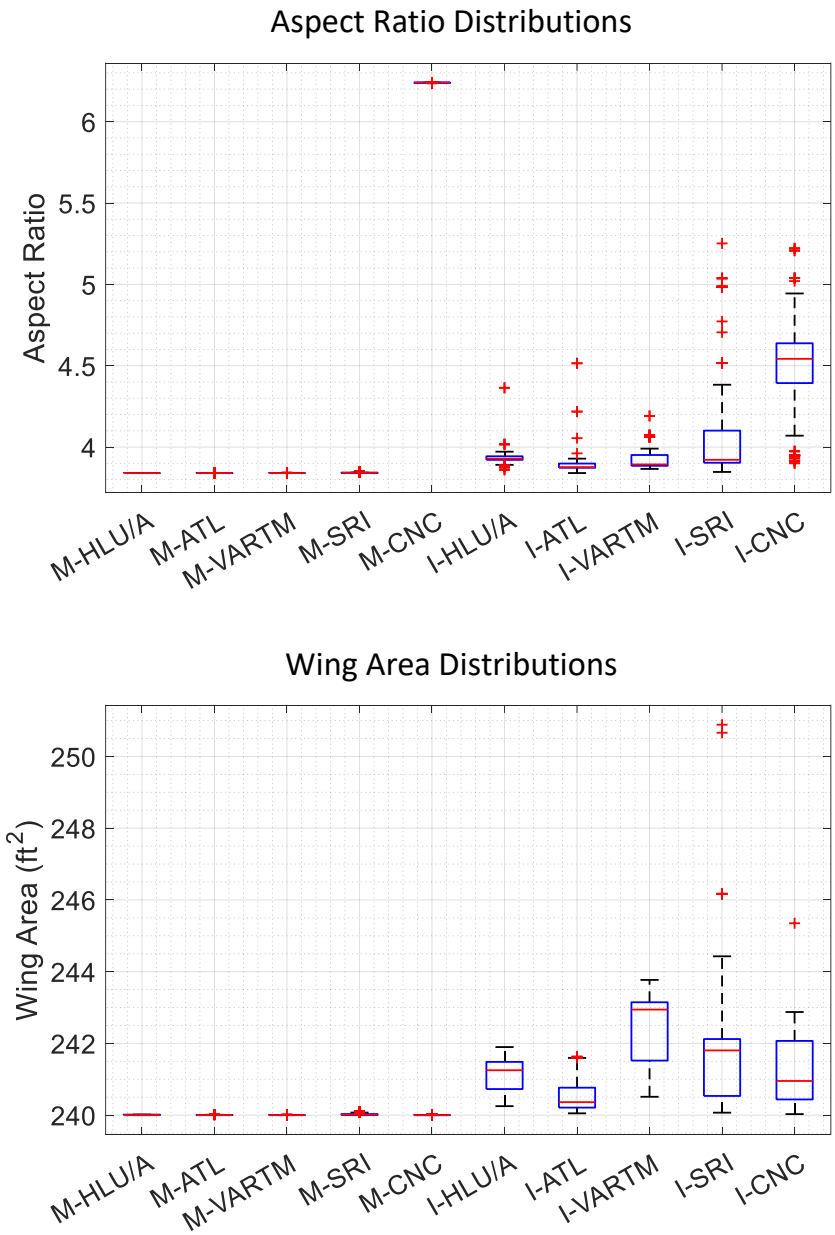


Figure 7.17: Boxplot Distribution Comparison of Wing Area (ft^2) and Aspect Ratio for MInDPRO and INFRADAT Results for All Processes

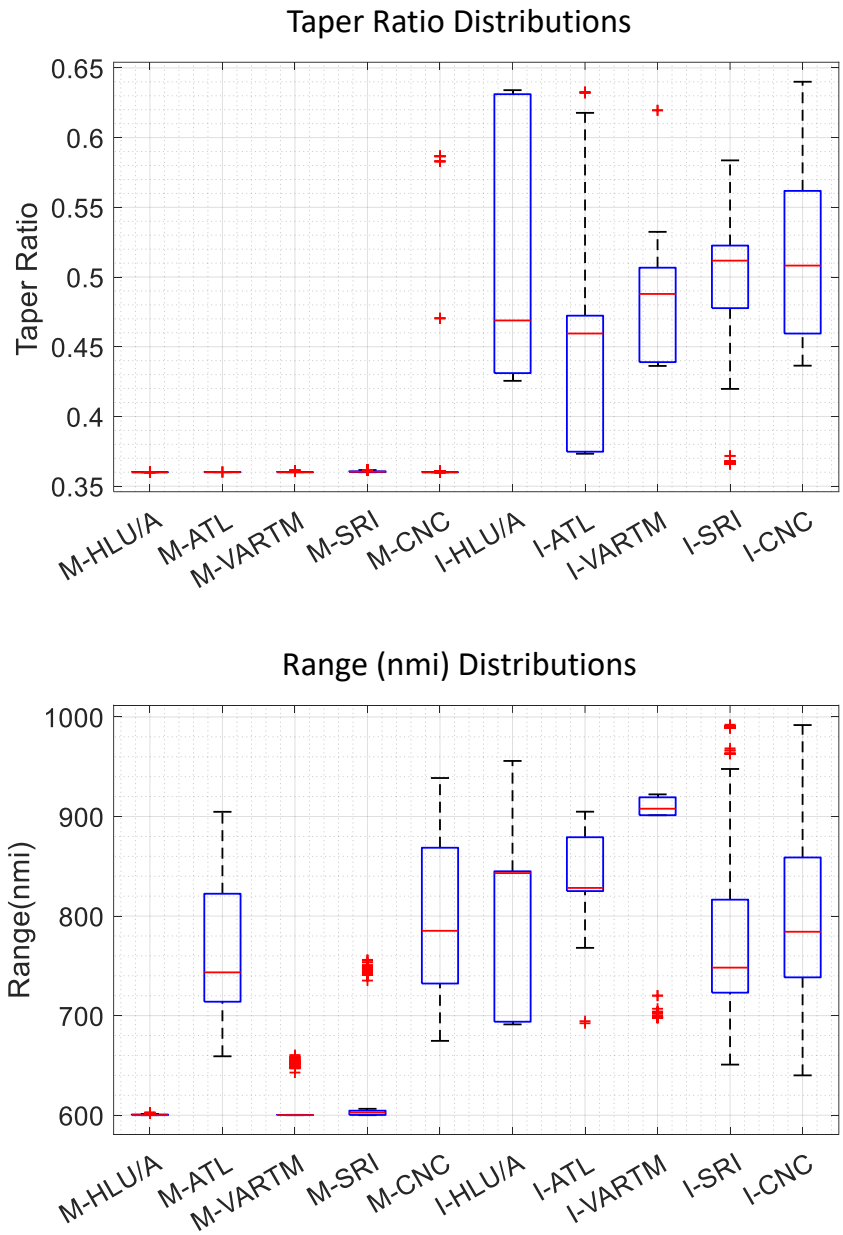


Figure 7.18: Boxplot Distribution Comparison of Wing Taper Ratio and Range (nmi) for MInDPRO and INFRADAT Results for All Processes

What is immediately apparent is that none of the preferred aircraft design geometry variables match those of the baseline F-86 from Experiment 1 due to none of the requisite performance constraints not being set. As a consequence of this, the resulting production rates and costs of the throughput and cost-optimized designs show a 60% improvement over their respective counterparts in Experiment 1, as can be seen by comparing Figure 7.7 and Figure 5.11. This serves to highlight the importance and advantages of being able to consider early aircraft design when trying to optimize for production rates cost-effectively.

Diving into the geometry variable distributions themselves, perhaps the most uniform result is in Figure 7.17 where both frameworks prefer lower wing areas and thus smaller wings. This makes sense as smaller wings use less material and space and so are cheaper to make at very high production rates. Lower aspect ratios and taper ratios are also associated with lower weight and therefore lower material and labor costs due to the parts being thinner. Lower aspect ratios allow the wingboxes to be more square, which is conducive to being more space-efficient for manufacturing, while also reducing the wing's span. Reducing the wing's span eases any structural buckling issues that may arise and so allows the parts at the root of the wingbox to be thinner, reducing weight. Lower taper ratios mean that the wings and wingboxes taper more and have less chord at the tip, reducing bending moments at the root and reducing needed structural strength and weight as well. The aircraft's range is only very weakly correlated with weight. This is because, unlike commercial aircraft, fighter aircraft are sized for combat and so the extra fuel weight needed for extra range tends to not incur any actual structural weight penalties because the structure is sized for far more stringent conditions. This is why the INFRADAT results have few range-related preferences in Figure 7.18 since more effort was spent optimizing other variables that had greater impact on cost and throughput. This is as opposed to MInDPRO where, due to many of its limitations explained above, struggled to find any noticeable improvements with the manufacturing variables and so devoted a great amount of effort to minimizing range, as can be seen with how its results' distribution of range values is

minimized and preferring minimum range.

The most interesting peculiarities are with INFRADAT's results' taper ratio distributions. Given that lower taper ratio reduces weight and so is beneficial to manufacturing, it at first does not make sense that INFRADAT has such a wide range of taper ratios in Figure 7.18. However, given a constant aspect ratio and wing area, increasing taper ratio decreases the size of the root chord. It was explained in section A.5 that, when determining spatial constraints, the parts' average dimensions cannot be used because it would then not fit on the equipment or tooling. Instead, its largest dimensions are used, with the root chord generally determining the spatial requirement width for most of the parts. As a result of this, a higher taper ratio, while inducing weight penalties, reduces the amount of space required for a part, its equipment, and ultimately the station its tasks are in. INFRADAT's results' taper ratio variable distributions are the way they are because of the constant tug-of-war between higher taper ratios and lower spatial requirements for additional station duplications and lower taper ratios and lower wingbox weight and smaller parts for lower cost and higher throughputs. This constant pushing and pulling with the taper ratio is likely why the wing area distribution for INFRADAT's results are not focused entirely at the absolute smallest area.

Another interesting note is that the CNC Machining process's results prefer high aspect ratio wingboxes despite their higher weight. This turns out to be because the bottleneck tasks for the CNC Machining process tend to be the milling tasks for the ribs. With taper ratio and wing area set constant, higher aspect ratios produce smaller root chords as well, decreasing the overall size of the ribs that need to be machined out and so decreasing the task time for the rib milling tasks. Though the number of ribs needing to be machined is increased due to the greater span, this is ultimately made up for with the proportionately larger decrease in overall rib size. MInDPRO takes full advantage of this and maximizes the aspect ratio as results while INFRADAT only increases the aspect ratio to a certain extent. This is because too long of a span can also significantly increase the amount of space used,

and so INFRADAT found the rough variable range that balances out the competing factors of weight and size.

Though the two frameworks have very similar aircraft design capabilities, INFRADAT has shown it is able to flow manufacturing considerations back up to aircraft design with its spatial considerations having a direct effect on the types of aircraft preferred. However, the similarities in aircraft design capabilities also mean that most of the reasons why INFRADAT's results attained better throughput and cost values are not related to aircraft design.

The majority of the available trends from just one set of INFRADAT results have been elucidated and so additional sets are presented to gleam further insight. Figure 7.19, Figure 7.20, and Figure 7.21 depict INFRADAT results for when its assembly sequences are set to be the same as those used by MInDPRO to observe the effects of not being able to dynamically alter the assembly sequences as well as to see by how much INFRADAT's line balancing capabilities are able to improve MInDPRO's results. Figure 7.22, Figure 7.23, and Figure 7.24 depict INFRADAT results for when both its assembly sequences and line balances are set to be the same as MInDPRO's. This is to compare with the INFRADAT results that have no restrictions to see how much of a difference an optimal sequence and line balance make as well as to see how being able to selectively duplicate workstations is able to improve on MInDPRO's results.

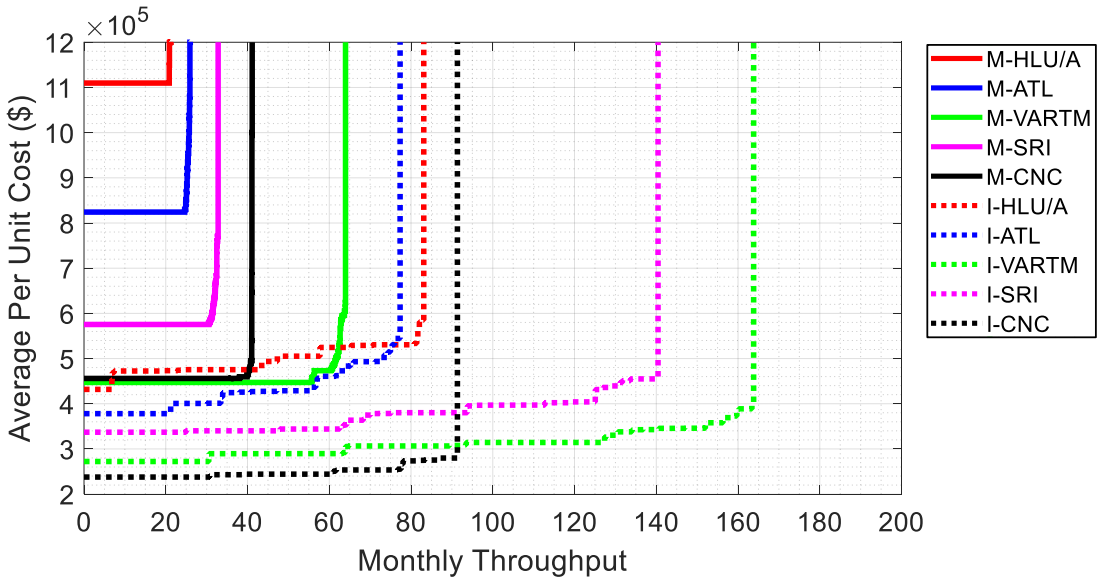


Figure 7.19: Average Summary Attainment Surfaces Comparing MInDPRO Results with INFRADAT Results Limited to MInDPRO's Assembly Sequences for All Processes

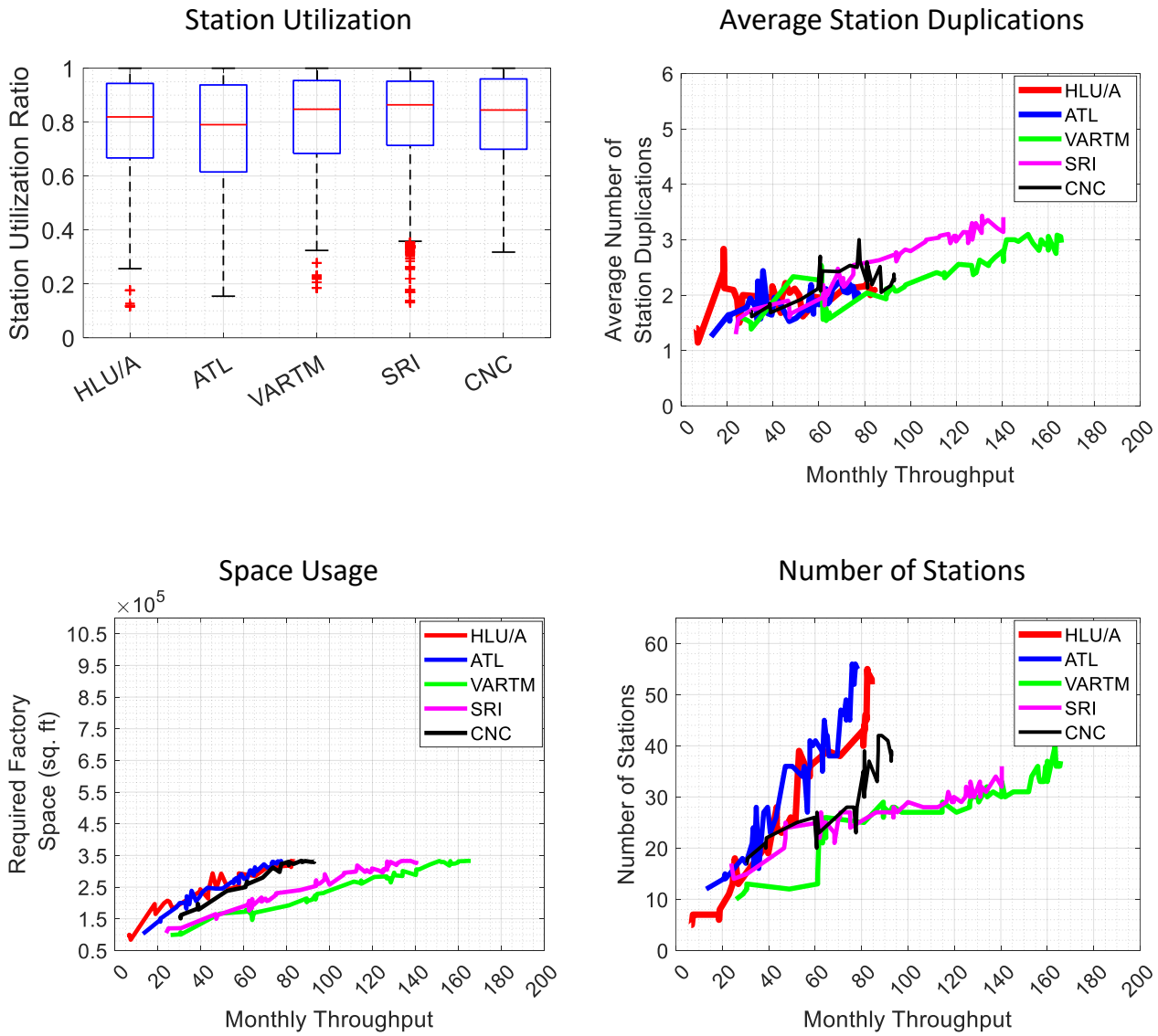


Figure 7.20: Boxplot Distribution of Station Utilization and Average Number of Station Duplicates, Number of Stations, and Space Usage as a Function of Throughput for INFRADAT Results Limited to MInDPRO's Assembly Sequences for All Processes

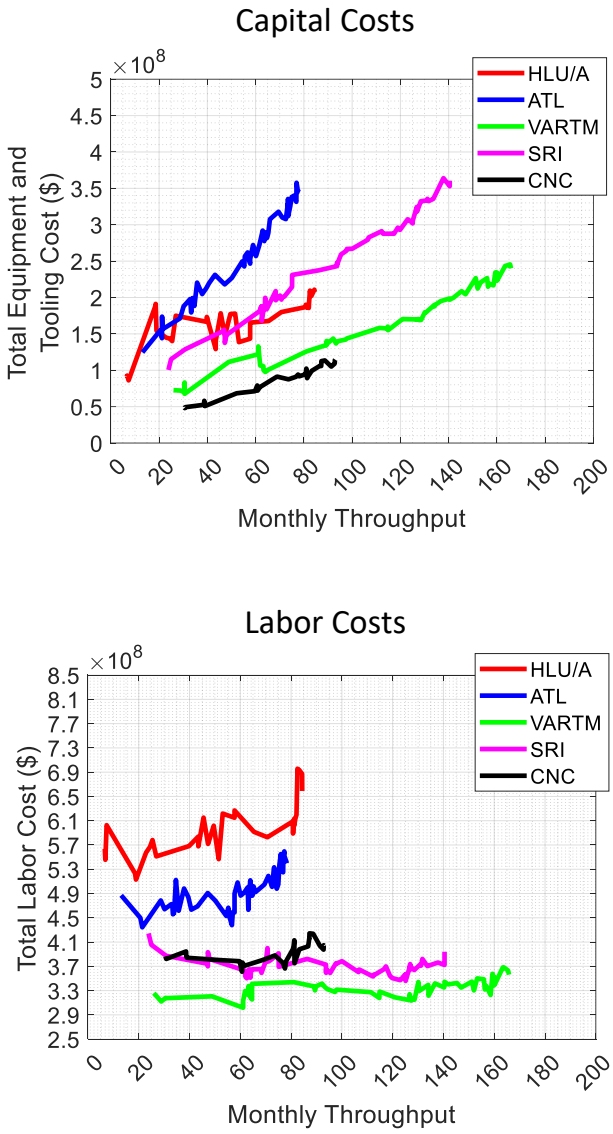


Figure 7.21: Capital and Labor Costs as a Function of Throughput for INFRADAT Results Limited to MInDPRO's Assembly Sequences for All Processes

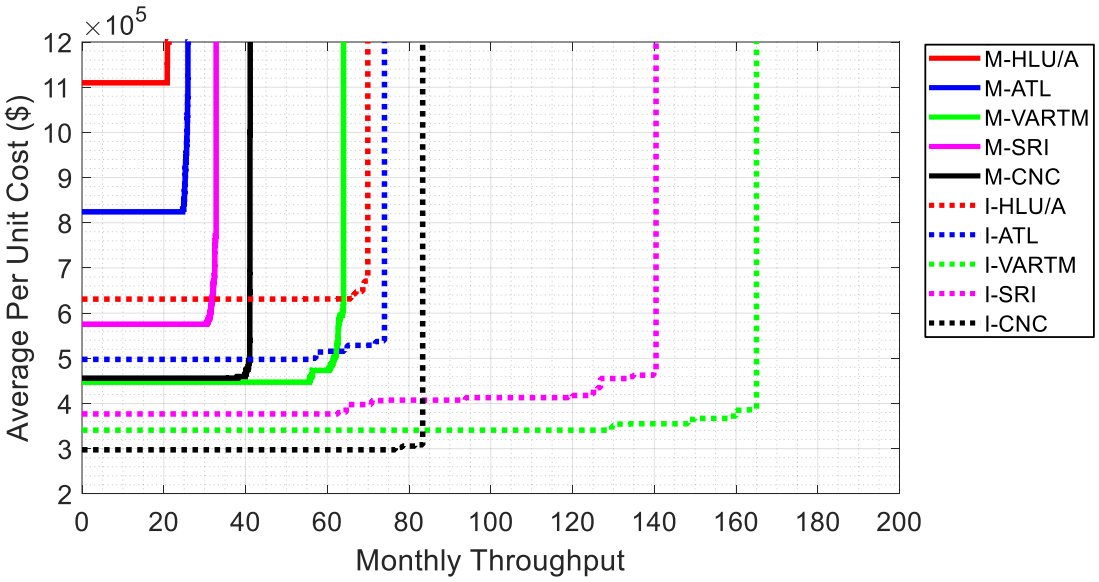


Figure 7.22: Average Summary Attainment Surfaces Comparing MInDPRO Results with INFRADAT Results Limited to MInDPRO's Assembly Sequences and Line Balances for All Processes

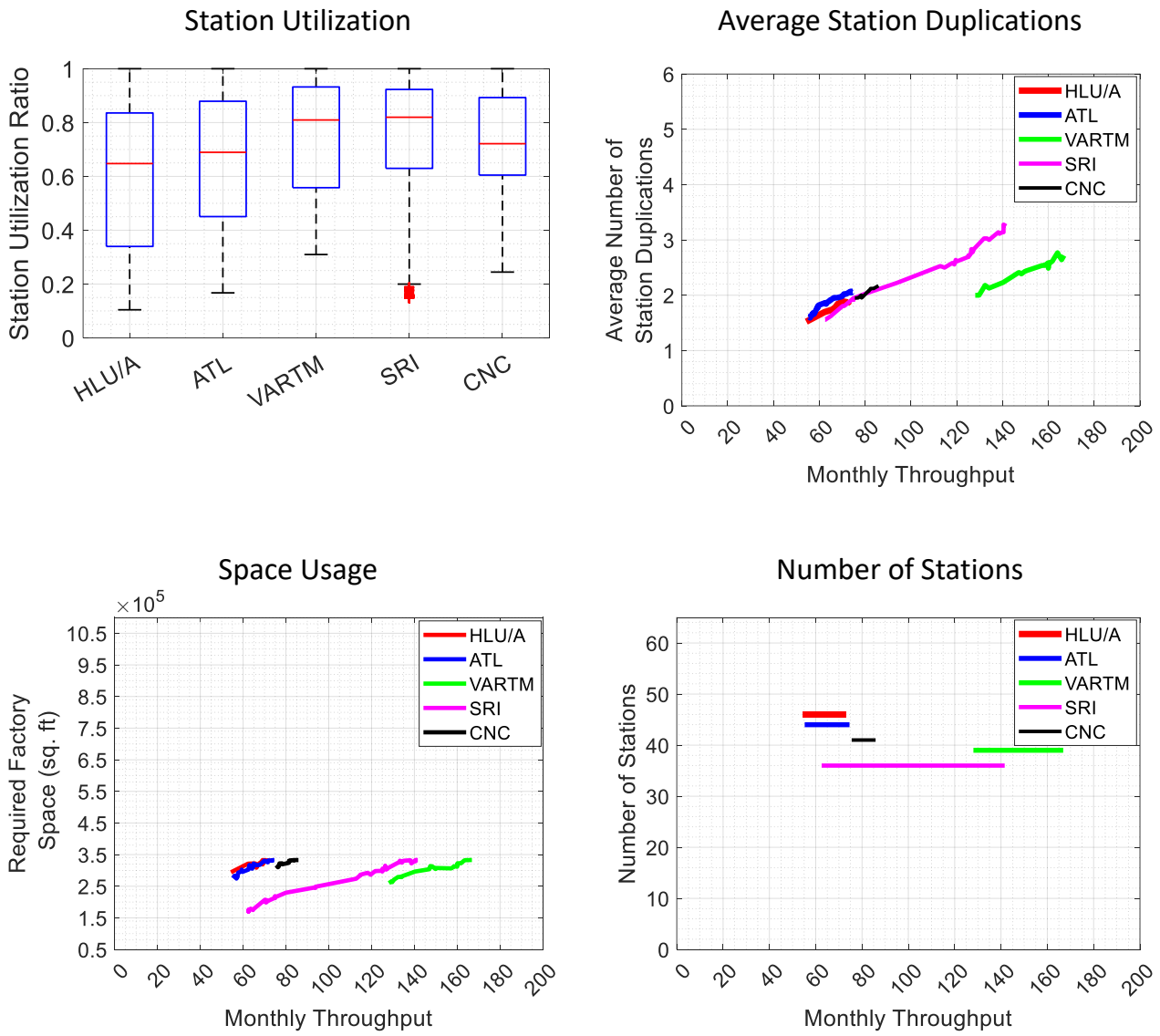


Figure 7.23: Boxplot Distribution of Station Utilization and Average Number of Station Duplicates, Number of Stations, and Space Usage as a Function of Throughput for INFRADAT Results Limited to MInDPRO's Assembly Sequences and Line Balances for All Processes

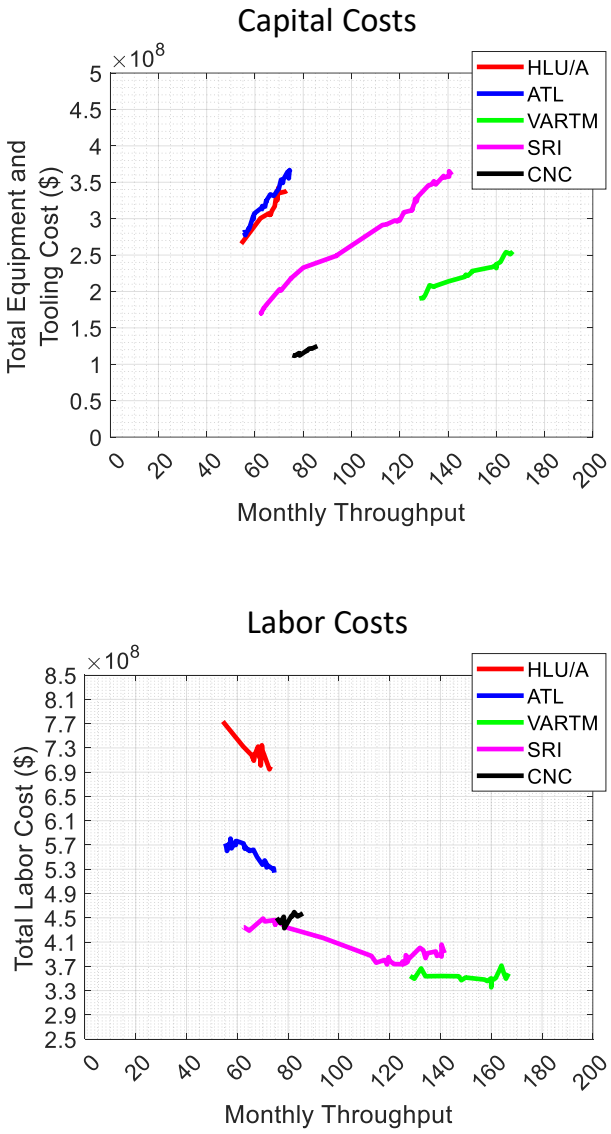


Figure 7.24: Capital and Labor Costs as a Function of Throughput for INFRADAT Results Limited to MInDPRO's Assembly Sequences and Line Balances for All Processes

Based on the average summary attainment surfaces in Figure 7.19, altering the assembly sequences to be the same as MInDPRO's most adversely affected the HLU/Autoclave process, which now costs far more and lost a third of its maximum production rate. This is because the MInDPRO HLU/Autoclave assembly sequence does not utilize any integral fabrication at all while the baseline INFRADAT one made full use of it, which speaks to the importance of integral fabrication. The VARTM process decreased its maximum production rate by about 5 wingboxes a month and the ATL process by roughly half that while the minimum-cost SRI designs now cost more as well. The MInDPRO VARTM and ATL assembly sequences had all 5 minor parts be integrally fabricated, as did nigh all the baseline INFRADAT sequences for those processes in Figure 7.13. The SRI process preferred a different set of integrally fabricated parts or subassemblies at lower throughputs than at high throughputs based on Figure 7.13 as well, revealing why its cost increased at lower throughputs. The CNC Machining processes' results remained mostly unchanged, indicating it is relatively flexible in regard to assembly sequencing since it does not use any integral fabrication.

The most noticeable changes in Figure 7.20 and Figure 7.21 tell the same story in that the HLU/Autoclave process was the most affected. Interestingly, despite its much worse throughput and costs, the station utilization ratio for the HLU/Autoclave process is still roughly the same, indicating that the ALB aspect portion of INFRADAT very effectively maximizes efficiency no matter the assembly sequence used. Similarly, the majority of the results still being far superior to MInDPRO's results' throughputs and costs demonstrate the efficacy of the INFRADAT ALB capabilities. Overall, these results show that having the proper assembly sequence by itself can impact a design's manufacturing performance and that the correct combination of integrally fabricated parts and subassemblies tremendously amplifies that. The results were only able to stay so relatively static because the number of major parts was very small at only 5, allowing a designer to possibly run into a favorable combination of integrally fabricated parts and its subsequent sequence. It only emphasizes

the importance of these factors and analyses should a more sophisticated model of the F-86 wingbox or a full commercial wingbox with far more parts be pursued and explored since, as can be seen in the case of the HLU/Autoclave process, if a non-optimal sequence or integral fabrication/subassembly combination is chosen, the manufacturing performance can seriously suffer. Additionally, without being able to analyze assembly sequences and part combinations this way, designers and subject matter experts would never even realize that the assembly sequences they chose are subpar in the first place.

Moving on to the INFRADAT results that have both the same assembly sequence and line balance as MInDPRO, the most striking differences between Figure 7.22 and the baseline results with no assembly sequence and line balance restrictions is that all the processes now have much higher minimum costs and lower maximum throughputs except for VARTM and SRI, which still have roughly the same maximum throughput. This is the case for the latter two because, as can be seen in Figure 7.23, the number of stations for those two processes is almost the same as the number of stations in the baseline results when maximum throughput is desired. In essence, SRI and VARTM prefer every task has its own station to obtain maximum production rates, which MInDPRO's line balance was able to more or less provide. However, this line balance is not optimal for cost, hence why their minimum cost values suffered. This can be seen as well when examining the station utilization ratios, which have all decreased as compared to even the results with just the limited assembly sequences. This indicates it is not just the station duplication aspect of INFRADAT that improves labor efficiency but the actual line balancing too.

Looking at the costs in Figure 7.24 shows that the maximum throughput costs for all the processes are very similar to the maximum throughput costs in the same-assembly sequence scenario while the minimum throughput costs are all much higher. Locking in the line balance to be the same as MInDPRO's thus appears to primarily set the line balances to cater towards maximum throughput while not quite being able to achieve that. Looking at Figure 7.23 and comparing with Figure 7.20 reveals the reason is that the number of

stations is not high enough for maximum throughput for the HLU/Autoclave and ATL processes while the number of duplications is not high enough to counteract bottleneck tasks for the CNC Machining process.

These INFRADAT results with the same sequence and line balance additionally demonstrate that, even with exact same line balance, they still have better throughputs and cost than their MInDPRO counterparts despite their only advantage at this point being the usage of station duplication. This again emphasizes how disadvantageous it is to have to group together different stations to duplicate due to needing to limit the number of variables being actively optimized, subsequently highlighting the weaknesses of MInDPRO needing those variables to be explicit ones instead of implicit ones. Overall, these results show the significance of the impact of ALB.

INFRADAT has shown it is able to obtain results with better production rates and costs than MInDPRO in all aspects on the same use case, the F-86 wingbox. To fully meet the first condition to prove the Overarching Hypothesis mentioned in section 7.2, the reasons why must be justifiable given the available tradeoffs and, specific to INFRADAT, traceable to the geometry and material factors that INFRADAT was developed for. For the geometry factor, the two frameworks are relatively similar in regard to aircraft design and so the geometry factor did not play a noticeable part. However, INFRADAT was able to show that it is able to flow manufacturing considerations upstream to aircraft design by favoring designs with larger taper ratios to reduce the amount of space used, which MInDPRO was only somewhat able to do with its favoring CNC Machining designs that have higher aspect ratios, a trait that INFRADAT has as well. Considering ASP as a geometry factor, INFRADAT was able to show as well that at least a small part of the reason why it performed better is due to more optimal assembly sequences. Additionally, the trends with the HLU/Autoclave process reveal that these small effects can become very large effects if the improper assembly sequence or integral fabrication part combination is used, with the latter being closely related to assembly sequencing and so both a geometry and material

factor. This highlights the fact that the importance of assembly sequencing will increase more and more the higher the level of the assembly's complexity, especially beyond the relatively simplistic F-86 wingbox examined for this Case Study.

For the material factors, INFRADAT was able to demonstrate the uttermost importance of the usage of integral fabrication and subassemblies, spatial constraints, and assembly line balancing, which encompasses the former two. MInDPRO is able to consider variable throughputs and cost as well as variable manufacturing processes, so those are not examined. The impact of the material factors, however, cannot be overstated, with proper combinations of any of them significantly improving cost and throughput. Integral fabrication and subassemblies reduce the number of available tasks proportionately more than the increase in average task times, allowing stations to be more easily line balanced and providing lower costs and higher production rates. Line balancing allows the stations to have as similar an effective station time as possible, maximizing every worker's productivity and increasing labor efficiency immensely, which also improves cost and production rates. And spatial constraints carry with it the concept of station duplication, which optimizes that labor efficiency even further. In particular, spatial constraints and the usage of targeted, automatic station duplication have more of an effect than either of the other two factors since, even when the other two factors were completely limited to be the same as MInDPRO's, station duplication utilizing spatial constraints still allowed INFRADAT's results to attain much better values than their counterparts attained from MInDPRO.

The reasons why INFRADAT was able to obtain its better results have been justifiably traced back to the geometry and material factors it was developed for as well as its usage of the ASP and ALB analyses that were designed to wrap around those factors. In this regard, INFRADAT is able to meet the first condition required for proving the Overarching Hypothesis by showing it is able to obtain better production rate and cost results than the state-of-the-art systems-thinking method that considers aircraft design. To meet the second condition and be able to fully accept the Overarching Hypothesis, it must show it is capable

of more tradeoffs as well.

7.4.2 Second Condition: More Cost and Throughput Tradeoff Capabilities

There are 4 other major sets of tradeoff capabilities INFRADAT is capable of that were not displayed in the previous section. These are: its ability to incorporate aircraft constraints; its ability to incorporate categorical aircraft design geometry variables and be able to flow them down into manufacturing analysis; its ability to further expand on integral fabrication and subassemblies; and its ability to alter its spatial constraints. All of these abilities and capabilities are interchangeable with the tradeoff capabilities displayed in the previous section and so only the most pertinent ones will be presented alongside the additional capabilities when they are demonstrated. On the other hand, the majority of MInDPRO's capabilities as they specifically relate to improving cost and production rates, and as presented by Siedlak et al. in their works [62, 29], were already displayed in the previous section. The only other major capability it has specifically regarding throughput and cost improvements that has not been demonstrated yet is one it also shares with INFRADAT, which is that it is able to account for aircraft constraints as well.

The first capability to be covered is in regard to aircraft performance as well as the incorporation of aircraft performance constraints. The baseline INFRADAT and MInDPRO results from the previous section only examined the manufacturing performance in terms of production rate and cost and selected aircraft design variables with the sole aim of improving those values. The resulting aircraft's performance was not considered. However, both INFRADAT and MInDPRO are indeed able to examine them. The performance characteristics of the aircraft resulting from the usage of the previous section's aircraft design variables purely to improve throughput and cost are shown in Figure 7.25, Figure 7.26, and Figure 7.27. They depict the most pertinent traits for the F-86 fighter jet, which are its Takeoff Gross Weight (TOGW), landing approach velocity V_{app}, Takeoff Field Length (TOFL), Landing Field Length (LFL), radius of turn, and climb rate. The baseline F-86's

TOGW is about 18,000 lbs., its V_{app} about 130 knots, its TOFL is 3,770 feet, its LFL is 3,450 feet, its climb rate at its combat altitude of 35,000 feet is approximately 1,500 feet per second, and its radius of turn at its combat altitude of 35,000 feet while flying at Mach 0.8 is about 15,000 feet [225].

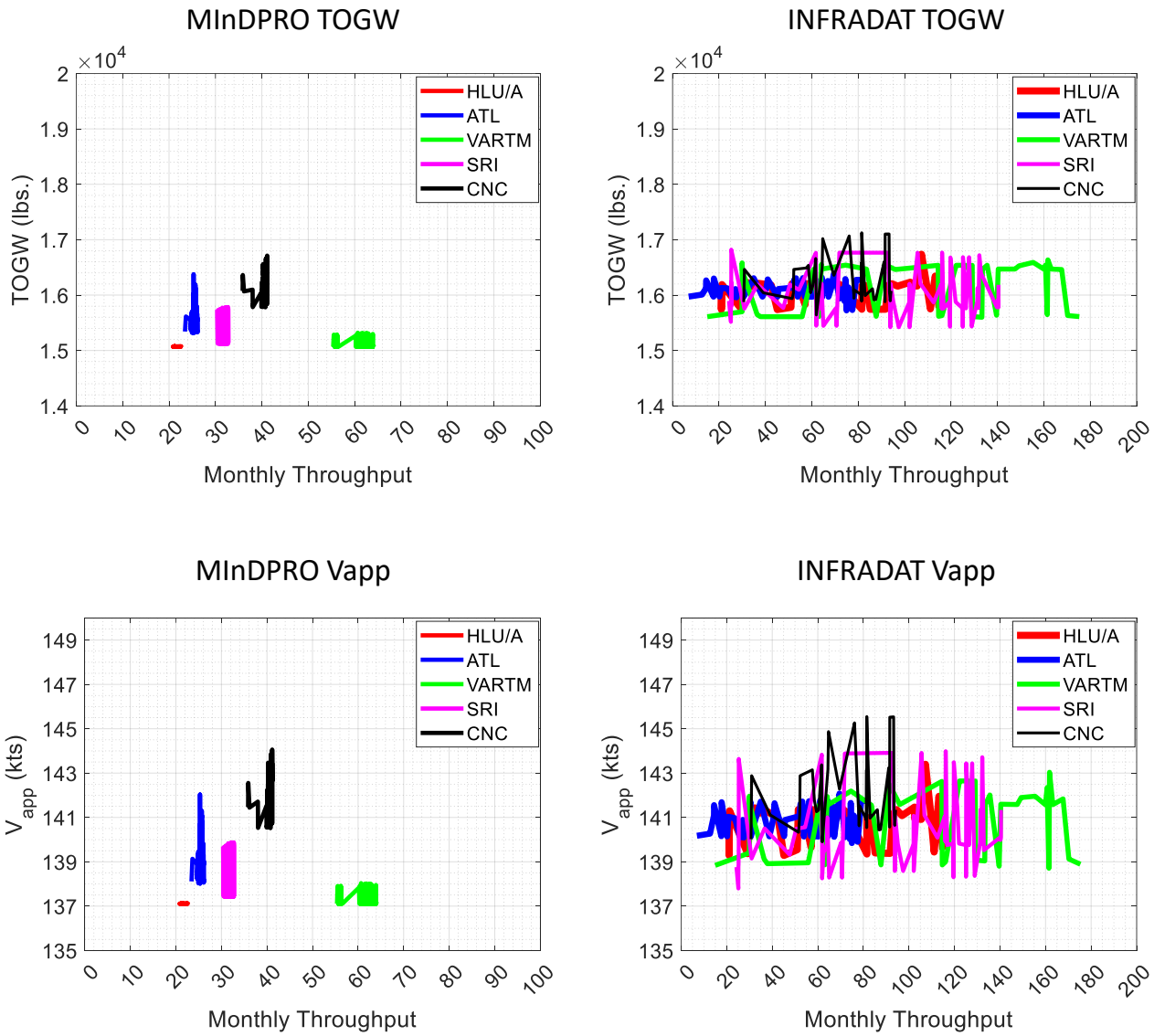


Figure 7.25: Takeoff Gross Weight (lbs.) and Landing Approach Velocity (kts) as a Function of Throughput for MInDPRO and INFRADAT Results for All Processes

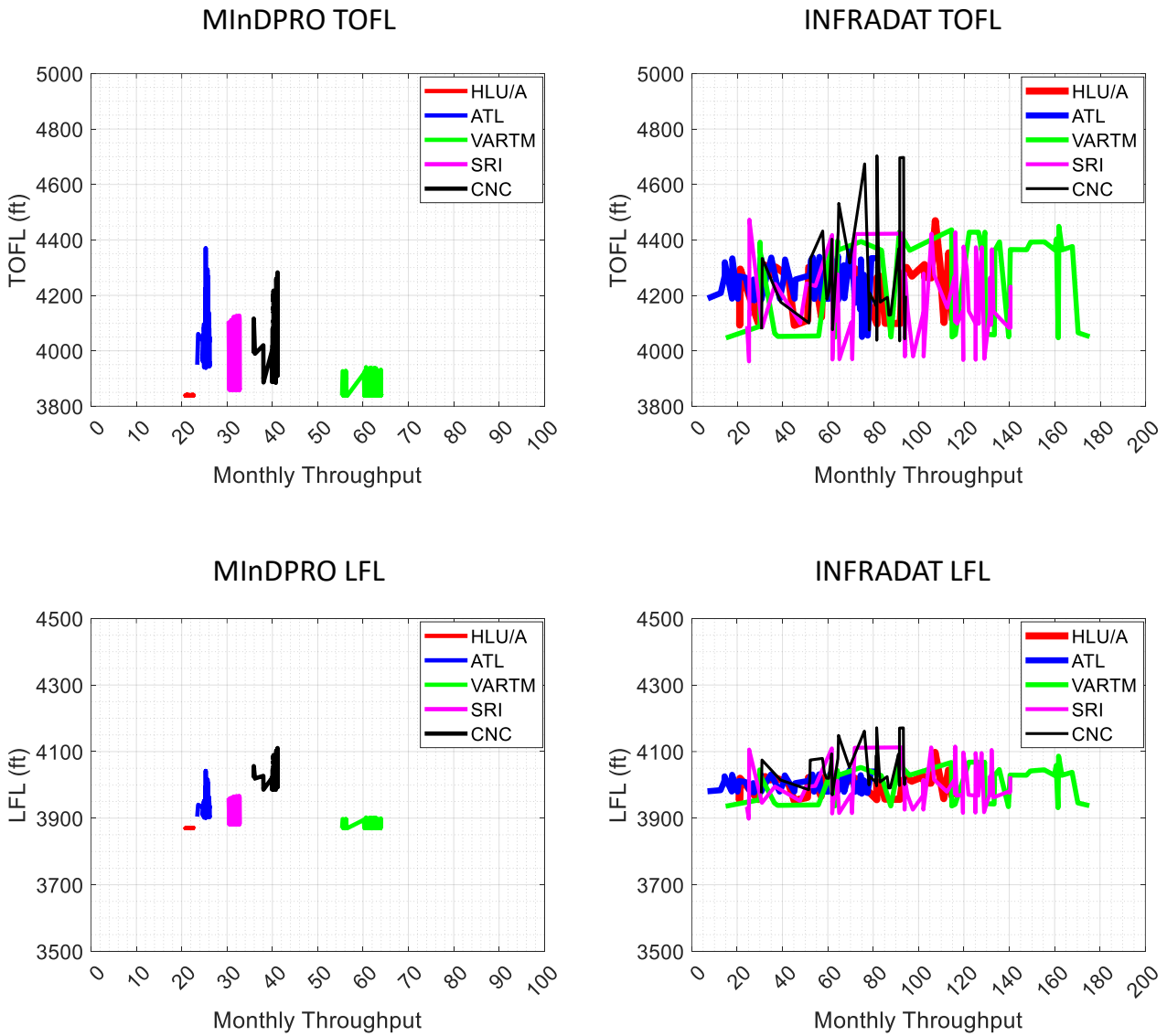


Figure 7.26: Takeoff Field Length (ft) and Landing Field Length (ft) as a Function of Throughput for MInDPRO and INFRADAT Results for All Processes

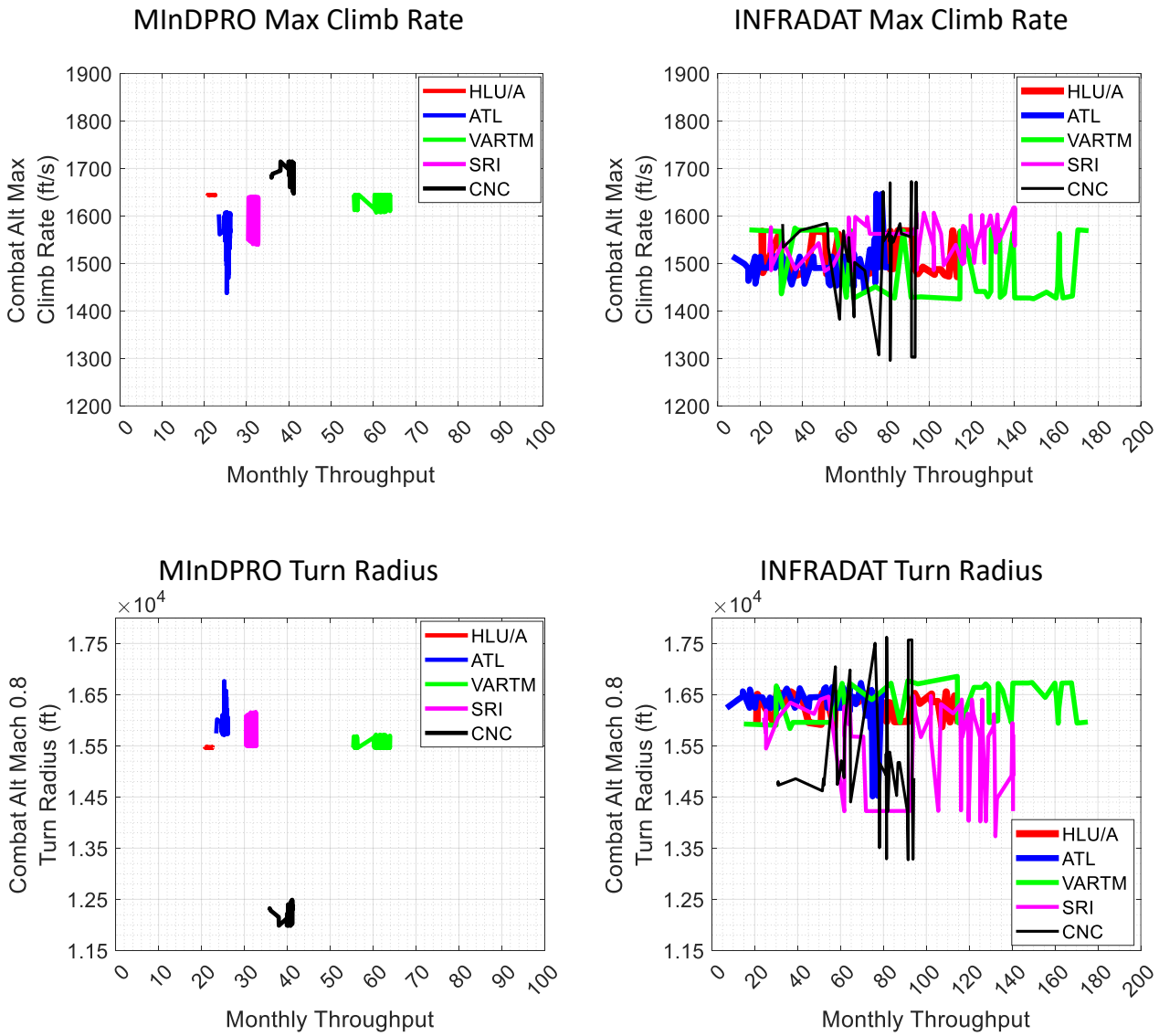


Figure 7.27: Max Climb Rate (ft/s) and Radius of Turn (ft) at Mach 0.8 and Combat Altitude as a Function of Throughput for MInDPRO and INFRADAT Results for All Processes

The TOGW values for both frameworks for all the manufacturing processes in Figure 7.25 show that the sized aircraft is much lighter than the baseline one. This is primarily due to the fact that the wing area for all the designs in both frameworks are also significantly smaller than the baseline F-86's. The MInDPRO aircraft all predictably have lower TOGW values due to their lower wing area and taper ratio design variables discussed in the previous section. The smaller wing areas of both frameworks' results subsequently made their landing approach velocity and their TOFL and LFL values in Figure 7.26 suffer as well, with all of them being worse than the baseline F-86's. The overall lower weights made aircraft designed with both frameworks have a higher maximum climb rate than the baseline aircraft in Figure 7.27 while their radius of turn fared slightly worse.

The MInDPRO aircraft generally outperform the INFRADAT ones in terms of aircraft performance characteristics due to INFRADAT prioritizing the manufacturing discipline outputs with no regard for aircraft performance, as discussed in the previous section. This is from a combination of the wing areas not being absolutely minimized as well as a preference for higher taper ratios, with the latter in particular adversely affecting the aircraft's aerodynamic characteristics and thus its performance as well. The composite aircraft generally had superior performance to the metallic aircraft due to being made from a lighter material in the form of carbon fiber, lowering their TOGW, landing approach velocity, TOFL, and LFL. The more combat-oriented characteristics in Figure 7.27 showed the opposite, where the metallic aircraft tended to have better values for climb rates and turn radius. This is a byproduct of a trend identified in the previous section where metallic wingboxes preferred higher aspect ratios to reduce rib machining costs. The higher aspect ratios caused the wings to be more aerodynamically efficient, noticeably improving their performance over their composite counterparts in terms of climb rate and turn radius. This is especially true with the MInDPRO metallic wings which favored a maximized aspect ratio and so has amazing turning performance compared to everything else. It represents an unlikely occurrence in which manufacturing considerations actually improved the aircraft

design's performance.

Another identifiable trend is that the performance characteristics stayed relatively constant given changes in production rate, indicating that a design conducive to low costs is also conducive to high throughputs. This means that, at least for the F-86 use case, a single design is likely to be sufficient for any manufacturing setup or changes to that setup.

Overall, incorporating aircraft design with the manufacturing discipline has a significant impact on production costs and rates. This can be seen by comparing the INFRADAT baseline results in Figure 7.7 with those in Figure 5.11 in Experiment 1 where aircraft design was not included at all. Tailoring the design to meet the manufacturing needs greatly improves the manufacturing performance, able to improve the maximum production rate among all processes from 100 to 160 wingboxes per month and decreasing the minimum APUC from $300,000 to roughly $240,000, a 60% and roughly 25% improvement each. However, the aircraft design discipline has performance requirements as well to carry out their missions and optimizing the aircraft design just for manufacturing performance is likely to degrade its mission performance, which can be seen by the general decrease in performance discussed above. To address this both frameworks have the ability to impose performance requirements on the aircraft design, which can be used to examine what the resulting manufacturing design space looks like. Figure 7.28 shows the average summary attainment surfaces for the baseline aircraft with no constraints of any kind as well as aircraft with combat constraints and another with takeoff/landing constraints, which were all described in Table 7.8. Figure 7.29, Figure 7.30, Figure 7.31, and Figure 7.32 show how the aircraft design input variables are distributed given those constraints to see what changes resulted from them.

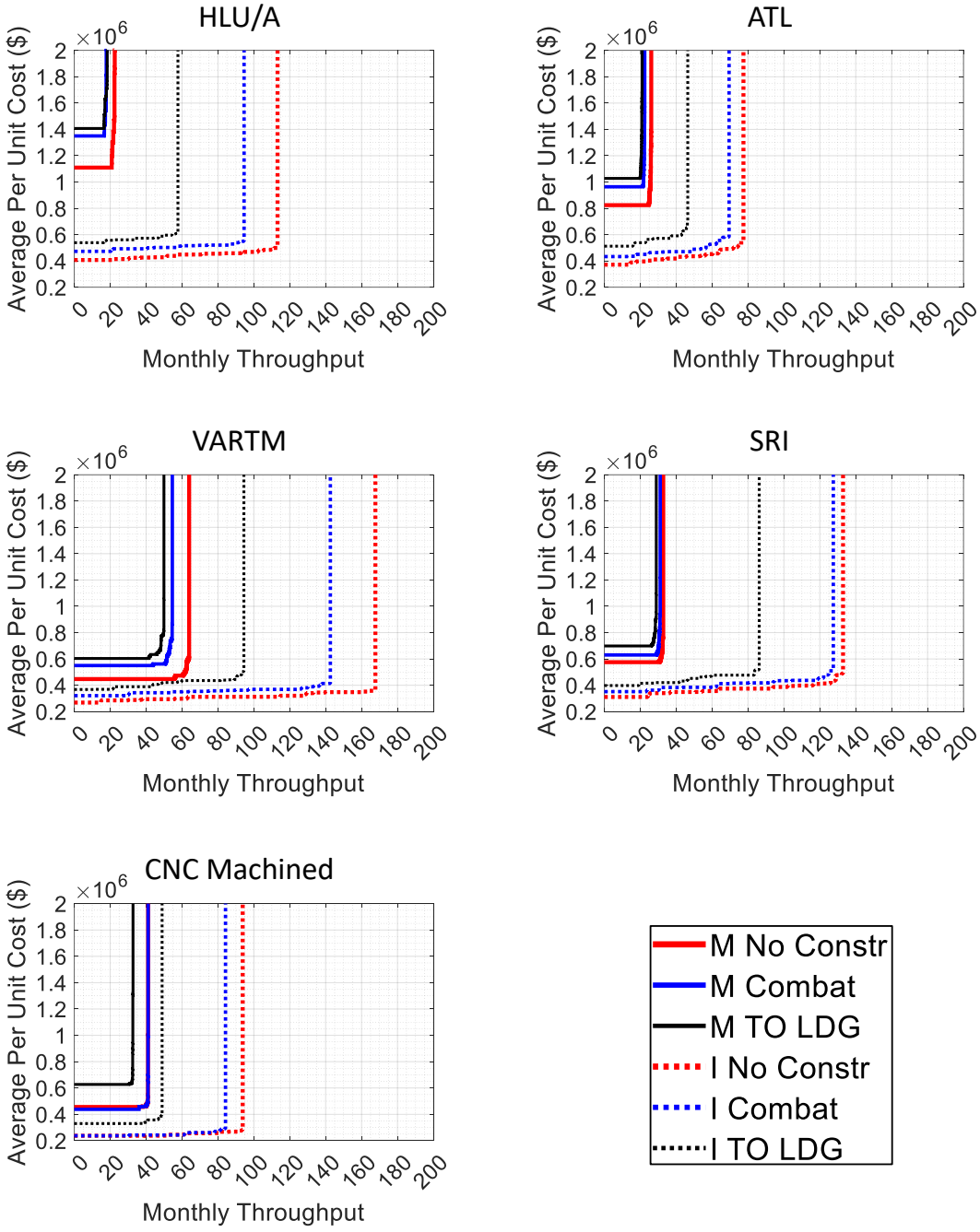


Figure 7.28: Average Summary Attainment Surfaces Comparing INFRADAT (I) and MInDPRO (M) Results for All Processes for Baseline No Aircraft Constraints, Aircraft Combat Constraints, and Aircraft Takeoff and Landing Handling Constraints

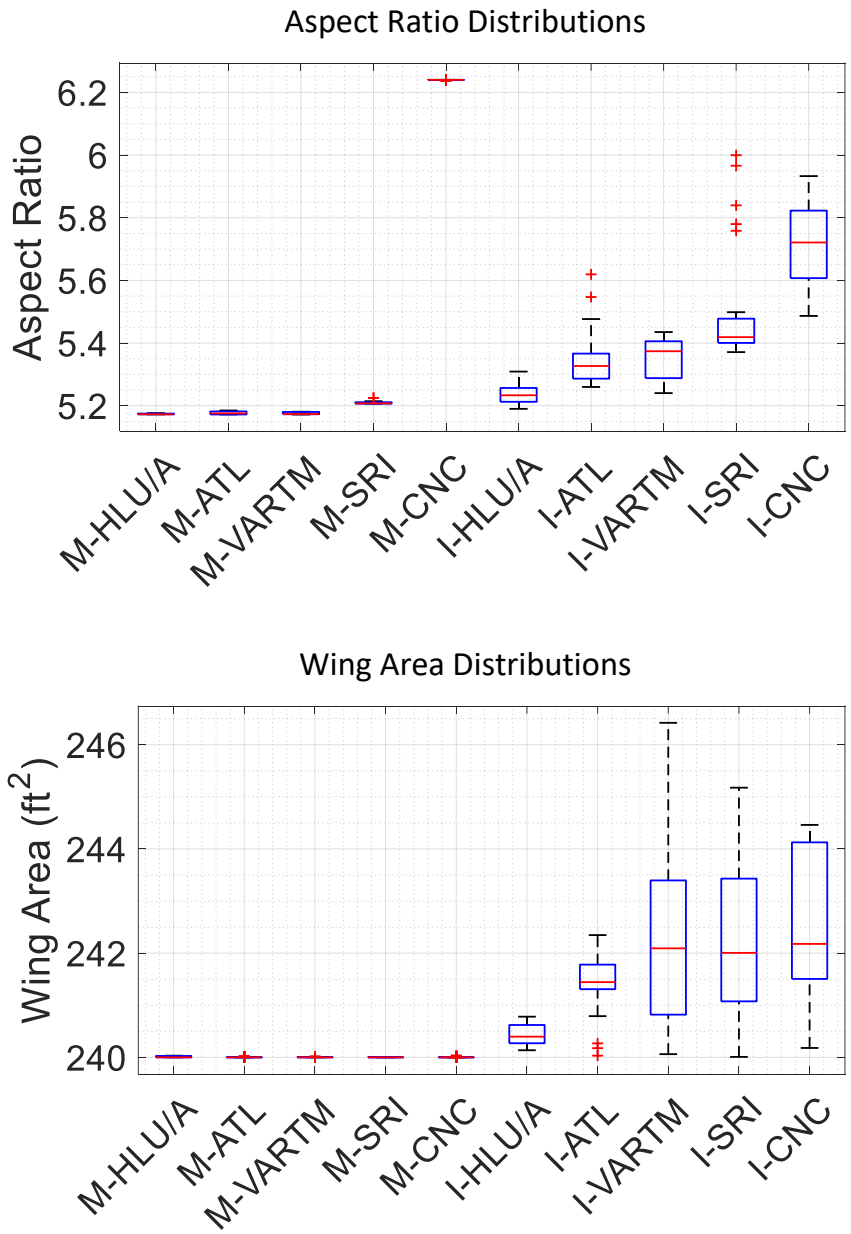


Figure 7.29: Boxplot Distribution Comparison of Wing Area (ft^2) and Aspect Ratio for MInDPRO and INFRADAT Results for All Processes For Aircraft Combat Constraints

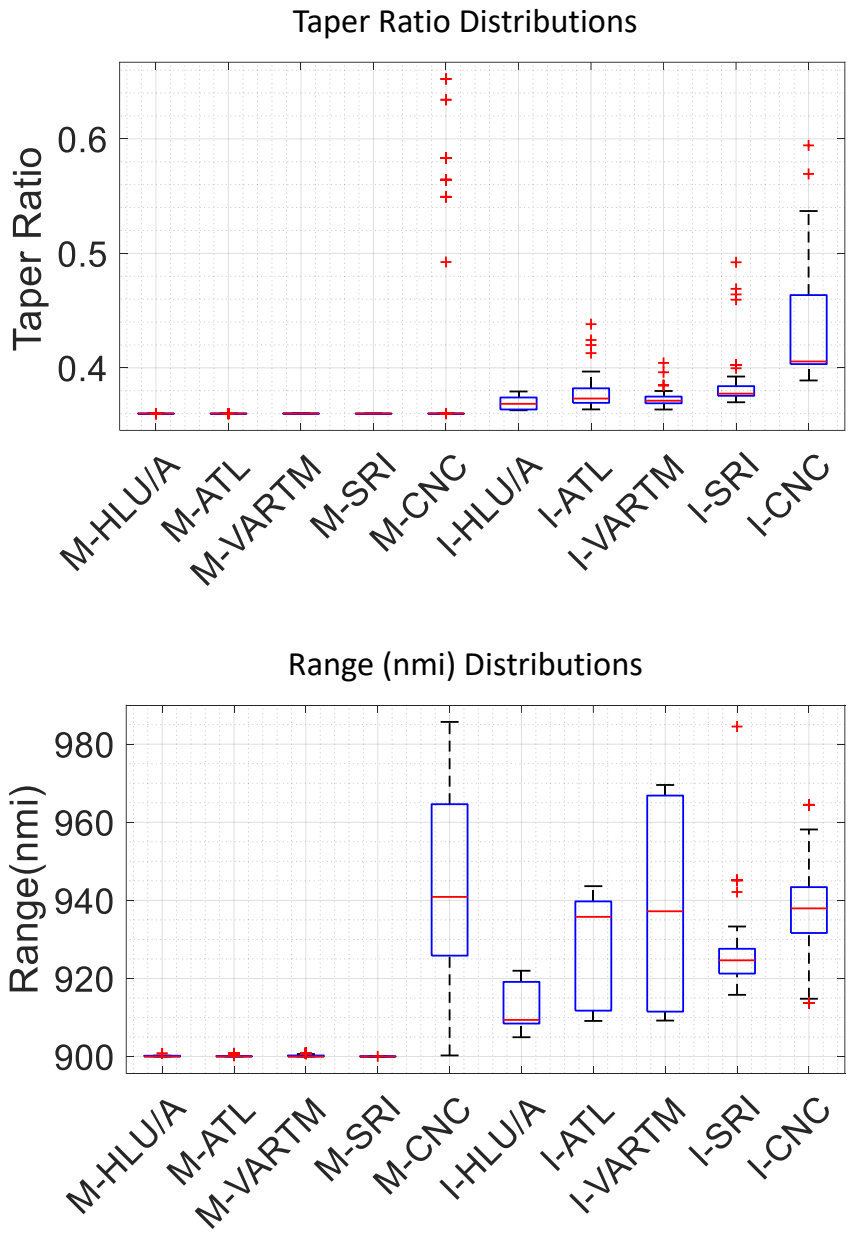


Figure 7.30: Boxplot Distribution Comparison of Wing Taper Ratio and Range (nmi) for MInDPRO and INFRADAT Results for All Processes for Aircraft Combat Constraints

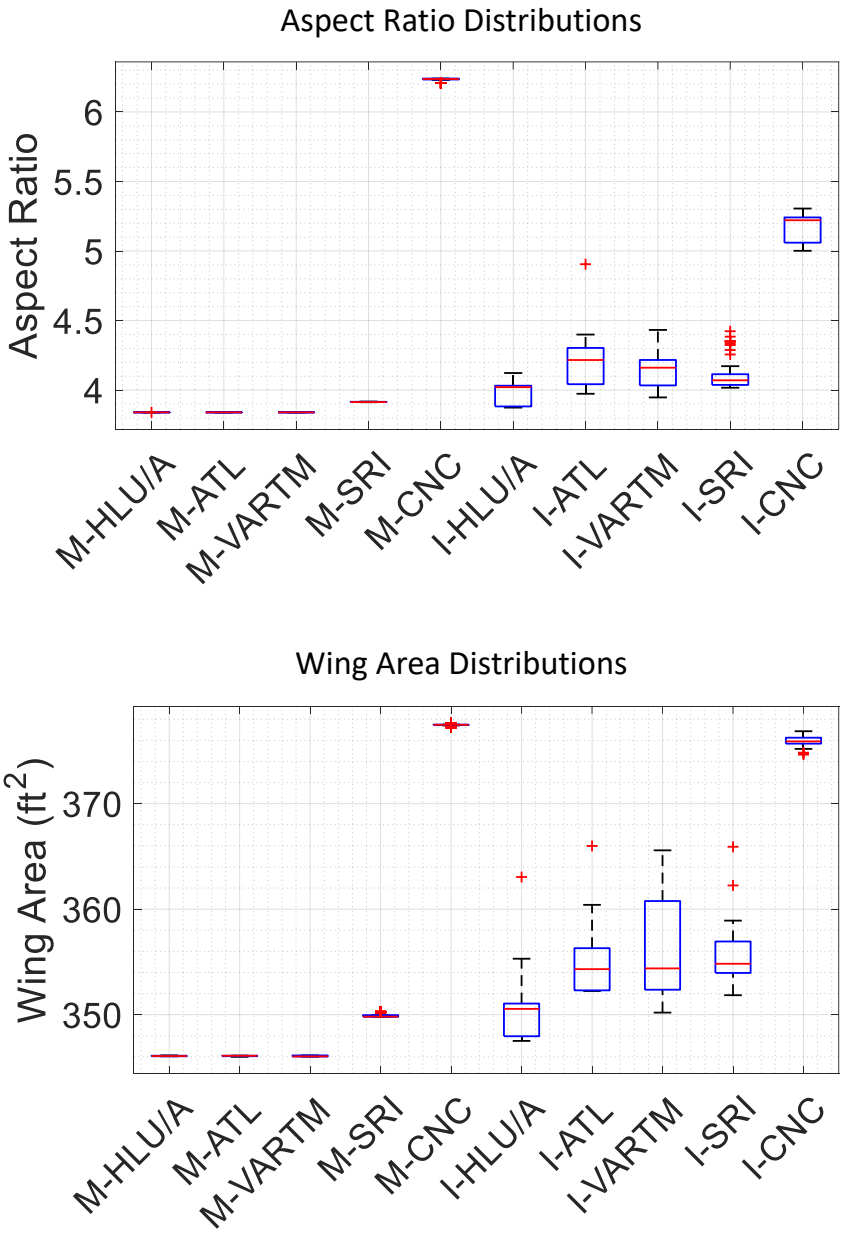


Figure 7.31: Boxplot Distribution Comparison of Wing Area (ft^2) and Aspect Ratio for MInDPRO and INFRADAT Results for All Processes For Aircraft Takeoff and Landing Handling Constraints

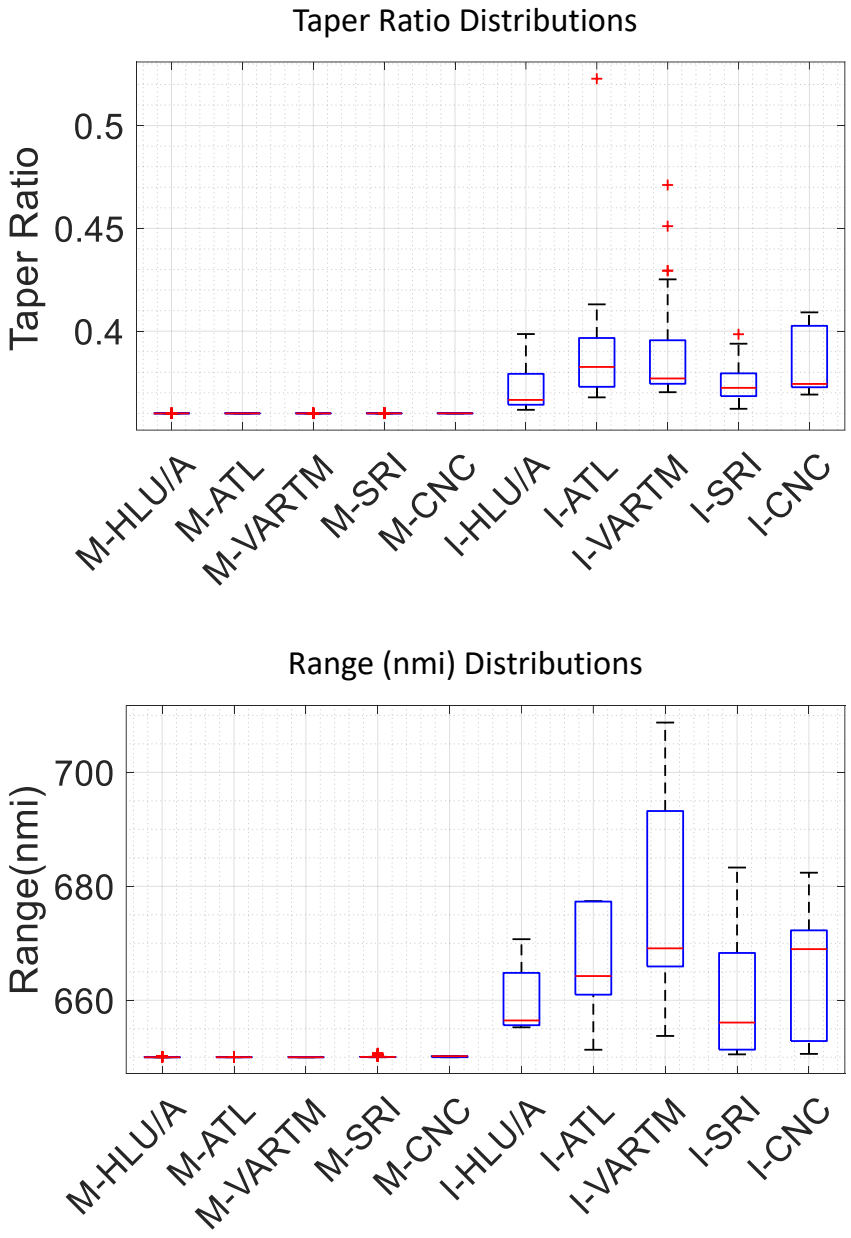


Figure 7.32: Boxplot Distribution Comparison of Wing Taper Ratio and Range (nmi) for MInDPRO and INFRADAT Results for All Processes for Aircraft Takeoff and Landing Handling Constraints

The manufacturing performance of the designs with performance constraints, as expected, decreased. However, what is unexpected is the magnitude of their decrease. With the combat constraints, all the composite designs suffered a throughput and cost penalty in Figure 7.28 due to their aircraft being forced to have higher aspect ratios in Figure 7.29 to be able to improve their aerodynamic characteristics and meet the turn radius requirements. Meanwhile, the metallic designs were almost unaffected since they preferred larger aspect ratios anyways. While there was a hit to maximum production rates and minimum costs, the designs with combat constraints actually still fared better in the manufacturing sense than the baseline F-86 in Figure 5.11 in Experiment 1. They can thus be manufactured more easily, cheaply, and quickly while having better combat performance characteristics to dogfight more effectively on top of that, though they do suffer in takeoff and landing capabilities with TOFL and LFL values that are, on average, 500 feet greater.

Incorporating the takeoff and landing constraints while keeping roughly baseline combat capabilities, however, is a different story. The wing area had to be significantly increased in Figure 7.31 to be able to meet the landing approach velocity, TOFL, and LFL requirements. This led to the wing and wingbox being much larger and less optimized for manufacturing, causing the maximum production rates to be almost halved while incurring large cost penalties as well. Additionally, the severity of the constraints caused the range variable, which normally does not play a significant role, to become one of the limiting variables, with there being no design space for ranges above roughly 710 nautical miles in Figure 7.32. This is because while the range variable's effects on weight are small and seemingly imperceptible at times, they still exist, with greater ranges requiring slightly greater aircraft weights. These greater weights negatively impact the climb rate and turn radius constraints, causing there to be no design space left if the baseline 804 nautical mile range were used. However, as can be seen from Figure 7.32, the ranges for the composite aircraft can be higher than the metallic ones due to their lower weight based on one of the composite processes having a wider distribution of ranges than either framework's metallic

process designs.

There is thus a push and pull interaction between aircraft design and manufacturing design where their needs can conflict. A satisfactory midpoint must be found that is acceptable to both disciplines, and both MInDPRO and INFRADAT have the capability to perform tradeoffs that visualize those interactions, as demonstrated by the results just shown. This is therefore a tradeoff capability where MInDPRO is just as capable as INFRADAT and is a significant one in terms of being able to optimize manufacturing design while considering aircraft design.

A capability related to aircraft performance that MInDPRO does not possess is the ability to examine categorical aircraft design variables and assess their impacts on both aircraft design and manufacturing design. These categorical variables are primarily related to the shape of the parts and affect the structural performance of the assembly on the aircraft design side and the assembly's accessibility on the manufacturing side. MInDPRO has been shown to be able to consider alternative shape concepts such as different rib shapes but has primarily tied that to different manufacturing processes; it has not been able to show it can vary the shape of the part within the same manufacturing process and analyze its structural and manufacturing implications.

INFRADAT is able to do so through the combination of Conjectures 1.1 and 1.2 from chapter 4. MInDPRO is able to utilize Conjecture 1.1 as well, but Conjecture 1.2 is unique to INFRADAT and allows it to examine the effects of different part shapes on manufacturing beyond just how large they are or how much material they use. It brings assembly sequencing into the picture due to different shapes having different feasible sequences and similarly includes accessibility in the discussion, neither of which are capabilities MInDPRO is able to replicate. To showcase this capability, for this first implementation of INFRADAT for the F-86 use case, the spars are the primary part examined with their shapes being the geometric detail altered. They are chosen because the spars are essentially the backbone of the wingbox and the entire wing and so any changes to them can have large ef-

fects downstream on the wingbox's structural integrity and, through that, the performance of the aircraft as a whole. Changing the spars' shape can subsequently produce noticeable effects downstream due to the different spar cap configurations changing how loads are distributed. On the manufacturing side, the spars' shapes affect which sequences are feasible and also plays a large role in how accessible the assembly is, with some shapes being more likely to cause accessibility issues due to obstructing joining locations.

To analyze the effects of different spar shapes, 5 different ones are used on the baseline F-86 wingbox with no other constraints or limitations and the resulting wingboxes optimized via INFRADAT. The manufacturing results in the form of average summary attainment surfaces are shown in Figure 7.33. The subsequent aircraft design performance characteristics from using different spar shapes are shown in Figure 7.34, Figure 7.35, and Figure 7.36. Only the TOGW, TOFL, and combat altitude climb rate are examined because they showed the most changes as a function of spar shape. Since it was pointed out that the performance characteristics tended to not change as a function of throughput, boxplot distributions are used instead. The baseline shape for reference is the Out-C spar shape.

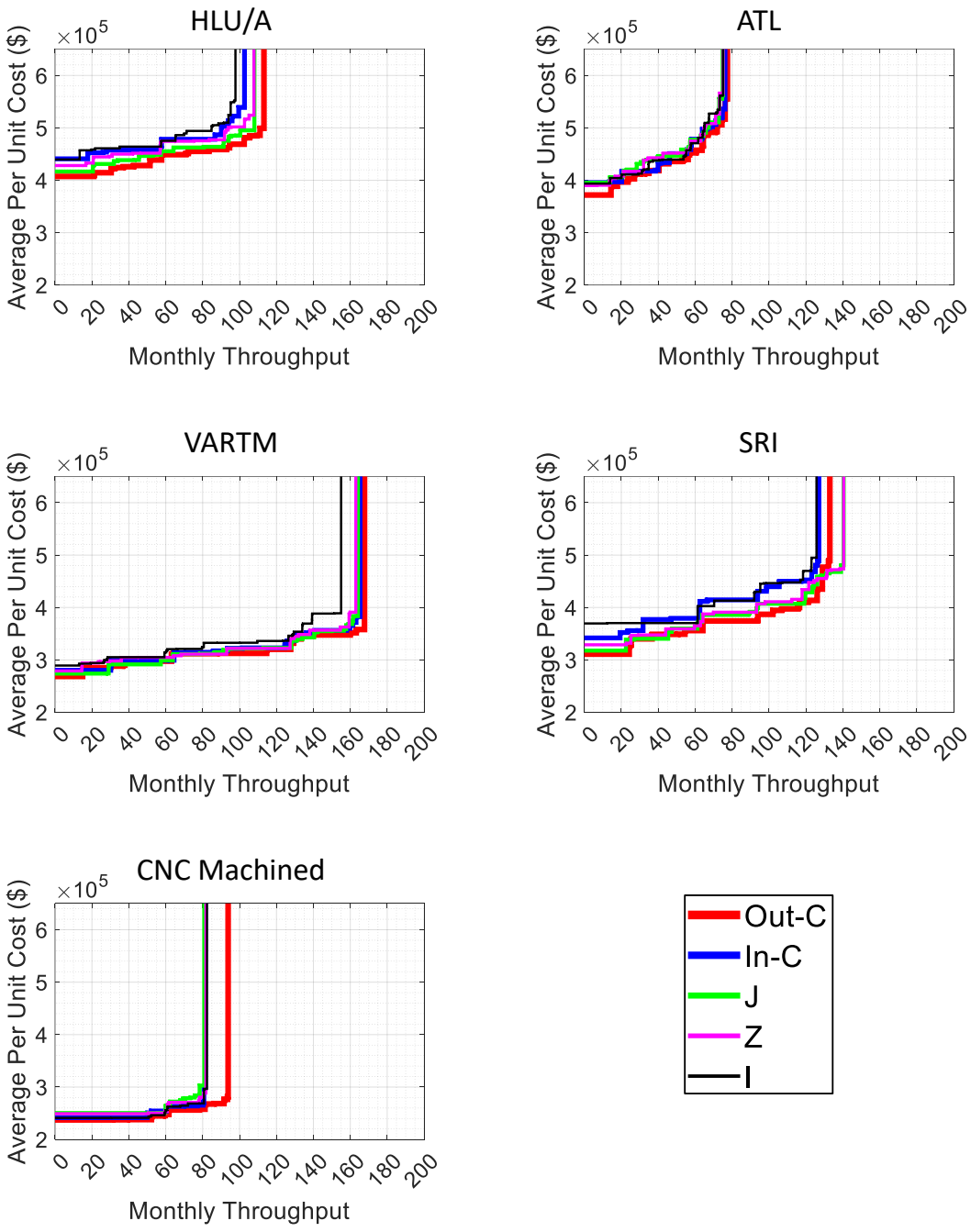


Figure 7.33: Average Summary Attainment Surfaces Comparing INFRADAT Results for All Processes for All Different Spar Shapes

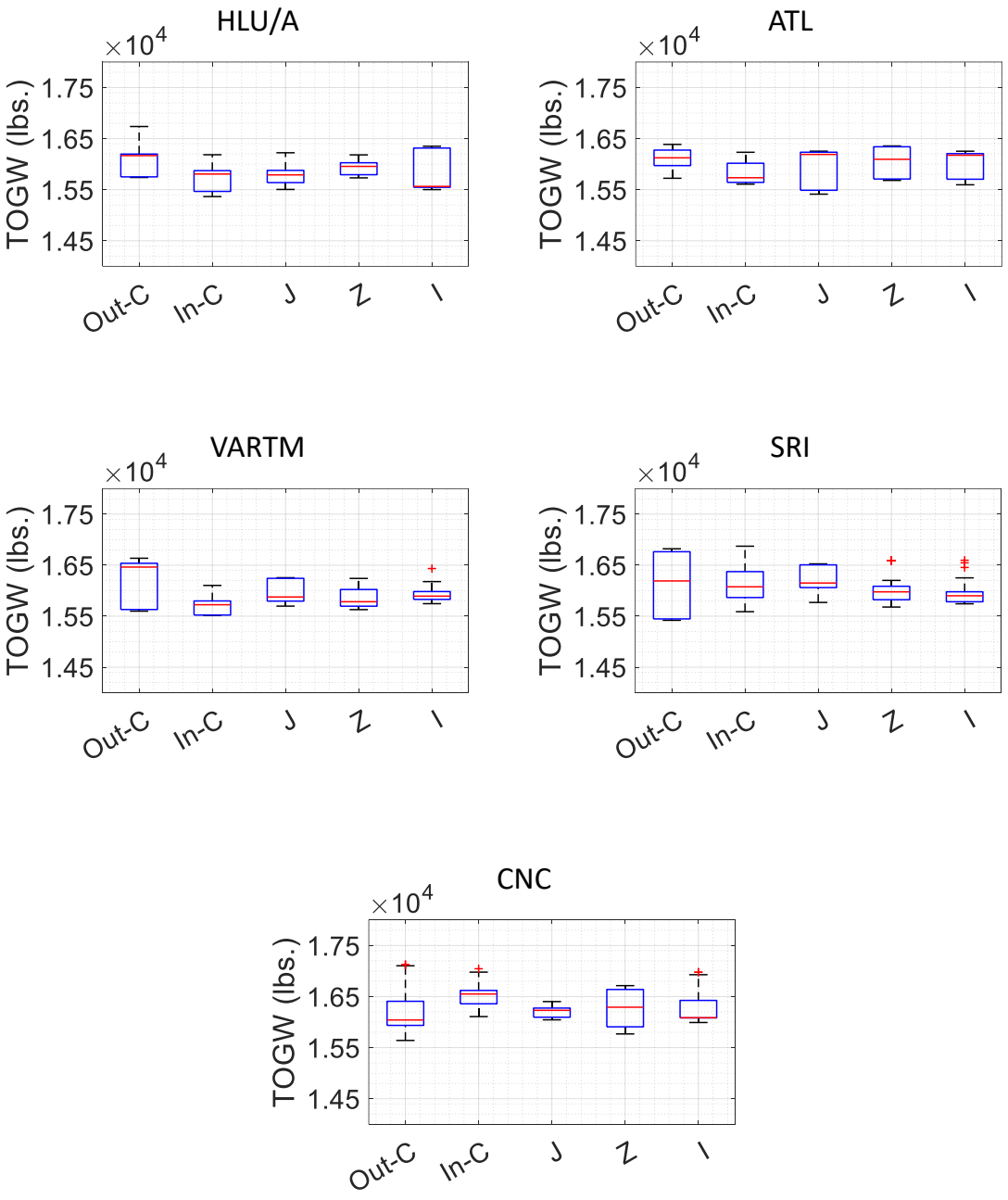


Figure 7.34: Boxplot Distribution of Takeoff Gross Weight (lbs.) as a Function of Spar Shape for MInDPRO Results for All Processes

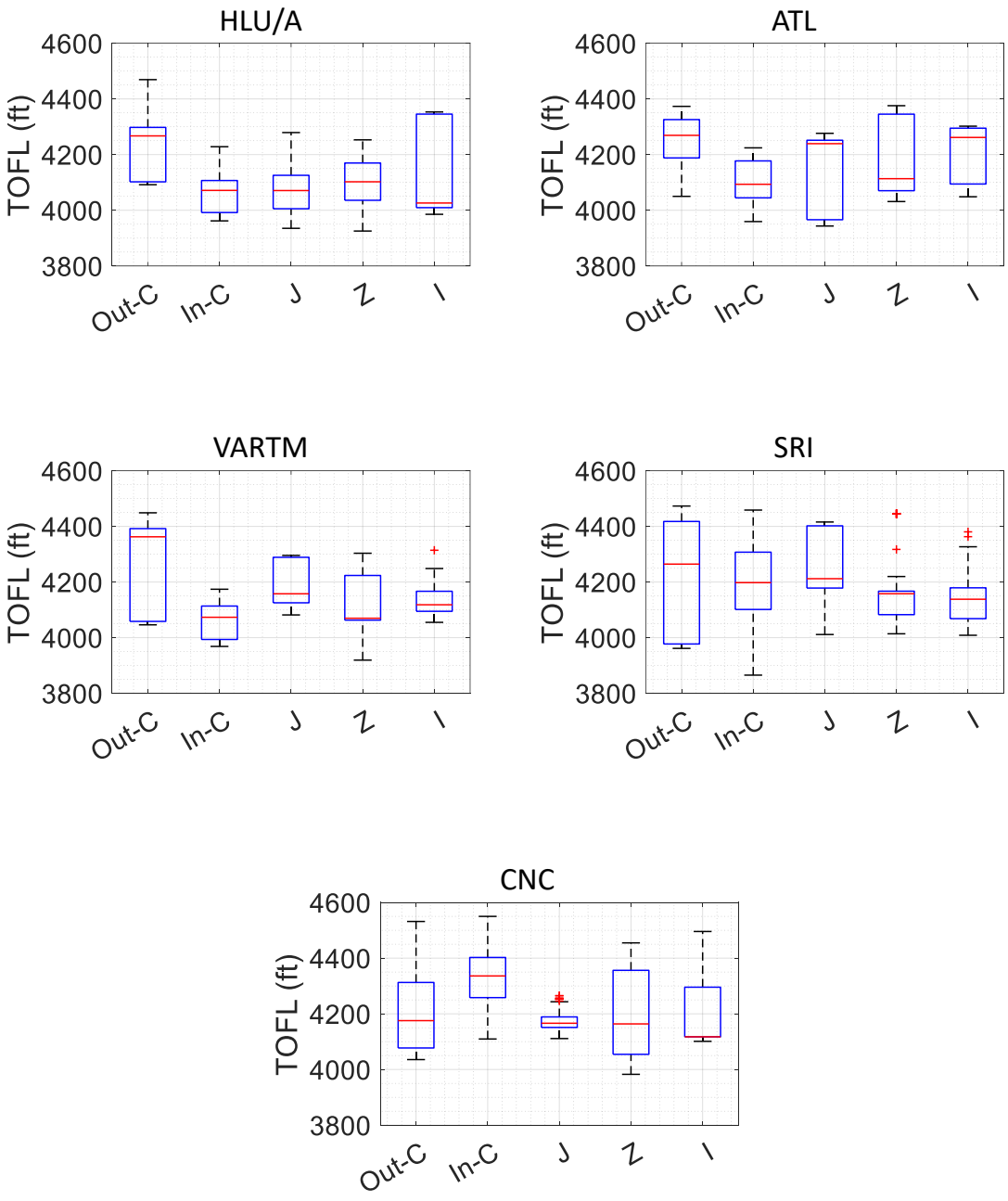


Figure 7.35: Boxplot Distribution of Takeoff Field Length (ft) as a Function of Spar Shape for MInDPRO Results for All Processes

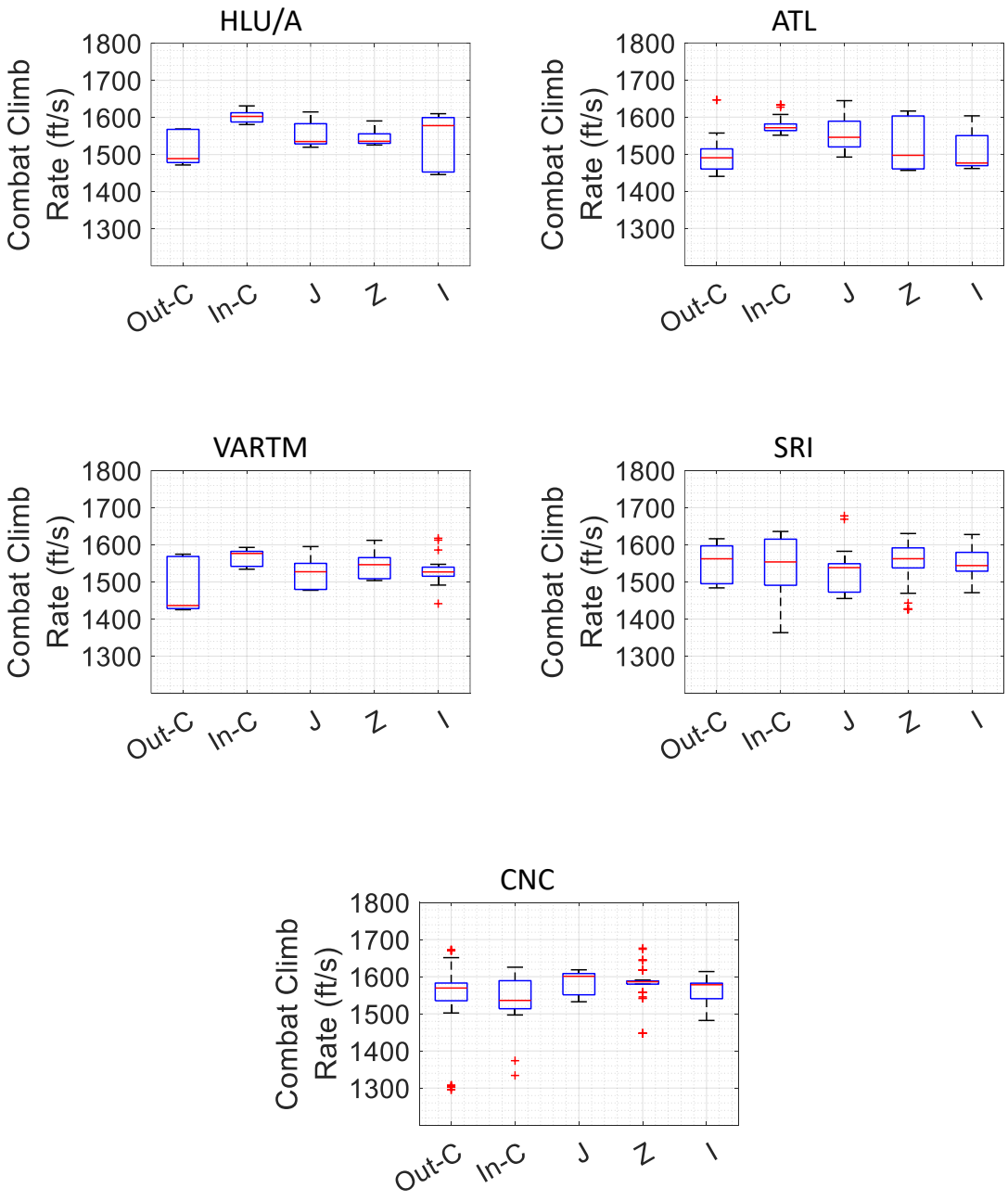


Figure 7.36: Boxplot Distribution of Combat Altitude Max Climb Rate (ft/s) as a Function of Spar Shape for MInDPRO Results for All Processes

From Figure 7.33 it is clear that spar shapes can have a large impact on the resultant production rates and costs, with the HLU/Autoclave process demonstrating this most clearly by having the optimal shape have a 20 wingboxes-per-month higher production rate and an APUC $50,000 lower than the least optimal shape. The other manufacturing processes are affected to varying degrees. What is notable is that the default Out-C shape generally performs the best in terms of cost and throughput. This is likely because the Out-C shape is able to yield the most feasible sequences by never obstructing the insertion of the ribs while all the others are liable to do so given specific sequences. The I and In-C spars tend to perform the worst because they yield the lowest number of feasible sequences, with the ribs always needing to be joined with both spars before any of them can be attached to either wingskin part. On top of this, their flanges extend into the interior of the wingbox, which causes additional accessibility issues because a worker will always need access holes to attach the IML portions of the flanges to the wingskin that closes out the wingbox. The Z and J spars are intermediate shapes that have less accessibility issues and more feasible sequences than the I and In-C spars but do not perform quite as well as the Out-C spars.

Looking at Figure 7.34 it is clear that these assembly sequencing and accessibility benefits are almost solely the reason why the Out-C shape is preferred in terms of production cost and rate. The Out-C shape is generally not the most structurally sound because it is typically not the spar shape that yields the lowest weight. In fact, the Out-C shape tends to be on the higher end in terms of how much it causes its wingboxes to weigh. This means it is preferred in Figure 7.33 not because of having lower weight and requiring less material, but because of its other benefits despite generally higher weights and requiring more materials. Examining Figure 7.35 and Figure 7.36 confirms this as, for the composite processes, the Out-C spars generally have one of the worse, if not the worst, TOFL and turning radius values. This all means the Out-C spars are not particularly structurally sound but are preferred for the ease of use during assembly, a trend MInDPRO would not have been able to pick up on because it is only able to analyze the impact of part geometries on

manufacturing via their size, not their handling.

The SRI process stands out among all this by not having the Out-C spars be the best shape, at least for its designs that yield its maximum production rates. This is because, due to their usage of stitching, their spar flanges need to be wider for enough space to attach the stitches. This improves the structural efficiency of the J and Z spars and reduces their weight enough in Figure 7.34 that, combined with their middling assembly handling penalties compared to the I and In-C shapes, they are able to outperform the Out-C spars when maximum throughput is needed. This is all despite their assembly sequencing and accessibility drawbacks compared to the Out-C shape. For the CNC Machining process, the Out-C and In-C spars are the simplest shapes to mill out, but due to the accessibility and number of feasible sequences drawbacks with the latter shape and due to the former also weighing less in Figure 7.34, the Out-C shape is the superior one by a large margin.

This all emphasizes the importance of the structural shapes as they can have impacts on both the aircraft's performance characteristics and on its manufacturing outputs. Even more so, it demonstrates the importance of assembly sequencing and how it interacts with differently-shaped parts in aircraft design through Conjecture 1.2. The ideal spar shape in the absence of any aircraft constraints and focusing only on throughput and cost turned out to not be the spar shape that was the most structurally sound and so caused the whole assembly to weigh the least. Instead, it was the spar shape that optimized its assembly sequence's handling capabilities that itself resulted in more ideal assembly sequences being chosen. As mentioned before, MInDPRO would not have been able to capture this tradeoff since it does not look at assembly handling, or manufacturing handling in general, at all. INFRADAT is therefore able to better capture reality by accounting for how differently shaped parts are handled for assembly while still including its aircraft performance traits. Even if it did turn out that the spar shape, or any part's shape, did not significantly affect the aircraft or manufacturing performance, it is better to have the capability to check it anyways as opposed to assuming a shape is optimal and not knowing whether it actually is,

as must be done in MInDPRO since it lacks this capability.

This incorporating of considerations for assembly handling based on the shape of the parts, which affects both aircraft and manufacturing characteristics, is an important tradeoff because it can be greatly expended on to further connect the two normally separate disciplines. It can be augmented with higher fidelity structural analysis than RADE is currently able to handle to further differentiate the spar shapes. The accessibility analysis can also be expanded beyond just the spar's shape, thereby enabling the design of other part's shapes to be included too. And should a more detailed accessibility analysis technique efficient enough to be used in early preliminary aircraft design be found, accessibility penalties can be applied to all assembly tasks instead of just the wingbox-closeout ones, increasing how much reality can be captured through this modeling. This last improvement would improve the ergonomics of working on wingbox assemblies too, decreasing other secondary health-related costs not covered in this work. This tradeoff capability is basically a gateway to further, more influential tradeoff analyses that can be implemented in the future given more detailed analysis capabilities and is one of INFRADAT's most distinguishing features. It ultimately establishes more feedback loops between the aircraft and assembly disciplines and allows them to communicate better to benefit both.

Another tradeoff capability INFRADAT has is related to integral fabrication. INFRADAT is not only able to determine which parts should or should not be integrally fabricated or in a subassembly, but can change their bounds as well. In Experiment 1 this was done by forcing there to be no integrally fabricated parts or subassemblies. INFRADAT can do the reverse too and see whether, given enough technological advancements in the future, more major parts able to be integrally fabricated or in a subassembly actually provides any advantages. This is simulated by allowing 4 major parts to be integrally fabricated or in a subassembly and adjusting the assembly sequence generation to produce such sequences more often and then comparing the results with the baseline where only 3 major parts can be in a combination at a time. The results are shown in Figure 7.37 and Figure 7.38.

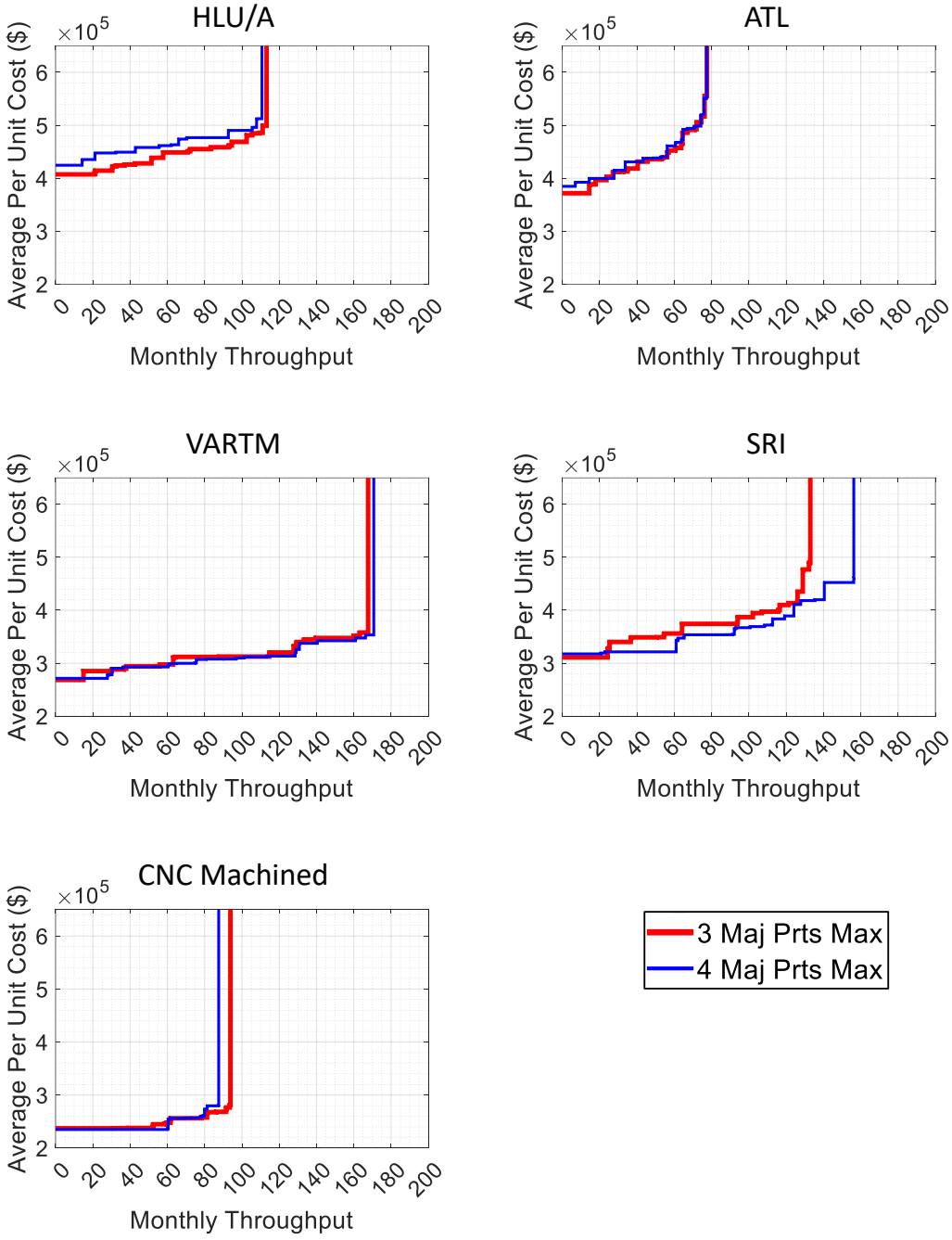


Figure 7.37: Average Summary Attainment Surfaces Comparing INFRADAT Results for All Processes for Case Where Max of 3 Major Part Combinations Allowed and Case Where Max of 4 Major Part Combinations Allowed

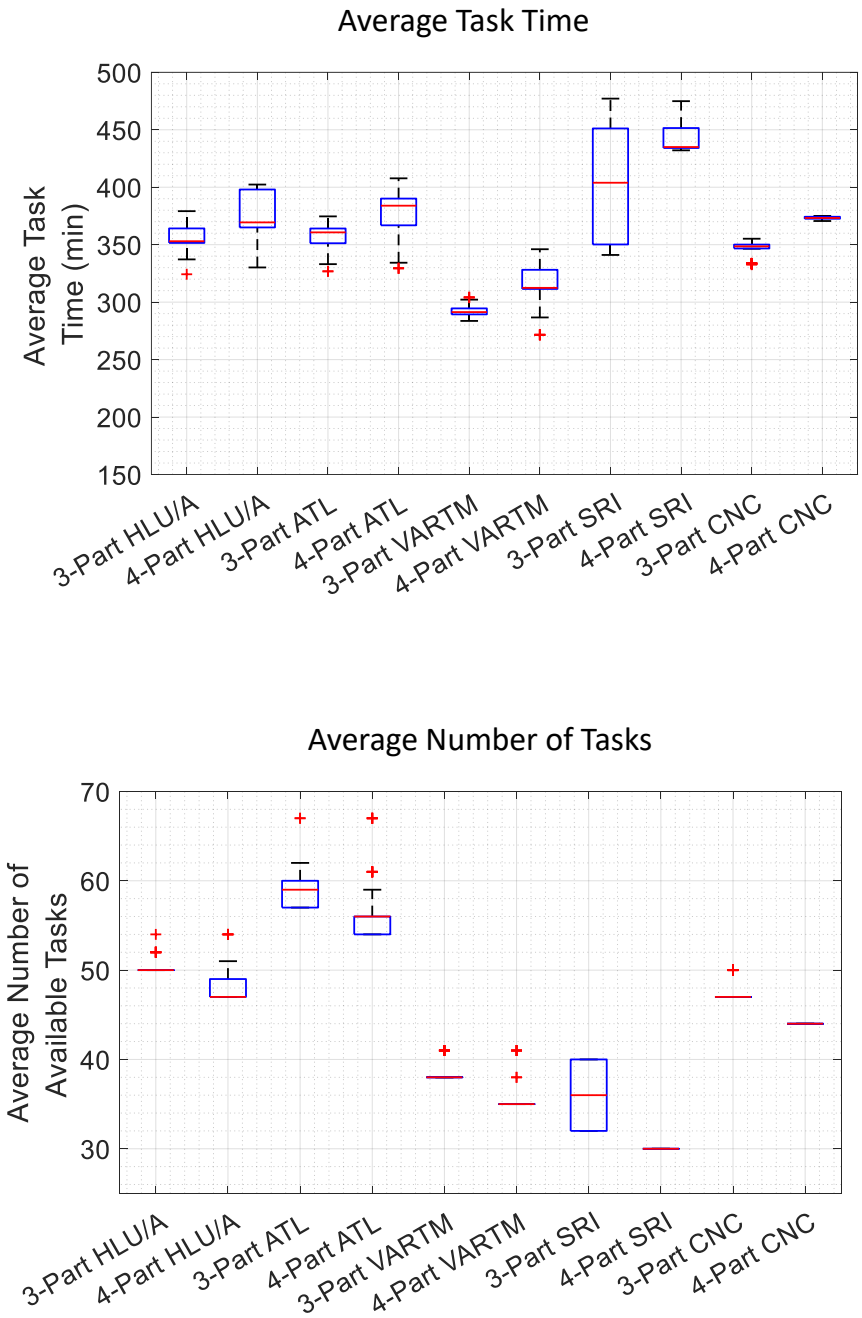


Figure 7.38: Boxplot Distributions of Average Task Times and Number of Available Tasks for INFRADAT Results for All Processes for Case Where Max of 3 Major Part Combinations Allowed and Case Where Max of 4 Major Part Combinations Allowed

Surprisingly, allowing a greater number of major parts to be integrally fabricated or in a subassembly does not always result in improved throughput or cost. For the HLU/ Autoclave and ATL processes, there is no improvement at all in being able to use a 4-part subassembly versus a 3-part one and actually appears to be detrimental. The reason why can be seen in Figure 7.38 where, despite the number of available tasks in those processes going down, their average times increased by an even larger factor, making the 4-part subassemblies actually disadvantageous for the HLU/Autoclave process and barely on par for the ATL process. This ratio of number of tasks to task time is roughly equal for the VARTM process with a slight advantage for the 4-part combination version, allowing it to barely outperform the baseline 3-part subassembly results. This ratio is also roughly equal with the CNC Machined process, hence why both sets of subassembly results performed about the same with the 3-part subassembly having higher maximum production rates. In general, increasing the subassembly part limit does not appear to bear any significant advantages for the F-86 wingbox use case because the decrease in the number of tasks tends to made up for by an almost equal or greater increase in the average task times, keeping the result production rates and costs roughly the same or slightly reducing them. The tradeoff capability that comes with being able to analyze different number of parts in a subassembly is thus able to provide insight as to whether larger subassemblies are worth pursuing before too many resources are dedicated to advancing that technology.

The SRI process, however, demonstrates that larger combinations of major parts for integral fabrication could indeed be worth it because its decrease in its number of tasks is proportionately greater than its increase in its average task times. This allows the 4-part integral fabrication combinations to generally outperform their 3-part counterparts by a noticeable margin until the lowest costs are being calculated. In fact, the 4-part combinations allow it to push its maximum production rate to almost the same level as the VARTM process, though its costs are much higher. This is because when major parts are integrally fabricated, their stitching allows for their major part assembly's drilling and fastening tasks

to be completely eliminated. This is in contrast to subassemblies where those tasks are essentially just combined with other major parts' drilling and fastening tasks, with their main advantage being a lower usage of aisle space due to the lower number of stations from all the parts being consolidated. In this situation, despite there being little benefit to developing technology for 4-major-part subassemblies, a case could be made for developing 4-major-part integral fabrication.

This tradeoff capability of not only being able to determine whether and which parts should be in a subassembly or integrally fabricated, but also being able to vary the limits of how many major parts are allowed in each, is thus an important one. It provides the designer with insight as to the benefits of using large subassemblies and integrally fabricated parts on top of their possible impact on the resulting production rates and costs. Without it, a designer is liable to believe that larger subassemblies are always better and could end up wasting valuable resources instead, such as if they had attempted to develop 4-part subassembly technology for the HLU/Autoclave process.

The last major tradeoff capability INFRADAT has that can significantly affect production rates and costs is that it is able to not just include spatial constraint considerations via station duplication, but it is also able to change the actual spatial constraints as well. This is useful because designers may own differently-sized factories and wish to see what the capabilities of each are. It can also be used to assess if the factories they currently own are able to provide the production rate capacity that they want to achieve. Similar to what was done in Experiment 1, results for a 500k ft^2 and 1M ft^2 factory are presented in Figure 7.39, with the difference between this one and Experiment 1's results being that this one includes aircraft design considerations.

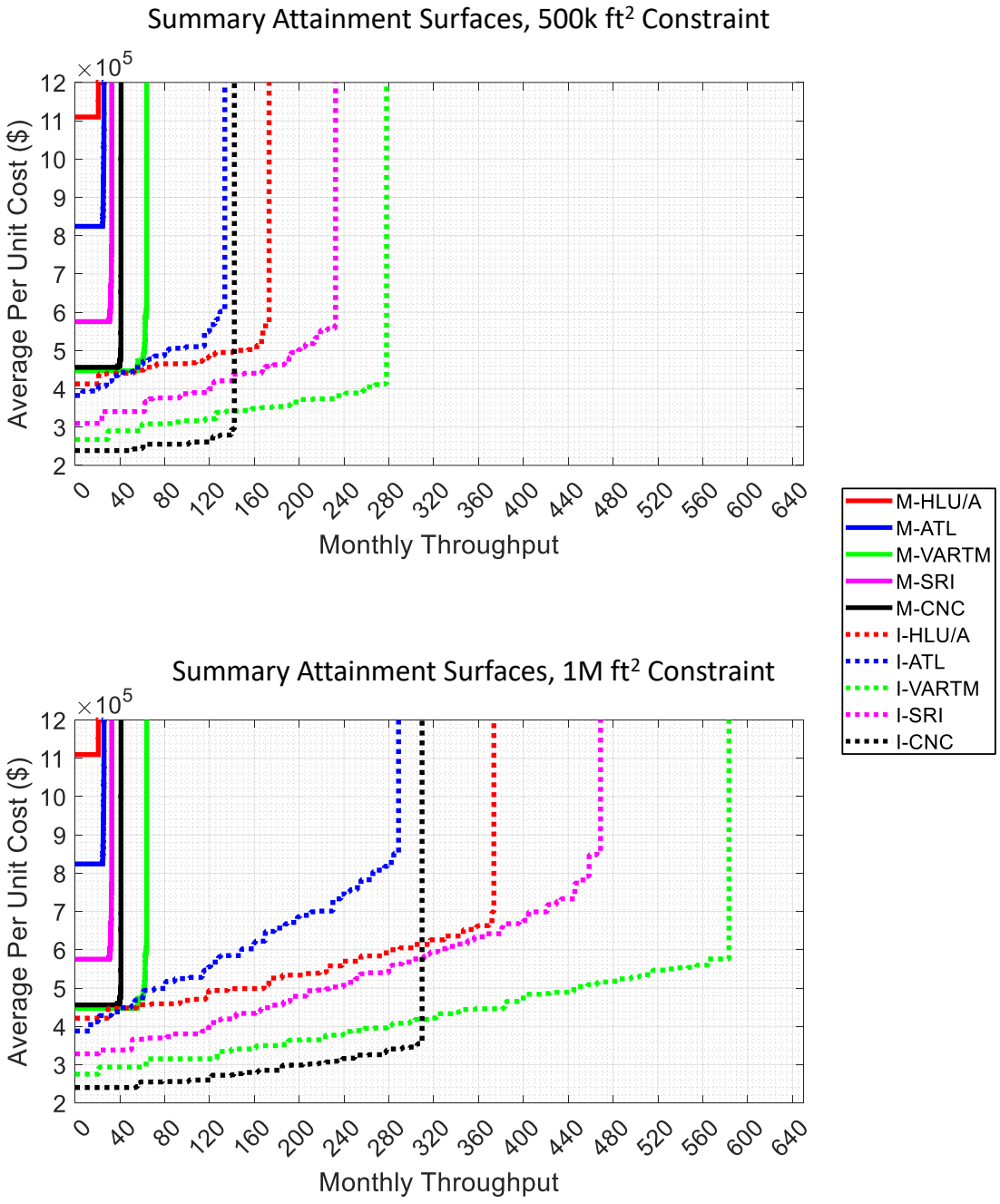


Figure 7.39: Average Summary Attainment Surfaces Comparing INFRADAT (I) and MInDPRO (M) for All Processes With No Limitations or Aircraft Constraints for Spatial Constraints of 500k ft^2 and 1M ft^2

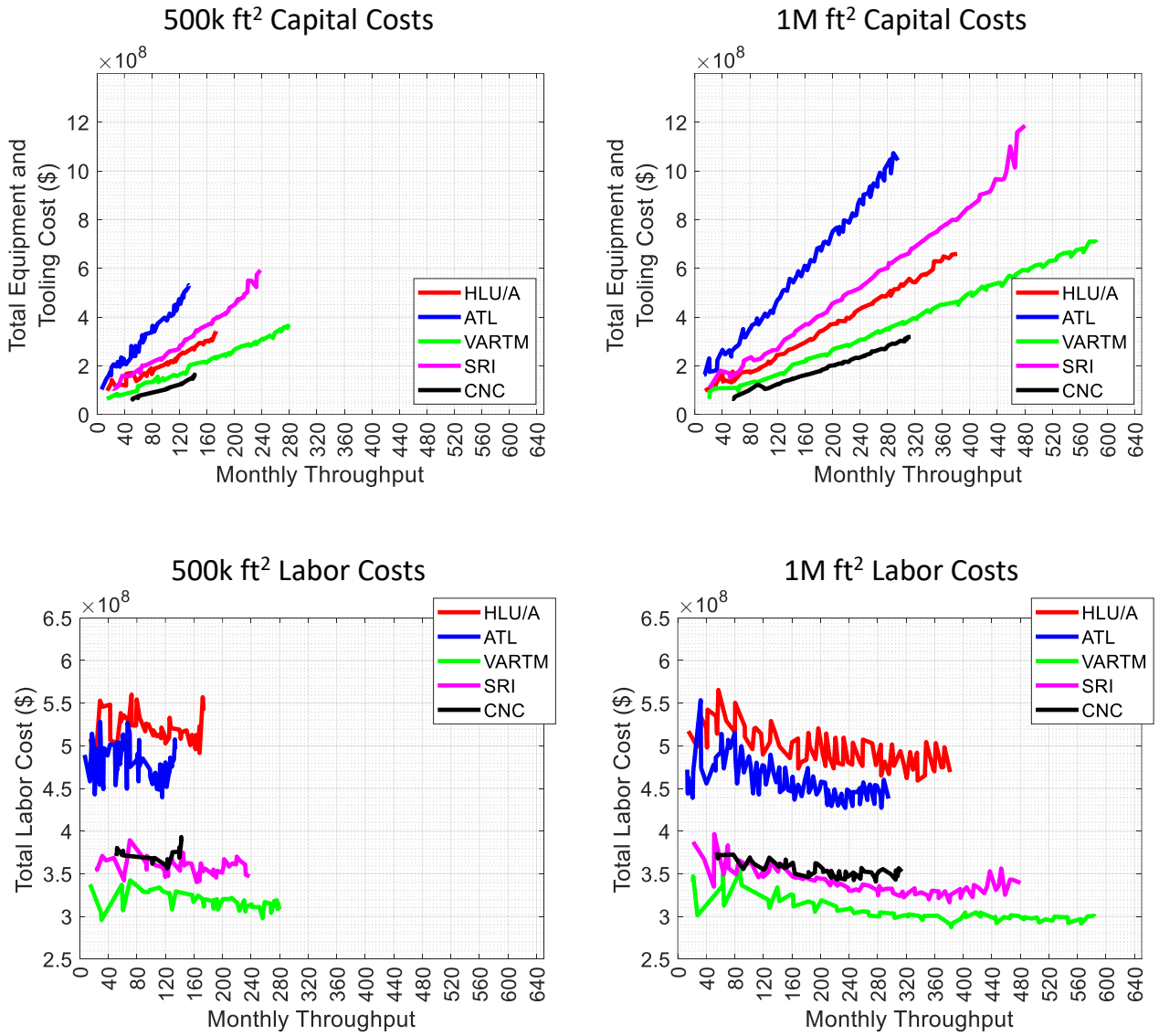


Figure 7.40: Combined Equipment and Tooling Cost and Total Labor Cost for INFRADAT Results for All Processes With 500k ft^2 and 1M ft^2 Spatial Constraints

The maximum production rates increase relatively linear as a function of the space available, with double the production rate obtained given double the space available. The costs, however, do not increase so proportionately and it can be seen that the APUC is not doubled despite the production rate being so. Examining Figure 7.40 reveals the same trend as was seen in Experiment 1 where the labor costs actually decrease given an increase in space and production rate due to station duplication yielding increasingly efficient station utilization ratios. This is because the product of all the variables in Equation 5.8 except the number of units in the production run stays relatively constant when increasing the amount of station duplication, causing the labor cost to not change drastically despite the number of workers going up. It essentially means the labor cost is primarily a function of labor efficiency and the number of units in a production run, not on the number of workers or number of stations. However, the increased efficiency does appear to be reaching an asymptotic limit, as opposed to the capital costs which linearly and proportionately increase over time. This is also why distinct spatial constraints should be sized for when two very differently-sized factories are available instead of basing everything on just the sizing of the largest one; the labor costs become very nonlinear in their trends as the spatial constraints are approached due to squeezing out every bit of production rate available at that point using the remaining amount of space. Additionally, the throughput ε-constraints will be very spread out and provide too coarse of a Pareto front for the smaller factories, while trying to make the constraint levels as fine as possible is an improper use of computational resources.

Based on these cost versus production rate vs spatial constraint tradeoffs, designers can weigh whether it is worth it to buy a brand new, even larger factory or whether a much smaller factory is sufficient. From this, it can be seen that INFRADAT's tradeoff capabilities regarding available space include both accounting for it when line balancing as well as being able to alter the total amount of space available. MInDPRO lacks even the capabilities for the former, never mind the latter. This tradeoff capability is ultimately

valuable because it allows the designer to see how they can make the best use of the factory or factories that they have as well as see if what they have is able to meet the capacity they desire. It provides a way to include physical, real world constraints on a problem that is normally modeled without accounting for space at all.

All of the discussed capabilities are ones able to be examined by INFRADAT on top of all the other tradeoff capabilities already demonstrated when trying to meet the first condition in subsection 7.4.1. This includes the following myriad of tradeoff capabilities INFRADAT has that MInDPRO does not. INFRADAT is able to use and take advantage of ASP to determine the best order to put together parts, allowing the designer to see which sequences are preferential and which are not. It can use this ASP to connect the impact of a part's shape on the aircraft's performance via structural integrity to its impact on the assembly performance via the accessibility of the resulting assembly sequences. INFRADAT can carry out formal ALB for different manufacturing processes and for variable costs and throughputs to allow the designer to see which manufacturing process's line balances are preferential at their desired production rate and cost levels. It can do this while duplicating stations and keeping track of how much space is used to let the designer know how much space is needed. It can also alter the high level spatial constraints to allow the designer to see if their factory is too big or too small for their desired level of production. On top of this, INFRADAT's spatial considerations are linked to the aircraft design variables via their size, allowing space constraints to flow up and influence the best aircraft design variables to use and creating a tradeoff between parts sized for the optimal mission versus parts sized for optimal space usage and manufacturability. And throughout all this, INFRADAT is able to take into consideration whether parts should be integrally fabricated, in a subassembly, or neither, which affects the assembly sequences and line balances at a fundamental level and captures one of the main draws of using novel materials in the case of integral fabrication.

INFRADAT is able to do all this while maintaining MInDPRO's primary capability of being able to incorporate aircraft constraints and have them be able to affect the resulting

production rates and costs. And throughout it all, INFRADAT is able to store this information and make it traceable back to the original inputs, as demonstrated via the numerous diagnostic results presented. This data traceability combined with all of its listed tradeoffs allows for a better capture of reality during the early aircraft and assembly design process and so gives the designer a better understanding of the problem and what its major constraints are. This all combines to mean that INFRADAT is able to carry out more tradeoffs than MInDPRO and captures more aspects of real-world manufacturing as they pertain to increasing production rates and reducing costs to the greatest extent possible. It means that INFRADAT is able to meet the second condition laid out in section 7.2 to prove the Overarching Hypothesis.

During this entire process, INFRADAT does not need the designer to provide input on variables such as what the assembly sequence, line balance, number of duplications for each station, number of toolsets, or number of worker sets should be. It is all optimized internally based on high level variables set by the designer and so does not burden them with having to sift through thousands of different assembly sequences and millions of possible line balances and the associated number of station instances and toolsets to pick the exact best combinations. Indeed, it does not even burden the designer with having to pick a range of such variables to explore since even that is likely to result in sub-optimal combinations. This is all in contrast to MInDPRO which requires the designer to set an assembly sequence and line balance for every manufacturing process used and then define a range of allowable station instances, sets of workers, and number of toolsets. Due to limitations with surrogate models not being able to handle many dozens of variables, the designer is further limited to only being able to assign those variables to groups of stations as opposed to every individual station. This all leads back to MInDPRO performing worse than INFRADAT in a traceable way.

Despite all these disadvantages, it must still be remembered that MInDPRO has many other capabilities not directly related to maximizing improvements to production rate and

cost that INFRADAT still has not been able to incorporate yet. This includes accounting for demand variability, tracking backlog buildups and queues and being able to shut down entire production lines due to it, and calculating manufacturer cash flow and ROI. INFRADAT has more tradeoffs related to increasing throughput and reducing cost and is better able to carry out those tradeoffs, but it cannot wholesale replace MInDPRO.

In general, INFRADAT is able to carry out what it was designed to do. It is able to incorporate all the identified geometry and material factors that can link the aircraft design and assembly disciplines together through the usage of ASP and ALB analyses. With all these factors combined into one integrated ASP and ALB framework, INFRADAT has shown it is able to outperform the current state-of-the-art systems-thinking method that considers aircraft design, which itself is equivalent to current state-of-the-art systems thinking methods in general. It does this by creating more and better feedback loops between early aircraft and assembly design, which were demonstrated through all the different tradeoff capabilities listed just above. This all strongly supports the Overarching Hypothesis and, in fact, answers the Overarching Research Question of how to combine early aircraft and assembly design to increase throughput and reduce cost. As a result of all this, the Overarching Hypothesis is accepted.

The same caveat explained before Experiment 1 regarding the F-86 experimental platform and use case must be repeated due to its importance. The F-86 experimental platform and use case, as well as this entire work, do not claim to provide improved or more accurate estimates of how much tasks actually cost or how much time they actually take as compared to validated cost estimating relationships or collected labor data. All of the trends and values observed and discussed in the Case Study and in this work are applicable only to the F-86 use case being analyzed during the early aircraft and assembly design phases and it is unknown if they can be extended to the real world to design a real aircraft wing. Though the F-86 use case was designed to be as realistic as possible, it is still just a platform to carry out experiments on and is likely not translate to real life on a 1-to-1 basis. The emphasis

is on the INFRADAT tradeoff capabilities able to be demonstrated using the F-86 use case and the subsequent types of insight gained, not the physical tradeoffs of the F-86 wingbox itself, as is apparent from the F-86 no longer being in production and having no potential to being a viable fighter again.

CHAPTER 8
CONCLUSIONS

8.1 Summary

The motivation of this work is to increase aircraft production rate and do so cost-effectively due to the expected large increases in commercial aircraft demand. Addressing assembly changes is deemed an appropriate way to handle this due to the assembly process being very influential on both production cost and rate. Post-production assembly changes, however, are unable to provide a large enough impact without becoming extremely expensive due to all the design decisions already being locked down. Making assembly changes during design, before the production starts, is an attractive alternative and has shown significant improvements in the past. Past and present efforts to account for assembly considerations during the design process were reviewed. The first and most conventional DFA methods are only able to operate during the detail design phase, where most design decisions have still already been locked down, similar to making post-production changes. Conceptual DFA methods address this shortcoming but only examine conceptual assembly design and are still limited to detail design on the aircraft design side, where most design trades and costs have still already been set. Additionally, conceptual DFA methods occur too early in assembly design to be able to consider resource usage, which is needed to predict production rates, while conventional DFA methods occur too late in assembly design, after the resource allocation has already been set, and additionally focus only on the product and so are unable to further alter the resource allocations. While the conceptual and conventional DFA methods can address product assembly time, they are inadequate for addressing throughput by themselves.

More modern DFA-based methods focus on the systems-level aspects of assembly and

include resource allocation as part of their analyses, thus being able to predict and address production rates. These methods, due to their diversity and holistic view of the entire design and production system, are called systems-thinking methods and include methods such as PLM and systems engineering. However, the majority of these only focus on the conceptual and preliminary design phases of assembly design and still do not incorporate aircraft design. The systems-thinking methods that do incorporate aircraft design conversely do not consider important assembly aspects that have large effects on production rate and cost. These represent large missed opportunities in both groups of systems-thinking methods to cost-effectively improve production rates. As these systems-thinking methods are the current state-of-the-art and do not take full advantage of each other's capabilities, the Overarching Research Question is asked, which is how assembly considerations can be integrated with early aircraft design to reduce cost and increase throughput.

To answer this question, systems-thinking methods considering aircraft design were chosen as the baseline group of methods to improve since they already incorporate both the aircraft design and assembly disciplines. They just need enhanced feedback loops between those disciplines to improve their capabilities. To do this, different factors and variables common to and optimized by both disciplines were identified to better connect them. These were determined to be factors related to the geometry of the aircraft and the material systems it uses. The factors pertaining to the material systems were further broken down into considerations for variable manufacturing processes to represent different material systems, integral fabrication for better usage of advanced materials, spatial constraints to account for different resource usages of different processes, and variable cost and throughput estimation so that high cost solutions are not locked in to when high throughput alternatives are desired. For the geometry factor, ASP was identified as a way to connect the geometry produced by aircraft design to their ease of handling during assembly. For the material factors, ALB was identified as an analysis type able to incorporate all the individual factors listed. Systems-thinking methods that consider aircraft design currently do

not include ASP and only have basic ALB capabilities, so using them to establish better feedback loops between the two disciplines through the geometry and material factors was deemed enough of an advancement to answer the Overarching Research Question, leading to the Overarching Hypothesis that if those two analyses types are used to better integrate early aircraft and assembly design, then better production rate and cost trades can be made.

Implementation gaps exist with this idea. Implementation Gap 1 is that there is currently no guidance on how to properly connect ASP with aircraft design since it has never been done before, with ASP only being able to be carried out given sufficiently detailed geometry data and early aircraft design tending to not be able to produce such detailed data while still being able to make large configuration changes that can improve manufacturability. Implementation Gap 2 is that there are currently no works in the literature able to simultaneously integrate all the material factors and their related characteristics together with ASP and ALB. Implementation Gap 3 is that no integrated ASP and ALB works in the literature have been designed to accommodate being run numerous times as part of the aircraft design process and so do not describe how to do so efficiently enough to make the aircraft design iteration loop feasible.

Research Question 1 addresses Implementation Gap 1 by leveraging the aircraft early preliminary design phase to be able to rapidly make and iterate through large configuration changes while still providing ASP with parts that have enough geometric data to perform geometric reasoning with. Research Question 1 also identified the appropriate levels of fidelity for geometric reasoning's geometric and mechanical feasibility analyses to allow the geometry data to be quickly and accurately processed.

Research Question 2 addresses Implementation Gap 2 by first implementing many capabilities related to the different material factors' characteristics. This includes incorporating rule-based modeling to connect ASP and ALB in the context of also enabling integral fabrication and subassembly usage, developing an ε-constraint based station duplication method to simultaneously account for factory spatial constraints and variable cost and

throughput, and using parametric time and cost estimation methods to allow for dynamic determination of task times and costs given changes to the manufacturing process, assembly sequence, line balance, usage of integral fabrication and subassemblies, and aircraft design. This is all combined with all the ALB problem types most relevant to the material factors to create an integrated ASP and ALB framework. This framework was tested in Experiment 1 against the current state-of-the-art work most capable of handling all the material factors in Implementation Gap 2 to show it has all the desired capabilities and is able to perform better than the current best literature method, justifying its development and use. The developed framework was shown to be able to obtain significantly better throughput and cost values in a justifiable and traceable manner.

Research Question 3 addresses Implementation Gap 3 by developing a two step approach where the best aircraft design configurations and assembly sequences are pre-selected using metrics representative of both ASP and ALB so that only the most promising designs and sequences are line balanced, reducing the overall size of the problem by not considering sub-optimal designs and sequences. Experiment 2 tested this two step approach by comparing it with the full fidelity approach from Experiment 1 and found that, while it was twice as efficient and can sometimes yield better results, it tended to produce slightly worse results if the problem size is not sufficiently large. Research Question 3's Hypothesis 3 was thus only partially supported, though the two step approach is still valid for cases where the problem size is confirmed to be very large.

Combining all the capabilities developed in Research Questions 1 through 3 resulted in the creation of the Integrated Framework for Aircraft Design and Assembly Tradeoffs, INFRADAT for short. INFRADAT was tested against the current state-of-the-art systems-thinking method that considers aircraft design in the Case Study, essentially the overarching experiment. INFRADAT demonstrated it had greater tradeoff capabilities as they pertain to improving cost and throughput, being able to obtain solutions with better cost and production rate values, justify why its values are better and trace the reasons why back to the

pertinent factors, and perform more tradeoffs than the current state-of-the-art. This was all traced back to its usage of ASP and ALB to incorporate the geometry and material factors, thereby creating additional feedback loops between the aircraft design and assembly disciplines that allowed these tradeoffs and improvements to be made. Because of this, the Overarching Hypothesis was accepted and the Overarching Research Question considered to be answered.

8.2 Research Contributions

8.2.1 General Contributions

The main contribution of INFRADAT and its development is the ability to flow through the entire aircraft design process and be able to size a factory and do so with the two disciplines connected together enough that changes in one can affect the other, allowing for tradeoffs to be made that enable the designer to see what the best ways to optimize throughput and cost are and also see the resultant estimated throughput and cost values once those changes are implemented. The current state-of-the-art systems-thinking aircraft design considering methods are able to do this as well, but INFRADAT can do so with significantly greater capability.

This greater capability is due first to the inclusion of ASP and the consideration of assembly sequences, which relate the geometry of the aircraft and its assembly to how they can be assembled and the ease with which that can be done. No systems-thinking methods have done this before because the ones that do examine aircraft design have never examined accessibility or assembly sequence feasibility, while the systems-thinking methods without aircraft design that have been able to look at ASP and accessibility have never been able to connect it with aircraft design. ASP thus bridges these two classes of methods and so makes INFRADAT more capable than both individually. The greater capability is also due to inclusion of the material factors and their related characteristics. No systems-thinking works of any kind have actively considered incorporating integral fabrication as one of their

tradeoffs. Dynamic line balancing and station duplication on the basis of available factory space, as well as using the ε-constraint method to determine intermediate solutions that are not maximum or minimum throughput ones, has additionally never been done before in any ALB works, never mind systems-thinking works of any kind. On top of this, the line balancing capabilities of the majority of systems-thinking works with aircraft design are subpar compared to their non-aircraft design counterparts, so INFRADAT including and significantly improving the ALB analyses further bridges the capabilities of the two classes of systems-thinking methods. And similarly, altering the aircraft such that it is still able to meet performance requirements while being more conducive to throughput and cost optimization during the ASP and ALB analyses is a capability not seen in the non-aircraft design systems-thinking methods.

Another contribution is the partial confirmation of the two step approach. The integrated ASP and ALB problem has never had to be iterated through so many times before since it has never had to be a part of the aircraft design process, and so a method able to improve the efficiency of solving that problem makes it more feasible to implement. The introduction of representative metrics able to consider ALB during the ASP analysis without having to carry out a formal line balance paves the way for further research into representative metrics in general. It also demonstrates it is indeed possible to bound the integrated problem by narrowing down the list of assembly sequences to perform line balancing on to just the most promising assembly sequences, which has not been explicitly acknowledged until now due to the relative scarcity of works integrating ASP and ALB.

There are a few other smaller, more implementation-oriented contributions as well. The first is the inclusion of fabrication tasks during ALB analysis to ensure fair comparisons of assembly sequences and line balances that do and do not incorporate integral fabrication, as well as to ensure fair comparisons of different manufacturing processes. Another implementation-focused contribution is the development of a way to perform crossover operations for assembly line balances in a genetic algorithm despite ALB precedence relations

constantly shifting due to being dependent on assembly sequences that are also constantly changing. And the third is the ability to dynamically determine different tasks and task times with which to carry out ALB for different assembly sequences, integral fabrication and subassembly combinations, manufacturing processes, and aircraft configurations. Few, if any, ASP or ALB works to date have been able to do this.

8.2.2 Specific New Tradeoff Capabilities and Associated Limitations

The specific tradeoff capabilities gained as a result of all this are as follows. The linking of aircraft design with ASP enables assembly accessibility to be calculated as a function of the aircraft's geometry, particularly its shapes. This is a new avenue with which the aircraft's geometry, and therefore its aircraft performance, can affect its assemblability, cost, and production rate in early design beyond simply just examining the assembly's size. The ASP capability itself allows designers to not need to know the exact optimal assembly sequence beforehand. There are currently a few limitations to this however. The usage of primarily visibility-based methods for this due to computational expense constraints means the specificity of the accessibility is relatively low; tool path checks and tool space usage calculations cannot be performed to calculate the feasibility aspect of mechanical feasibility, thus requiring simplifying assumptions on the correct-sized tools always being available and access holes always being present. Additionally, though the designer does not need to know the ideal assembly sequences a priori, they will need to know the general trend of the precedence relations beforehand in the form of the appropriate If-Then rules.

Many more new tradeoff capabilities exist on the line balancing side. The incorporation of factory spatial constraints with line balancing and station duplication enables estimation of the max throughput given a factory size, which can be used to see if the given factory size and/or max production rates are sufficient. Combined with the ε-constraint method, other possible configurations that use less space within a given factory size can also be obtained for lower throughput levels. The line balancing optimization process itself does not need

explicit inputs from the designer to perform the station duplication due to the usage of either the ε-constraint method or the factory spatial constraints. This means the throughput and labor efficiency can be maximized dynamically with minimal designer input. And the usage of a Pareto frontier to visualize the multi-objective design space allows the designer to see what the best possible production rate and cost combinations are and enable them to pick the most promising regions to explore. The limitations to this are that the factory is assumed to be owned and so the costs of buying a factory are not included. In a similar manner, the factory sizes that serve as constraints are discrete and not optimized for and so must be provided instead, meaning that trying to dynamically determine the perfect factory size for a desired throughput level is currently difficult. The line balancing also does not explicitly account for how the stations are oriented, packed, or arranged within the given factory space, meaning the estimated factory space required could be an underestimate. And the dynamic station duplication method described currently only allows for analyzing one aircraft model at a time for a given spatial constraint.

The integration of the aircraft design and assembly disciplines means that line balances can be performed not just as a function of the assembly sequence used but as a function of the aircraft's design as well. This is possible through the usage of task times and costs that vary as a function of the assembly sequences and aircraft designs used, allowing for the optimization to obtain better throughputs and costs by searching for assembly sequence and aircraft design-related improvements too. To do this currently, the assumption must be made that all the assembly lines are pulsed. Additionally, the line balancing is currently only able to account for deterministic task times, meaning variations in labor time as a result of efficiency or the learning effect are not included. In the same way, no movement times are accounted for. This means that, depending on the mode of transportation used to move the parts from station to station and their speed, the estimated throughputs could be overestimates. And in regard to being able to connect ASP and ALB together, the designer must know the tasks involved with fabricating and assembling each part or set of parts as

well as the If-Then precedence rules used to translate the ASP part sequence information into their ALB task counterparts.

The inclusion of fabrication tasks in the ALB, the usage of implicit precedence relations, and the leveraging of If-Then rules for rule-based constraint modeling to dynamically create ALB precedence graphs also means that assembly sequencing and line balancing can be performed as a function of integral fabrication in a scalable way. This enables the automatic selection and evaluation of many integral fabrication part combinations as opposed to just a few during the assembly sequencing and line balancing and allows for the determination of which part combinations yield the best throughputs and costs. This frees the designer from the pressure of manually selecting the single best combination a priori. The caveat, however, is that while the designer may not need to know what the single best combination is, they must still be able to define what the possible combinations are through the If-Then precedence rules.

The inclusion of fabrication tasks also enables line balancing for different manufacturing processes and the ability to directly compare them due to capturing both fabrication and assembly task differences. The best overall manufacturing processes can thus be selected based on the performance of their resulting line balances. The limitation to this is that the designer must know the task types that make up each manufacturing process and how they can change as a result of different assembly sequences. The designer must also have data on the size and cost requirements of the manufacturing process's different task types and how they scale with a part or assembly's size.

A more aircraft design-oriented tradeoff capability is that performance constraints can be set to see how the resulting production rate and cost effects on the different line balances, from which the best line balances and designs can be selected based on the subsequent solutions. Conversely, line balances and assembly sequences yielding the optimal costs and throughputs can be traced back up to their respective aircraft designs to identify the characteristics of aircraft that are the most manufacturable.

And lastly, the integrated aircraft design and assembly problem can be done with twice the usual efficiency and even obtain better quality results than the more conventional full fidelity approach by using the developed two step approach along with the developed representative metrics. The limitations with this are that the two step approach is only recommended when the number of possible assembly sequences in the problem space is very large, resulting in slightly more optimized line balances not being as important as more ideal aircraft designs and assembly sequences. When this condition does not apply, the default full fidelity approach should be used instead. Additionally, the two step approach and representative metrics are still relatively new as a concept and experimental and so it is unknown how far they can be extended and what their exact boundaries are.

8.3 Future Work

There are many possible avenues that can be pursued to improve INFRADAT, with the majority of them relating to being able to carry out higher fidelity sub-analyses at a computationally-acceptable level to allow for many of INFRADAT's existing capabilities to be expanded on.

The analyses for the feasibility and accessibility aspects of mechanical feasibility used in INFRADAT are limited in their sophistication due to the computational expense limits imposed by operating in the early aircraft design phases. Ideally, higher fidelity methods such as those related to tool use volumes and tool path planning are used for a more refined understanding of how accessible or mechanically feasible an assembly sequence is beyond whether its joining locations require the use of access holes and assuming an appropriately sized tool is always available, just perhaps more expensive. This can subsequently be used to further differentiate the assembly sequences and better distinguish their relative quality. However, this will need to be done in as inexpensive of a manner as possible since many such mechanical feasibility and accessibility analyses will need to be done for each assembly sequence, and as INFRADAT demonstrated, there will be many such sequences

evaluated. A future area of research is thus seeing how to implement these more detailed analyses in a manner amenable to the early aircraft design process.

The minor and major part integral fabrication modeled by INFRADAT currently assumes that the technology is fully mature or will become fully mature soon and so does not try to capture any of the structural ramifications of co-curing or co-bonding a minor part to a major part or co-curing a major part to another major one. This assumption may turn out to not be true in the future. To reduce the dependence on assumptions such as this, it is beneficial to add a structural analysis subroutine to the structural optimization that examines integral fabrication specifically. This is to see if the integrally fabricated parts need further pad-ups or other structural changes to account for the minor part integral fabrication. Essentially, as it is, integral fabrication's effects are primarily limited to the assembly discipline and so almost always seen as a positive, when in reality the situation is likely more complex. And in regard to the complexity, further structural analysis could reveal that, due to the different structural properties, integrally fabricated parts may need additional steps during their manufacturing to account for these differences, compounding the structural and assembly changes. Further research should be done to see how to cheaply implement integral fabrication analysis to account for its effects on aircraft design and possible subsequent effects on assembly and to allow more tradeoffs to be made between the two disciplines.

The benefits of further work on computationally inexpensive structural analysis is not limited to just part integral fabrication, but extends to better differentiating manufacturing processes as well. Different processes can have different effects on the resultant structure. For example, hand layups are prone to causing defects such as orienting the fiber improperly while automated tape laying is prone to causing wrinkles, both of which knock down the strength of the part and so require it to be thicker to compensate or cause the part to need to be reworked entirely. As another example, the SRI process's stitching significantly improves the adhesion properties of its joints, potentially allowing the integrally

fabricated part to be lighter or smaller. However, none of this is currently captured due to the lack of this analysis, with the possible advantages of SRI not being included and the possible disadvantages of the HLU/Autoclave, VARTM, and ATL processes not included either. Additional development on detailed structural analysis able to be performed during the aircraft early preliminary design phase would allow more tradeoffs to be made on these manufacturing processes in more than just one discipline.

Such developments would benefit in differentiating categorical part geometries as well. One of the main differences in part shapes not captured by INFRADAT is that different shapes cause the joints to be positioned and attached in different locations, which affects their joint strength. Being better able to capture the effects of the different joint strengths as a function of where the joints are located for each of the structural loads used will aid in further distinguishing part shapes from each other. It can subsequently result in larger weight differences between them and, if combined with the additional research in higher fidelity accessibility analysis mentioned above, can enable further tradeoffs between the aircraft design and assembly disciplines in the aspect of part shapes. This is not limited to just spar shapes, but can be extended to rib shapes or stringer shapes as well.

In regard to the assembly side, further research can be done on when exactly the two step approach is applicable and how to quickly determine if a problem is suitable for its use or not. Additional work can also be done to see if superior representative metrics exist as well as how they should be built and selected if it is determined an entire class of such metrics exists. On top of that, the two step approach's current efficiency improvements are limited due to the stopping criteria used, which themselves are a limitation set by the usage of the ε-constraint technique used for the integrated problem Pareto front generation. Improvements on either of those can also significantly improve the efficiency with which the integrated ASP and ALB problem can be solved.

One of the main assumptions used for modeling the ALB is that the entire production line is pulsed and that parts are essentially at the next station after an amount of time equal

to the cycle time has passed. While this is sufficient to capture the majority of the tradeoffs that exist between the aircraft design and assembly disciplines, it is not completely true to real life and so likely to overestimate real life production rates. The parts being so large mean that they will require time to physically move from one station to another. And the stations and factory being so large mean that the distances from one station to another are not negligible either, meaning they must be accounted for too. This results in the factory layout problem, where the physical location of the stations within the factory floor affects the travel time between stations and therefore affects their throughput. This dynamic inclusion of travel time means that the formal line balancing as presented and used will start to break down because it cannot be guaranteed that the production line is consistently pulsable anymore due to the travel time being dependent on how other parts are moving as well as where the stations are placed. The inclusion of factory layout and travel time means that a discrete event simulation similar to what was used by MInDPRO must be implemented since where the parts and resources currently are in the factory must be tracked at all times. This also means that the DES used for such a problem must be an improvement over MInDPRO's implementation and be able to dynamically determine assembly sequences, line balances, and number of station duplications and resources without needing to be determined or bounded at a low level by the designer.

Other possibilities for future work include augmenting INFRADAT to be able incorporate all of MInDPRO's other capabilities. This includes adjusting the ALB analysis to be able to account for learning curves and task time variability. It also includes adding additional analyses to be able to calculate manufacturer cash flow, determine ROI, and include demand variability to see what would happen if maximum production rate is desired but not needed at all times. These would all make INFRADAT a much more robust framework.

And finally, back when geometry and material factors and considerations were decided upon to better link the aircraft design and assembly disciplines, tolerance considerations were ruled out due to general unavailability of the data needed to make the desired tradeoffs

in early design. This is despite tolerance considerations and factors also having a significant effect on aircraft design and on the assembly process and so being able to incur large production rate and cost changes. Future efforts focused on being able to obtain that data early in design would prove extremely beneficial. Alternatively, developing techniques able to make all the tradeoffs related to tolerancing's impact on aircraft performance, effects on rework and repair rates, influences on equipment and tooling costs for each manufacturing process, and ramifications on different assembly sequences, and do all this for all the various geometries of interest and as a function of the tolerance requirements without requiring test parts be built or even needing the aforementioned data, would prove even more advantageous. This is because not relying on building expensive pre-production samples or large swathes of data that most companies would rather keep confidential would make the analysis much more flexible and feasible. Regardless, either effort would ultimately go a long way to enabling all three considerations linking the two disciplines to be examined at the same time.

Appendices

APPENDIX A

F-86 EXPERIMENTAL PLATFORM

This section goes over all the modeling assumptions used in the F-86 experimental platform.

The North American F-86 Sabre is a swept wing fighter jet most well known for being the US's primary fighter aircraft in the Korean War. The F-86F is the variant of interest and the experimental platform revolves around its wing and wingbox. Some parameters that define the F-86F, from now on called just the F-86, and its wing are displayed in Table A.1. They are adapted from [225].

Table A.1: Major F-86F Parameters, Adapted from [225]

Parameter	**Value**
Wing Sweep (25% Chord)	35°
Wing Taper Ratio	0.52
Wing Planform Area	313.4 ft^2
Aspect Ratio	4.88
Span	39.1 ft
Length	37.5 ft
Height	14.7 ft
Empty Weight	11,125 lbs.
Design Takeoff Weight	18,152 lbs.

A.1 Aircraft Design and Sizing

The goal of the design and sizing is to, given the OML geometry of the aircraft as well as some basic wing definition parameters, be able to determine the size and thickness of the wingbox's components such that it is able withstand the internal, aerodynamic, and

structural loads imposed by the mission it is flying and any other subsequent performance constraints and requirements. The flow of this process is laid out in Figure A.1 and is based on what was done for Siedlak et al.'s work [29].

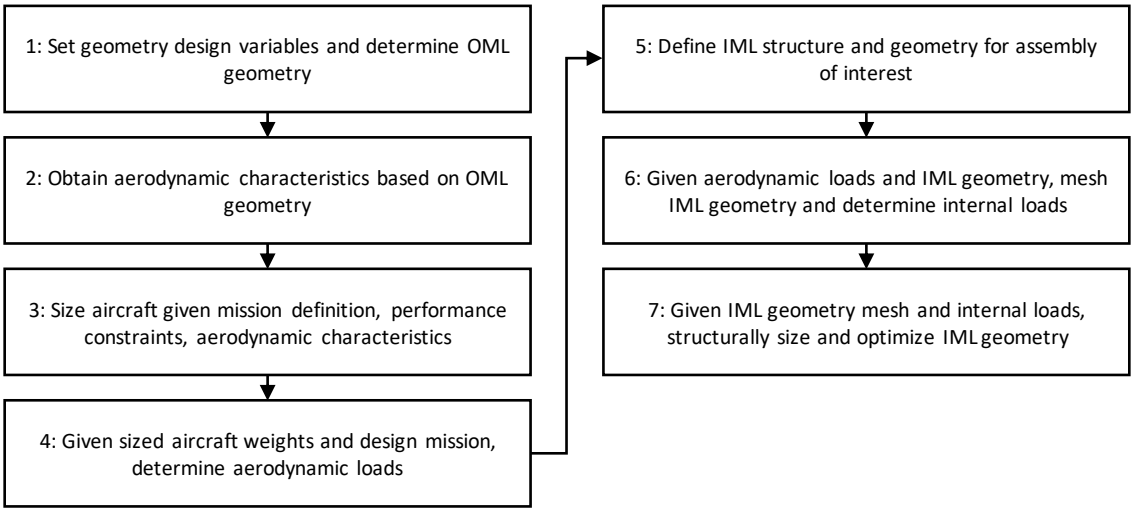


Figure A.1: Summary of Aircraft Design Process Leading to Wingbox Geometry Definition and Sizing

The OML geometry is created in NASA's OpenVSP program given the major parameters in Table 5.1 as well as other drawings of the F-86. The F-86 OpenVSP OML is shown in Figure A.2

The aerodynamic characteristics, which include the aircraft's parasite drag as a function of altitude and velocity and its induced drag as a function of angle of attack and velocity, are calculated using OpenVSP's Parasite Drag analysis module and VSPAERO analysis module, respectively. The velocities range from Mach 0.1 to 0.9, the altitudes from 0 to 50000 ft, and the angles of attack from 0° to 12 °. The Parasite Drag module does not account for transonic wave drag and so the parasite drag values are compared to the ones obtained in Siedlak et al.'s work and increased by 28% at roughly Mach 0.9. The aerodynamic characteristics, the definition of the F-86's design mission, and other performance constraints are afterwards both inputted into NASA's FLOPS programs to size its fuel weight, empty

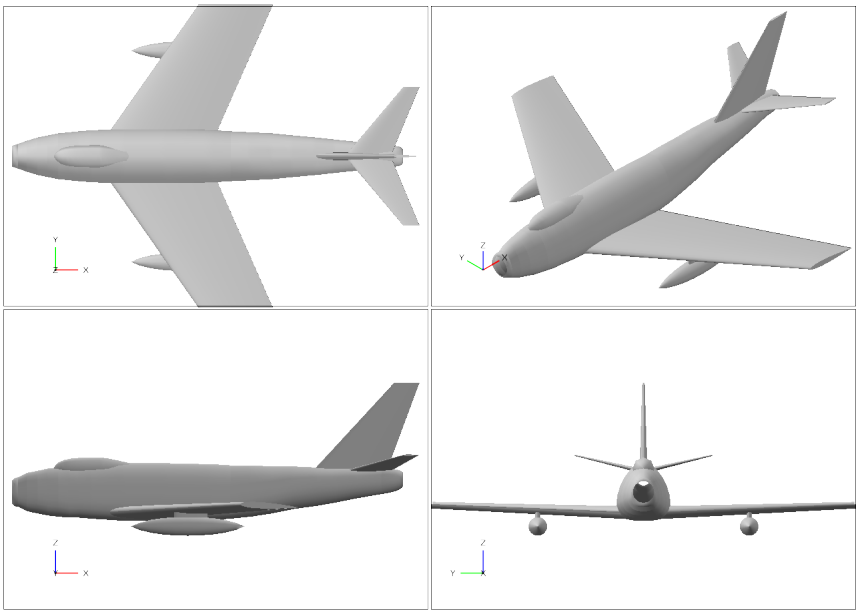


Figure A.2: F-86 OpenVSP OML Four View

weight, and takeoff gross weight. FLOPS requires the induced drag data to be a function of Mach number and coefficient of lift, which was obtained by interpolating VSPAERO's Mach number and angle of attack versus induced drag data. The mission is shown in Figure A.3 and is based on the data from [225]. The aircraft takes off, climbs to 39,600 feet, cruise climbs to 40,900 feet at 460 knots, combats for 20 minutes between 35,000 and 40,900 feet, cruise climbs further to 45,500 feet to leave the area, and then descends and lands. The engine data for FLOPS is based on data from [225], Siedlak et al.'s work [29], and from the engine's manuals, with missing data filled in via both linear and non-linear interpolation.

Steps 4-7 in Figure A.1 are done using Corman et al.'s [187] RADE framework. Step 4 is done using the Athena Vortex Lattice (AVL) program. AVL requires the specifications of the flight conditions to calculate the aerodynamic loads. Only three are defined and they capture the most strenuous flight conditions the F-86 faces in its mission, meaning the definition of any more is likely to be redundant and not needed. The first flight condition is to fly at Mach 0.9 with a load factor of 5 at combat altitude to capture the requirements of combat. A load factor of 5g is used because, based on [225], it is the max load factor the

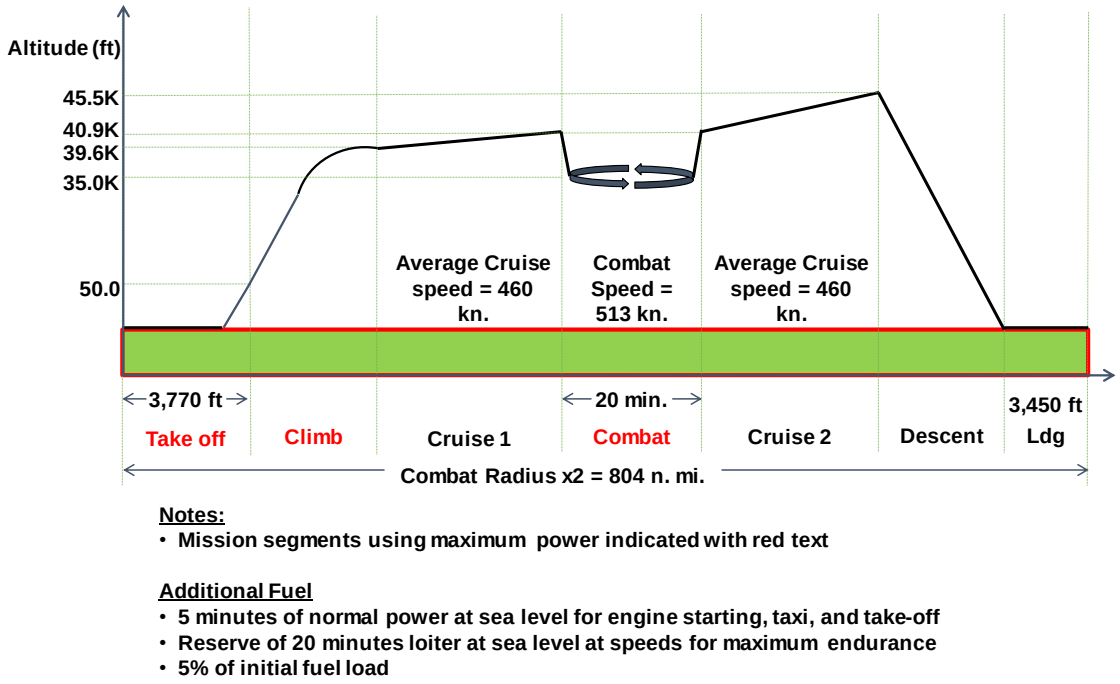


Figure A.3: F-86 Design Mission, Axes Are Not to Scale

F-86 can handle without dropping its fuel tanks, which is what the aerodynamic characteristics all include by default and so is more limiting than flying without fuel tanks. The second flight condition is to fly at cruising speed and altitude with a load factor of -1.0g to capture the possible need to fly inverted during non-combat conditions, similar to the requirements for a commercial aircraft. The third flight condition is to fly at cruising speed and altitude with a load factor of 2.5g to capture any possible, sudden need to perform basic non-combat maneuvers during regular flight, also similar to the requirements for a commercial aircraft. The aircraft is also trimmed out with a static margin of 0.05 at all these conditions to remain stable.

Step 5 is done using the RADE framework's proprietary Internal Mold Line (IML) structure generation program, with IML structures consisting of parts like spars, ribs, and stringers. In order to be able to easily and parametrically generate many different F-86 wingboxes, the actual F-86's IML structures are simplified to be composed of just two spars, a set of very similar ribs, and an upper and lower wingskin for the major parts. The

front and rear spars are placed at 15% and 60% of the wing chord, respectively. A rib spacing of 13.25 inches between ribs is used to get the baseline number of 18 ribs per wingbox and match Ceisel et al.'s work [60]. There are two wingboxes per aircraft, with a left and right wingbox for the left and right wing. The left and right wingboxes are symmetrical. The ribs are oriented to be perpendicular to the rear spar, which runs down the length of the wing.

The geometries and structures are then meshed. The wingskins, the spars' webs, and the ribs' webs are meshed as smeared shells, meaning they implicitly contain the stiffeners and stringers' stiffness, mass, and other properties without the stiffeners and stringers themselves being explicitly meshed. The spars' upper and lower caps, the ribs' upper and lower shear ties, and the ribs' spar-attachment joints are meshed as bar elements. These elements and this mesh are used by NASTRAN in Step 6 to calculate the internal loads. The F-86 wingbox is thus composed of five major part types and seven minor part types. The major part types are front spar, rear spar, upper skin, lower skin, and ribs, while the minor part types include the front spar's stiffeners, the rear spar's stiffeners, the upper skin's stringers, the lower skin's stringers, and the ribs' shear ties, stiffeners, and spar joints.

Step 7 uses the mesh of elements and internal loads from NASTRAN to perform structural optimization in the Hypersizer program. Before the optimization can be carried out, however, each part's shapes must be further defined, with the shapes of the stiffeners and stringers in the smeared shells identified and the bar elements given a cross-sectional shape. This is done in Hypersizer itself. For the smeared shells, the wingskin stringers are hat shaped, the spar and metal rib stiffeners have an L-shaped cross section, and the composite rib stiffeners have a U-shaped cross section. For the bar elements, the cross-sectional shapes determine the overall shapes of their respective major parts. Both metal ribs and composites ribs have L-shaped spar joints. The metal ribs' shear ties are I-shaped, the composite ribs' shear ties are C-shaped, and the spars' stiffeners are L-shaped. The spars' L-shaped upper and lower caps, combined with their web, means that the spars can have

either an outward-facing C or inward-facing C overall shape, depending on the orientation of the bar elements. For the standard F-86 as modeled in the MInD works, orientations resulting in outward-facing C spars are used. A stringer spacing of 7.2 inches is used to get 6 stringers on each wingskin at the root of the wingbox while a spar stiffener spacing of 10 inches is used. There are two spar joints on each rib, one to attach to the front spar and one for the rear spar. The number of rib shear ties and stiffeners is dependent on the number of stringers since the shear ties must slot in between stringers and the stiffeners must slot in between the shear ties. The number of shear ties is one less than the number of stringers while the number of rib stiffeners is one less than double the number of stringers. More details on the shapes of these parts can be found in Appendix B.

After the geometric details have been filled in, Hypersizer's typical default failure analyses for each material system are performed with an ultimate load factor of 1.5 applied. Regarding the materials themselves, Aluminum 7075 is used for the metallic processes and an AS4/3502 effective composite laminate is used for the composite processes. A 35/50/25 layup is used due to it resulting in fewer cases that failed from buckling and strength issues without having to manually tailor the layup down the length of every part for every configuration. The actual F-86 does not have access holes on its wingskin but instead on its spars. According to Niu, heavily loaded beams with cutouts in them, like the F-86's spars and their accessibility holes, are up to 50% heavier than ones without cutouts in them [259]. Therefore, to account for these access holes, the spar's webs are conservatively made 50% thicker. From all this, it can be seen that steps 5 through 7 in Figure A.1 are the implemented form of early preliminary aircraft design described in Conjecture 1.1.

A picture of the F-86's wing, the wingbox inside it, and its major parts' IML geometry is shown in Figure A.4. One of the other simplifications done to the wingbox is that the centerbox is replaced by an extended wingbox to reduce the already large number of different manufacturing tasks needing to be modeled for the parametric estimation.

With the wingbox being sized also comes an updated value for its weight different from

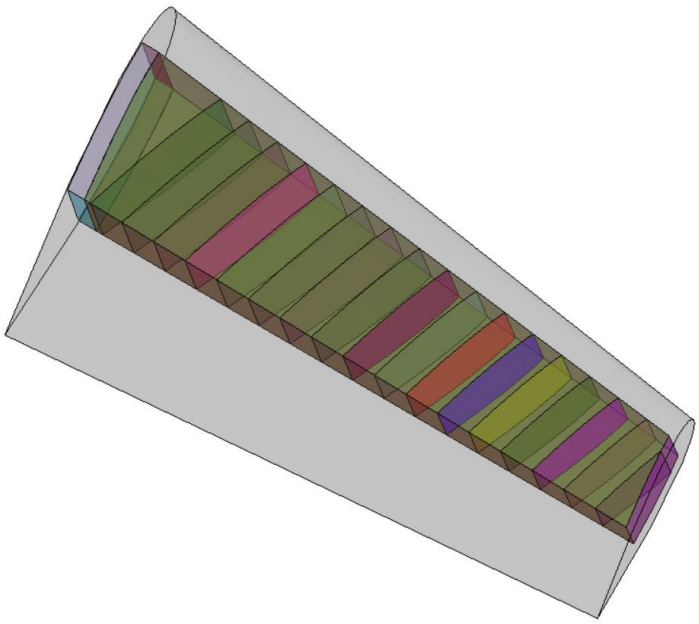


Figure A.4: Major Parts IML Geometry In Wingbox Inside Baseline F-86 Wing

the one estimated by the FLOPS program, which is used to obtain the final performance characteristics. To update these characteristics, the wing's weight is estimated given the weight of the wingbox. According to Ajaj et al., a typical wingbox makes up about 70% of a wing's weight, with secondary structures such as control surfaces, fittings, high lift devices, and leading and trailing edge structures making up the other 30% [260]. Based on this, the sized wingbox weight is doubled, since two wingboxes are needed for a full wing instead of half a wing, and divided by a factor of 0.7 to account for the secondary structures. This full wing weight is reinserted into FLOPS and replaces its internal wing weight estimation. The FLOPS program is run again and a new aircraft total weight along with all of its other updated performance characteristics is obtained. These are used to see if the F-86 is able to meet its performance requirements.

A.2 Geometric Reasoning

Once all the parts are appropriately sized and their geometries set, this geometry data is extracted and used to perform geometric reasoning and provide geometric constraint information for assembly sequencing. The procedures outlined in subsection 5.4.1 are used to calculate the geometric feasibility and accessibility. Given the level of detail of all the parts in the wingbox in the previous section, several observations are made. The first is that changing the size of the wing, resulting in changing the size of the wingbox and its constituent parts, does not affect the interference matrices for geometric feasibility in any way. Parts that cause interference will do so regardless of their size and, similarly, parts that do not interfere will continue to not interfere no matter how large or small they become. This applies for accessibility as well. The determinant of accessibility in this work is being able to form a path from a joining location to an accessibility point and these paths are not at all affected by parts being simply bigger or smaller. The second is that the order that the individual ribs are assembled in does not affect the assembly sequencing at all. Feasible sequences will stay feasible regardless of the order the individual ribs are inserted, and infeasible sequences will likewise stay infeasible. In the same fashion, the order the individual ribs are inserted does not affect accessibility. Any orderings that can possibly affect accessibility will result in infeasible sequences before the accessibility is even considered. This observation applies to all the rib stiffeners, rib shear ties, spar stiffeners, and stringers as well. Their individual assembly orderings relative to their own part types do not impact accessibility or feasibility at all. And the third observation is that the only assembly step that affects the accessibility in a feasible sequence is the last step that essentially closes out the wingbox. All the assembly steps before that are equally accessible, only the last one has any impact.

The result of these two observations is that several heuristics can be made that considerably reduce the time spent performing ASP for the F-86 wingbox. The first is that, because

the assembly order of the individual ribs does not matter, all the ribs are treated as though they are one part. If one rib is to be assembled, then they are all to be sequentially assembled at the same time or one after the other. The same is applied to all the minor parts. In regard to integral fabrication and subassemblies, it is impractical to integrally fabricate or subassemble some ribs and not others, or try to do the same to some minor parts and not others. Therefore, if one rib or minor part is to be in a subassembly or integrally fabricated, all the individual parts of that part type will be in a subassembly or integrally fabricated as well. The second heuristic that can be made is that because changing the size of the wing and wingbox does not affect the interference matrices, they do not have to be regenerated. The same ones can be used so long as all the shapes remain the same, meaning the projection techniques to create those matrices only need to be performed once. The third heuristic is that because only the last assembly step in an assembly sequence affects the mechanical feasibility, the mechanical feasibility analysis procedures will only be carried out for the last assembly step. This saves a significant amount of time by not having to perform this analysis for every single step. The fourth heuristic is that, in the same way that interference matrices only need to be generated once if the parts' shapes stay the same regardless of the size, the accessibility analysis is not dependent on the size of the assembly, only on geometry such as the parts' shapes and the last assembly step in the assembly sequence. All of these ASP heuristics that are valid for the F-86 wingbox are listed below.

- All ribs and minor parts are treated as only a single part each during assembly sequencing
- Interference matrices do not have to be regenerated so long as the parts' shapes stay the same, regardless of their changes in size
- Accessibility analysis only needs to be done on the last assembly step in an assembly sequence
- Accessibility is only dependent on the parts' shapes changing and the last assembly

step in an assembly sequence, not on the size of the part

These heuristics mean that there are effectively only five major parts for assembly: the front spar, the rear spar, the upper skin, the lower skin, and the ribs. Their respective minor parts are front spar stiffeners, rear spar stiffeners, upper stringers, lower stringers, shear ties and spar joints, and rib stiffeners. In terms of the assembly sequence representation, the front spars are represented by the number 1 in the first row, the rear spar with the number 2, the upper skin with 3, the lower skin with 4, and the ribs with 5.

A.3 Manufacturing Process Details

In this work, a total of five different manufacturing processes are examined: HLU/ Autoclave, ATL/Autoclave, HLU/VARTM, HLU/SRI, and CNC Machined. The ATL/Autoclave process from now will just be referred to as the ATL process, the HLU/VARTM process as just VARTM, and the HLU/SRI process as just SRI. The HLU/ Autoclave process is a composite process where pre-impregnated, or prepreg, carbon fiber is laid up by hand onto a mold, which is afterwards placed inside of an autoclave that applies tremendous amounts of heat and pressure to cure it. The ATL process replaces the HLU portion with an ATL machine that places the fibers down in an automated way before curing it in an autoclave. The VARTM process has dry plies of carbon fiber being laid down by hand instead of prepreg ones. A vacuum bag is then placed over the plies and the mold it is on and a vacuum is applied to compress and consolidate the plies, draw resin into the bag, and impregnate the plies. The bag, mold, and impregnated plies are moved into an oven that applies a tremendous amount of heat to cure the plies. SRI is similar to VARTM except that the plies are stitched together first with thread before they are vacuum bagged. The CNC Machining process is a metallic process where a block or billet of metal is machined into the correct shape using milling and drilling machines.

There are a few different reasons these specific processes were chosen to be a part of the F-86 experimental platform. One of the primary ones is that the HLU/Autoclave, ATL,

VARTM, and CNC Machining processes are all very well documented in previous MInD works and so are easily implementable. SRI is very similar to VARTM and so, given a few changes based on works by Linton et al. [232] and Velicki and Jegley [233], just as easily implementable. The main reason is that each process has a particular characteristic related to the material factors of interest that can be used to show the associated tradeoffs. HLU/Autoclave and ATL both operate on composite materials and are the benchmark for all composite processes. VARTM operates on composite materials as well and has a more advanced version of integral fabrication for its minor parts. SRI also uses composite materials and is one of the only known processes able to support integral fabrication of major parts, not just minor parts. And CNC Machining operates on aluminum materials, providing a material system contrast with the composite processes and also being the baseline that the aircraft industry has traditionally used for many decades. The tradeoffs that can be made with these five is why they are of interest.

The ATL, VARTM, and SRI processes use slightly different material systems as compared to the HLU/Autoclave process due to each of their process's needs: the ATL process's carbon fiber must be amenable to being laid out via machine, the VARTM process's must be amenable to having resin infused instead of being prepreg in nature, and the SRI process's have the same requirements as the VARTM process's on top of the fact that they must be amenable to being stitched. This normally necessitates different carbon fiber systems be used during the sizing process described in section A.1. However, the required material properties for each material system to perform the sizing tend to be proprietary and so are generally unavailable, especially for the more advanced processes and materials. As a result, the material properties are assumed to be roughly the same as the baseline composite material system described in section A.1, meaning all the composite processes have roughly the same structural strength.

On the manufacturing side, all of these processes carry with them different precedence constraints that cannot be captured via liaison graphs or interference matrices and so must

be captured via If-Then precedence rules for ASP. These are listed below, starting with general constraints that apply to all processes and then moving down the list of constraints that apply to only the CNC Machining, HLU/Autoclave, ATL, VARTM, and SRI processes. More rules could be added that further eliminate infeasible integrally fabricated parts and subassemblies, but these are already captured via the interference matrix updating step creating infeasible matrices and so are redundant to include here.

- IF wingbox is assembly being analyzed, THEN all minor parts must be assembled before their major parts can be assembled with other major parts
- IF parts are integrally fabricated, THEN they must be integrally fabricated before they can be assembled with other major parts
- IF parts are part of subassemblies, THEN they must be preassembled into said subassemblies before they can be assembled with other major parts
- IF major parts are integrally fabricated or part of subassemblies, THEN that subassembly or integrally fabricated part must consist of a minimum of two major parts
- IF major parts are integrally fabricated or part of subassemblies, THEN that subassembly or integrally fabricated part can consist of a maximum of three major parts
- IF a process other than SRI is used, THEN major parts cannot be integrally fabricated
- IF CNC Machining is manufacturing process used, THEN integral fabrication cannot be used
- IF stitched resin infusion is manufacturing process used AND IF major parts are integrally fabricated, THEN all minor parts associated with those major parts must also be integrally fabricated

With the If-Then precedence rules for assembly sequence generation covered, attention is now turned to the rule-based modeling used to convert those assembly sequences into

task information for ALB. Before the rules themselves can be covered, the task breakdown for each manufacturing process must be explained first. As explained in subsection 5.4.2, the tasks for all the processes can be broken down into fabrication task sets and assembly task sets. Typically, there is one fabrication task set for each part and one assembly task set every time a part is inserted, an integrally fabricated part is fitted up, or a group of parts are pre-assembled into a subassembly, all as determined by the assembly sequence. Assembly task sets are focused on first. The assembly tasks for all five of the covered processes are manual in nature due to almost all aircraft assembly tasks for all processes requiring manual work. Automated assembly machines are currently unable to meet the exacting tolerances and oftentimes non-standard fits involved with aircraft assembly and so automated assembly lines have not been implemented in aerospace industry yet. This means that the assembly tasks for all five processes are extremely similar with the only notable differences being with regards to material handling because there are some quirks to composite assembly that do not exist for aluminum assembly. That being said, disregarding any differences caused by the existence of subassemblies and integrally fabricated parts that will be covered later, a typical assembly task set is composed of three tasks: fitting up the parts to situate and locate them properly to be assembled, drilling through them to create holes for the fasteners that will connect them, and then installing said fasteners. These tasks stay the same even with different materials and processes, with those material differences primarily handled by the parametric time and cost estimation.

The fabrication task sets are where the majority of the manufacturing processes' differences are. A typical fabrication task set for the HLU/Autoclave process has 4 tasks, the ATL process has 5 tasks, the VARTM process has 4 tasks, the SRI process has 5 tasks, and the CNC Machining process has a variable number of tasks depending on the part type being fabricated.

The HLU/Autoclave process's first fabrication task is to lay up the carbon fiber plies by hand and cut the plies as needed via manual cutting. The kitting of the plies is included

in this task by default along with debulking and compacting the part every time 5 plies are placed down. Both of these aspects are done for all hand layup tasks. The second fabrication task is to prepare the plies on their mold before inserting them into the autoclave, where they are cured over the course of 6 hours before being taken out and removed from their mold. Due to an industrial autoclave and resin being used, the 6 hour cure is all that is needed to fully cure the part, with no post cure necessary. This is true for the ATL process as well. Afterwards the part is trimmed via hand router before going through NDI via an ultrasonic pulse echo machine. The NDI machine and task is based off of PaR System's LaserUT laser ultrasonic NDI system, which can scan at speeds up to 160 ft^2 per hour [261].

The ATL process's first task is to lay down the plies via an ATL machine at a very conservative, by current technology standards, rate of roughly 50 pounds per hour, which was obtained from previous MInD works. The plies are debulked and compacted every 8 plies since the machine head will be able to squeeze out air bubbles better than laying down by hand. The process is called automated tape layup instead of automated fiber placement due to the desire to be more inclusive of the larger ply widths used when laying down plies for the wingskins, which are 6 inches in width and deemed too large for the process and task to be called automated fiber placement. The second task is to hot drape form the parts in a hot drape form machine to get the parts into their final shapes. This task is needed because the ATL machine can only lay down parts on a relatively flat surface, they must be given their final curvature and contours with another task. As the wingskins do not have significant curvature, this second task does not apply to them for this process. The third task is to cure the part in an autoclave similar to the HLU/Autoclave process. The fourth task is to trim the parts via water jet, which is a more automated method than the HLU/Autoclave process's hand router. The final task is the same NDI task as for the HLU/Autoclave process and, in fact, all of the composite processes end with the same NDI task.

The VARTM process's first task is similar to the HLU/Autoclave process's hand layup task except that it uses an ultrasonic knife machine to cut the plies instead of manually cutting them by hand. Dry carbon fiber is also used instead of prepreg due to the resin infusion that will occur in the next task. The second task is to bag the part, compact it, infuse resin into it, and then place it in the oven and let it cure in there for 2.5 hours before removing it and debagging it. The third task is to trim it via water jet and the fourth task is to perform NDI on it in the same way as the ATL process.

The SRI process is identical to the VARTM process except that if there are any minor or major parts being integrally fabricated, that specific part's fabrication task set will have an additional stitching task between the hand layup and oven cure tasks. The stitching is done on an automated stitching machine and involves threads being sewn into the parts to connect them together in a similar manner to fasteners. The rest of the SRI process is the same as the VARTM process. The SRI process is modeled this way to be as generic and conservative as possible for this new and developing manufacturing technology to leverage its similarities to a previous process while still acknowledging its main benefits, which is the usage of stitching.

The CNC Machining process's tasks vary depending on which parts are being fabricated. For the spars, their first task is that they are machined out of a block of aluminum via a CNC milling machine. This milling task allows for the built-in creation of the spar's stiffeners as well as its flanges, meaning that the spar stiffeners are essentially integrally fabricated by default with this process. The second spar fabrication task is that they are inserted into a vat of dye penetrant, rinsed off, have developer applied onto them to make noticeable any flaws or defects, and then inspected via ultraviolet lights as part of their NDI task. The third spar fabrication task involves them being anodized to increase corrosion resistance. It is composed of 6 steps, which are based on the anodization process described by Norsk Hydro [262]. The steps consist of first degreasing the part, then rinsing them, acid etching them to remove imperfections, rinsing them again to desmut them, anodizing

them, and performing one last rinse in ionic water. Each step requires its own tank of the appropriate solutions and liquids. The fourth and last spar fabrication task is to spray the part with a zinc chromate primer after solvent wiping it to clean it off. The chromate spray is done to further improve its corrosion resistance properties. These last three tasks, the NDI, anodize, and chromate spray ones, constitute the last three tasks for all the parts in the CNC Machining process and are the same in all three.

The first fabrication task for the ribs for the CNC Machining process is much the same as the spar's, with a milling machine being used to mill out all of the ribs' features. This includes its shear ties, stiffeners, and spar joints, making those minor parts also integrally fabricated by default in the same way the spar's minor parts are. The ribs' last three tasks are the same as the spar's.

The wingskins in the CNC Machining process are first pre-formed via manual drill routing to get them into the correct overall shape. Stretch forming on a Hufford stretch forming machine is then done to induce the correct amount of curvature on the wingskins. 3 separate stretch forms are performed, with one setup used for all 3 stretch forming operations for a single wingskin. They are post-formed via manual drill routing afterwards before a CNC machine is used to trim off any undesired residual pieces. The last three fabrication tasks are the same as the others.

The stringers are first pre-formed in a numerically controlled drill router, due to their large numbers, before being roll formed into roughly their final shapes in a roll forming machine. After that, they are post-formed in another numerically controlled drill router and afterwards go through the same three tasks as every other CNC Machined part.

A diagram summarizing all the constituent tasks of each process's fabrication and assembly task sets is shown in Figure A.5, Figure A.6, and Figure A.7.

With the task sets defined, the total number of tasks can now be set. There are five major parts and seven distinct minor parts. As the major parts are tantamount to the assembly sequence, all five need their own individual fabrication task sets. For the minor parts, given

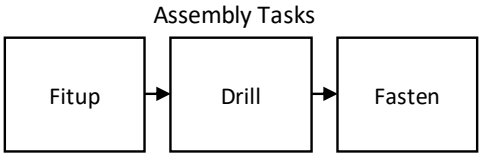


Figure A.5: Summary of Tasks in Assembly Task Set

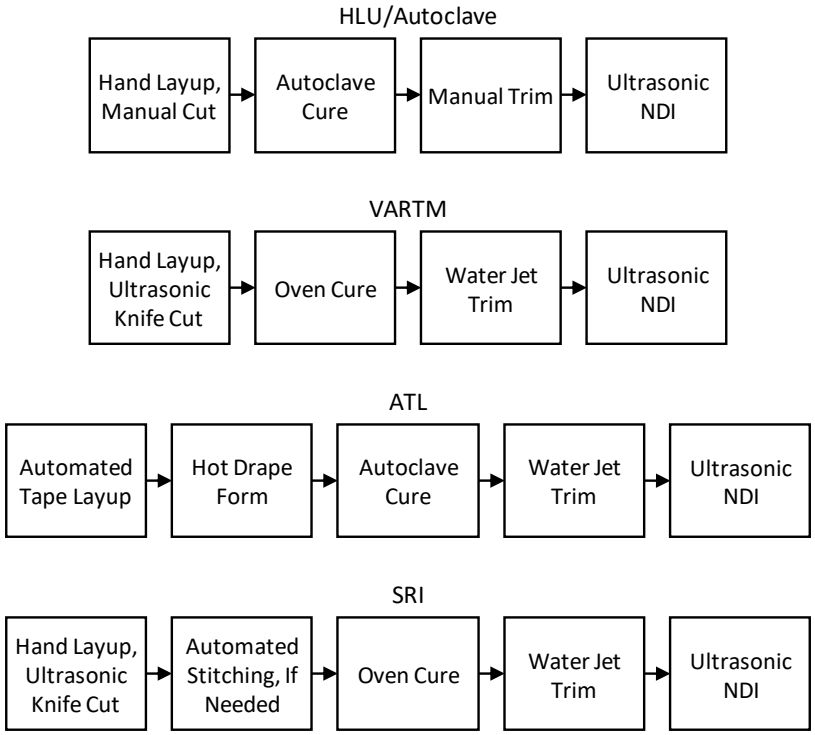


Figure A.6: Summary of Tasks in Composite Fabrication Task Set

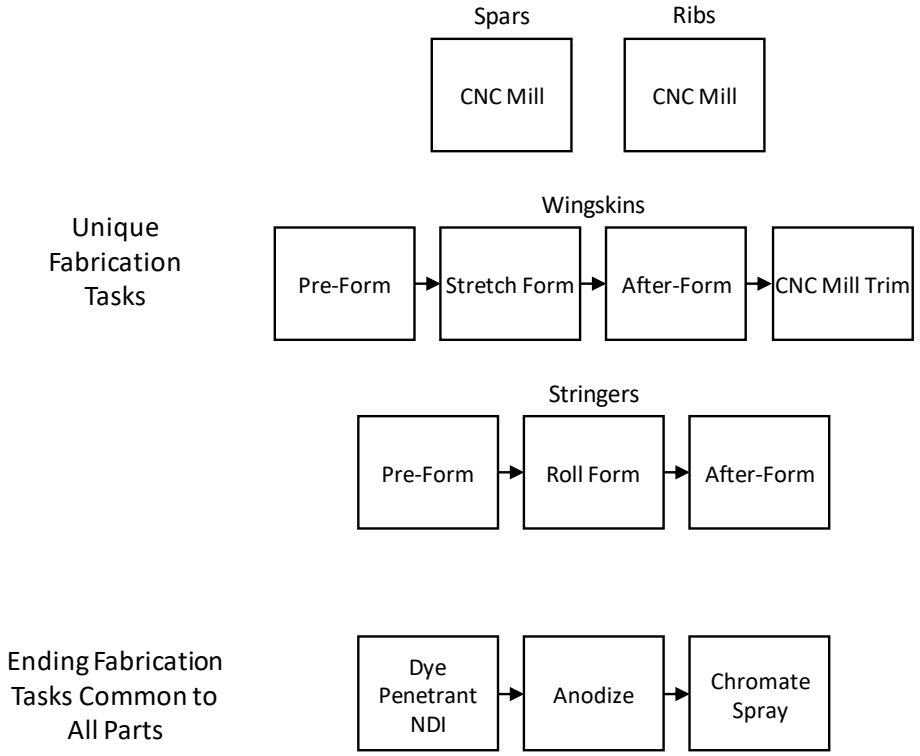


Figure A.7: Summary of Tasks in Metallic Fabrication Task Set, Separated By Appropriate Parts

a set wingbox size, the stringers are all identical to each other, as are the spar stiffeners. There is thus no need to differentiate between upper and lower stringers or front spar versus rear spar stiffeners, meaning stringers get one fabrication task set and spar stiffeners another, except in the case of the CNC Machining process for the latter, where they do not exist on their own. Another caveat is that the stringers and stiffeners do need to be differentiated for their SRI integral fabrication stitching task, meaning the SRI process has two add-on fabrication tasks compared to all the other processes. Rib shear ties and spar joints are almost identical in shape and also belong to the same major part. They therefore only have one fabrication task set. The last minor part type, the rib stiffeners, by default also have their own fabrication task set. None of the rib minor parts have their own fabrication task sets for the CNC Machining process, however. This totals up to 9 fabrication task sets for the composite processes, with an additional two tasks for the SRI process, and 6 fabrication task sets for the CNC Machining process.

For the assembly task sets, there are only 5 major parts, indicating a maximum of 4 assembly task sets for the major part operations assuming no integral fabrication or subassemblies are used. Additionally, each of the minor parts also needs an assembly task set for when no integral fabrication is used. This total up to 9 assembly task sets for all the processes, consisting of 4 major ones and 5 minor ones. Adding this all up, the maximum number of available tasks is 63 for HLU/Autoclave and VARTM, 72 for ATL, 74 for SRI, and 50 for CNC Machining.

With the usage of subassemblies and integral fabrication, however, it is unlikely that the list of available tasks will actually contain all the possible tasks, and so how subassemblies and integrally fabricated parts affect task availability is discussed next. Subassemblies are covered first since they are simpler. Subassemblies are essentially just regular assemblies that can allow two or more parts to be assembled together in a single assembly operation or with a single assembly task set. And like regular assemblies for the wingbox, the minor parts must be attached to their major parts first before major parts can be subassembled.

Because of these characteristics, subassemblies with 3 major parts reduce the total number of assembly operations, and therefore the number of assembly task sets, by 1. Theoretically, if subassemblies with 4 major parts were allowed or possible, the number of assembly task sets would subsequently be reduced by 2, leaving only 6 major part assembly tasks or 2 major assembly task sets. Subassemblies thus do not significantly affect ALB precedence relations beyond changing the number of tasks in the list of available tasks. Because the CNC Machining process does not allow for integral fabrication and only allows for subassemblies, this means its precedence relations and resulting precedence matrix is the simplest of the five to construct.

The integral fabrication for the composite processes is much more involved. Two types of integral fabrication exist among the four composite processes, co-bonding and co-curing. Both of these, as used to attach minor parts to major parts, are assumed to be mature enough of a technology to no longer need so-called "chicken rivets." Co-bonding is the act of taking a fully fabricated minor part, fitting it up to a major part just before its cure task using the minor part's fitup task, and then curing both the major part and its minor part at the same time. The minor part is afterwards permanently attached to the major part and goes through the major part's trim and NDI tasks with it. It is only available for the HLU/Autoclave and ATL processes. As the description suggests, co-bonding is only possible between minor parts and their corresponding major parts. Their benefit is that they eliminate the need for fasteners to attach the minor parts to their major parts, reducing the tasks in the minor part's assembly task set to just the fitup task that is used to locate the minor part onto the major part. The minor part's drilling and fastening tasks are completely eliminated. For the ribs, it is impractical to co-bond just one of its minor parts and not the others, and so if it is decided to perform co-bonding for the ribs, the shear ties and spar joints as well as the rib stiffeners will all be co-bonded to the ribs. Co-bonding can thus result in significant precedence relation changes beyond just changing the number of tasks in the list of available tasks.

Co-curing is even more advanced than co-bonding. Co-curing is the act of taking a

minor part that has not gone through its cure task yet, fitting it up to a major part that also has not gone through its cure task yet using the minor part's fitup task, and infusing resin into both of them and curing them at the same time. As they are now joined together, they go through the same trimming and NDI tasks. It is only available to the VARTM and SRI processes because resin infusion must be used in order to co-cure a part; the other two composite processes use prepreg plies that cannot be infused with resin because they come with resin already impregnated. Co-curing is more complicated than co-bonding because not only does it eliminate a minor part's drilling and fastening tasks, it additionally eliminates its trimming and NDI task because it uses the major part's version instead. The complication ends here with VARTM because VARTM cannot integrally fabricate major parts; the SRI process can.

Not only is the SRI process able to integrally fabricate major parts, but it requires an additional integral fabrication-only fabrication task any time integral fabrication is used. This exists in the form of the stitching task, the second task in the SRI process fabrication task set. If a minor part is to be integrally fabricated for the SRI process, the minor part and its major part are fitted up using the minor part's fitup task and located after they have both gone through their hand layup tasks. They are then stitched together via the stitching task which, in the precedence matrix, is represented by the minor part's stitching task. If the major part is not subsequently integrally fabricated, then the major and minor parts go through the major part's oven cure, trim, and NDI tasks like in the VARTM process, with the minor part's drill, fasten, oven cure, trim, and NDI tasks all eliminated. If the major part is subsequently integrally fabricated, then all the major parts that will form the integral part are fitted up using one of the major part's fitup tasks. All major parts that partake in integral fabrication must also be integrally fabricated with their minor parts and must already be stitched together with them beforehand because the resin infusion must occur for everything at the same time; the constituent parts of an integrally fabricated part cannot selectively choose whether they have been infused with resin beforehand. After all the

major parts are fitted up, they are stitched together using one of the major part's stitching tasks, after which the integrally fabricated part goes through that major part's oven cure, trim, and NDI tasks. The resulting integrally fabricated major part does not need any further operations to assemble its constituent parts together because the stitching threads act as fasteners. In total, if major parts are integrally fabricated, then all the constituent minor parts' drilling, fastening, oven cure, trim, and NDI tasks are eliminated as well as all but one of the constituent major parts' oven cure, trim, and NDI tasks. In regard to the assembly task set, if two major parts are integrally fabricated, then one of the major assembly task set's drilling and fastening tasks is eliminated; for three major parts, one of the major assembly task set's drilling and fastening tasks is eliminated as well as another entire major assembly task set; and for theoretically four major parts that are integrally fabricated, two entire major assembly task sets are eliminated while the third is only able to keep its fitup task. All these precedence requirements can thus make the precedence matrix for SRI processes quite complicated.

An additional note for the SRI process is that, according to works such as Velicki's [263] and Karal's [264], stitching is able to safely replace the use of fasteners when combining together major parts. Stitches are required to assemble together major parts without fasteners because their connections are much more critical than the minor parts' connections to their major parts. The minor parts for the SRI process are stitched whereas the ones for the HLU/Autoclave, ATL, and VARTM processes are not to further improve their fatigue properties, as noted by Aymerich [265]. However, as described by Sawyer, stitches need a certain amount of overlap width in order to improve the adhesion strength of two joints beyond what the curing material is able to impart [240]. Sawyer describes a 2 inch width having 28% better joint failure strength, and a 3 inch width having 38% better joint failure strength. For this reason, for the SRI process, the geometries of the parts, as described in Appendix B, are altered so that they have wider joints to allow for more stitches to be threaded in. Specifically, the spar caps' widths are increased from being 2.12 inches

to 3.5 inches wide, the ribs' shear ties' portion where they contact the wingskin are increased from being 1 inch wide to being 3 inches wide, the ribs' spar joints are increased from being 1 inch in width and height to being 3 inches in width and height. This is done to maximize the increase in joint strength for attaching the major parts together. Similarly, the base of the rib shear ties and rib stiffeners where they attach to the ribs are increased from being 1.5 and 0.75 inches in width, respectively, to 2 inches in width. Their joints are not as critical as the major part to major part joints and so only the 2 inch overlap strength increase is deemed needed. The base legs of the hat stringers are increased from being 1 inch in width each to being 1.5 inches in width each, for a total overlap of 3 inches. Due to all this, wingboxes made with the SRI process are slightly heavier than for the other composite processes but are also slightly sturdier.

In order to be able to make If-Then rules to represent all this, how all the tasks are encoded into the precedence matrix must be described so that the If-Then precedence rules have something to rule on. The task number assignments are shown in Figure A.8, Figure A.9, and Figure A.10. How the four major part assembly task sets are handled is as follows, since they do not pertain to any individual parts and are completely dependent on the assembly sequence used, unlike all the other tasks. If there are no subassemblies or integrally fabricated parts, then each major part assembly task set corresponds to an assembly operation in the assembly sequence. If there are subassemblies or integrally fabricated parts, then because those must be made before their assembly operations are carried out, their respective operations take up the first few major assembly task sets. As an example, in a sequence with a subassembly and an integrally fabricated major part, where the subassembly occurs first in the assembly sequence, the first major part assembly task set corresponds with the subassembly and the second major part assembly task set corresponds with the integral part fabrication. The last two assembly task sets correspond to any remaining regular assembly operations in the order dictated by the assembly sequence. Tasks that are not in the list of available tasks have their respective rows and columns in the

precedence matrix zeroed out and cannot be assigned to workstations.

	Task Type / Part	Spar Stiffeners	Stringers	Rib Shear Ties & Spar Joints	Rib Stiffeners	Front Spar	Rear Spar	Upper Skin	Lower Skin	Ribs
HLU/Autoclave Fabrication	Hand Layup	1	5	9	13	17	21	25	29	33
	Autoclave Cure	2	6	10	14	18	22	26	30	34
	Hand Router Trim	3	7	11	15	19	23	27	31	35
	Ultrasonic NDI	4	8	12	16	20	24	28	32	36
ATL Fabrication	Automated Tape Layup	1	6	11	16	21	26	31	36	41
	Hot Drape Form	2	7	12	17	22	27	32	37	42
	Autoclave Cure	3	8	13	18	23	28	33	38	43
	Water Jet Trim	4	9	14	19	24	29	34	39	44
	Ultrasonic NDI	5	10	15	20	25	30	35	40	45
VARTM Fabrication	Hand Layup	1	5	9	13	17	21	25	29	33
	Oven Cure	2	6	10	14	18	22	26	30	34
	Water Jet Trim	3	7	11	15	19	23	27	31	35
	Ultrasonic NDI	4	8	12	16	20	24	28	32	36
SRI Fabrication	Hand Layup	1	6	11	16	21	26	31	36	41
	Stitching	2, 73	7, 74	12	17	22	27	32	37	42
	Oven Cure	3	8	13	18	23	28	33	38	43
	Water Jet Trim	4	9	14	19	24	29	34	39	44
	Ultrasonic NDI	5	10	15	20	25	30	35	40	45
CNC Machining Fabrication	CNC Mill					7	11	18	25	29
	Pre-Form		1					15	22	
	Stretch Form							16	23	
	Roll Form		2							
	After-Form		3					17	24	
	Dye Penetrant		4			8	12	19	26	30
	Anodize		5			9	13	20	27	31
	Chromate Spray		6			10	14	21	28	32

Figure A.8: Precedence Matrix Task List Numbering for Fabrication Tasks Given Manufacturing Process, Task Type, and Part Type

The If-Then rules used to perform the rule-based modeling codify all of the above changes to the precedence matrix that result from different sequences being selected and different parts being used for subassemblies and integral fabrication. All these rules for all five processes are listed out in Appendix C. Following these rules, given the assembly sequence, essentially defines the matrix.

A major assumption throughout all this is that, given the assembly line is modeled as a pulsed line for the line balancing, it is assumed the assemblies themselves can be pulsed. In essence, the assemblies are not forced to sit in monument fixtures from start to finish waiting for parts to be brought to them and attached. This is a limitation in older aircraft assembly lines because the technology did not exist yet back then to be able to pulse assemblies from station to station on their fixtures. It is a major reason why the more typical

	Part / Task Type	Front Spar Stiffeners	Rear Spar Stiffeners	Upper Stringers	Lower Stringers	Rib Minor Parts
HLU/Autoclave Minor Assembly	Fitup	37	40	43	46	49
	Drill	38	41	44	47	50
	Fasten	39	42	45	48	51
ATL Minor Assembly	Fitup	46	49	52	55	58
	Drill	47	50	53	56	59
	Fasten	48	51	54	57	60
VARTM Minor Assembly	Fitup	37	40	43	46	49
	Drill	38	41	44	47	50
	Fasten	39	42	45	48	51
SRI Minor Assembly	Fitup	46	49	52	55	58
	Drill	47	50	53	56	59
	Fasten	48	51	54	57	60
CNC Machining Minor Assembly	Fitup			33	36	
	Drill			34	37	
	Fasten			35	38	

Figure A.9: Precedence Matrix Task List Numbering for Minor Assembly Tasks Given Manufacturing Process, Task Type, and Part Type

	Task Set / Task Type	Major Assembly Task Set 1	Major Assembly Task Set 2	Major Assembly Task Set 3	Major Assembly Task Set 4
HLU/Autoclave Major Assembly	Fitup	52	55	58	61
	Drill	53	56	59	62
	Fasten	54	57	60	63
ATL Major Assembly	Fitup	61	64	67	70
	Drill	62	65	68	71
	Fasten	63	66	69	72
VARTM Major Assembly	Fitup	52	55	58	61
	Drill	53	56	59	62
	Fasten	54	57	60	63
SRI Major Assembly	Fitup	61	64	67	70
	Drill	62	65	68	71
	Fasten	63	66	69	72
CNC Machining Major Assembly	Fitup	39	42	45	48
	Drill	40	43	46	49
	Fasten	41	44	47	50

Figure A.10: Precedence Matrix Task List Numbering for Major Assembly Tasks Given Manufacturing Process, Task Type, and Task Set Number

ways to model ALB were not applicable to aerospace manufacturing, often requiring the use of discrete event simulations instead. However, such technology exists today with examples including the many types of Electroimpact assembly jigs and tooling that can hold the entire assembly from start to finish [231]. The Boeing Company themselves have a fair few patents on such pulsable assembly jigs and tooling for the wingbox, such as in their patent for a high rate pulsing wing assembly line tool, indicating that it is possible and can be done [266]. With such pulsable jigs and tooling, large aircraft assemblies can easily move from station to station like in the commonly modeled ALBPs. It is for these reasons that ALB and its associated modeling is assumed to be able to be done on aircraft in the first place.

The last topic of importance for the manufacturing processes is about their zoning constraints, starting with the assembly ones. Because neither the minor nor the major assembly tasks actually need any equipment and only require the correct tooling due to the assembly being manual in nature, minor and major assembly tasks can be placed in the same station. The tooling is just a structure that the subassemblies and parts sit on and, without any large pieces of equipment needed to operate on the parts for an assembly task, it is reasonable to believe that a relatively skilled assembly worker can carry out various types of different assembly tasks in the same station. This is further emphasized by the fact that the same types of tasks such as fastening and drilling are performed for both minor and major assemblies. Although minor and major assembly tasks in real life tend to not be placed together in the same station due to the high chances of them obstructing the working space, several interesting tradeoffs can be explored if they are allowed but not forced to be in the same stations. The most notable of these is trading off the high probability of causing bottlenecks and task congestion with the possible benefits of using less aisle space usage, which can lead to higher throughputs. Despite being allowed to be placed in the same station, different assembly tasks will likely need different sets of tooling. This is further discussed in section A.5.

Regarding the fabrication tasks and their zoning constraints, it was already established that different task types cannot be placed in the same station because they require different equipment types and that tasks of the same type can use the same equipment type and so be placed in the same station. In reality, the equipment for a particular task is often custom-tailored to that task and so is not compatible even with other tasks of the same task type that use the same equipment type. As an example, in real life, it is not necessarily true that an ATL machine that is used for stringers can also be used for spars; the ATL machines for each will have their own tolerances and capabilities that make them incompatible with each other. However, as is be made evident in section A.5, there is insufficient publicly available data to capture this for all the different part types. Therefore, for the purposes of being able to model the equipment and perform the line balancing, it is assumed that the equipment for each equipment type is highly capable and able to operate on all parts of the same task type is used. This allows for tasks of the same task type to be placed in the same station and eases zoning constraints such that the fabrication tasks do not all need their own stations.

A.4 Parametric Cost and Time Estimation

The fundamental information needed by SEER-MFG is how big a part is. This is primarily represented by its length, width, and thickness. The thickness in particular is a variable that is often difficult to obtain in early aircraft design and is why the sizing of the parts using the RADE framework is performed. It is only because of the availability of geometric information enabled by that framework, which is able to operate in early design, that SEER-MFG is able to be used at all and is why it was the focus of many past MInD works. Thus, all the different pieces of information obtained from SEER-MFG below are obtained in the context that geometric information is available as a result of the RADE framework's outputs. The length and width of a part are also very important parameters. However, all the parts taper to an extent because the wingbox itself tapers, meaning there is no single constant length or width. As a result, whenever part lengths or widths are required, the

average length and width is used. This is straightforward for the wingskins. For the spars, the average length is the same as the actual length while the average width is the average height of the spars combined with the non-thickness dimension of both their flanges. For the ribs, the average length is the average height of the ribs multiplied by the number of ribs to represent them being made in batches, while the average width is the average length of the ribs. For the minor parts, the average length is the same as the average length of their respective major part, and the average width is set such that the product of their average length and width is the same as the area occupied by the minor parts if they were all laid out completely flat with no flanges, shapes, or contour changes.

As described above, SEER-MFG, from now on referred to as just SEER, is able to output labor costs, tooling costs, and material costs. How it calculates material costs is it sums up the individual material costs, which include the primary, raw material used for the parts, fasteners used for the fastening tasks, and secondary materials such as bagging material. The raw material estimates are obtained by SEER estimating how much material is used throughout the course of a task and multiplying that weight by a cost-per-weight value. The used weight is the weight of the part resulting from the task multiplied by the material utilization factor, which is the ratio of how much raw material is needed to produce a certain amount of final product, otherwise known in aerospace industry as the buy-to-fly ratio. This ratio is 2.2 for the hand layup tasks, 1.5 for the automated tape layup tasks, 1.15 for the metallic pre-form tasks, and 1.05 for the metallic CNC milling tasks, based on previous MInD works. The cost-per-weight values are also obtained from previous MInD works and are $58.61 per lbs. for dry fiber for the VARTM and SRI hand layup tasks, $85 per lbs. for prepreg for the HLU/Autoclave hand layup task, $90 per lbs. for prepreg for the ATL automated tape layup task, $2.2 per lbs. for aluminum sheets for the metallic pre-form tasks, and $2.42 per lbs. for aluminum blocks for the CNC milling tasks.

How SEER estimates the weights differs based on the task being estimated. For the composite tasks, SEER requires the layup in terms of the number of 0°, ±45°, and 90 °

plies used. The Hypersizer program used in the RADE framework outputs a tailored layup where there are constant ply dropoffs down the length of every part, meaning it is difficult to put in an exact number of plies of a certain orientation into SEER. What is done instead to simplify the data entry is that the weights of each part, combined with all their non-thickness dimensions shown in Appendix B, are used to back out an average part thickness. Given that a 35/50/25 effective laminate layup schedule was used and that the thickness of each ply is 0.0055 inches, the number of plies in each orientation can be backed out. Thus, the average number of plies in each orientation for each part is inputted for the composite tasks, which SEER uses to estimate how much raw material was used. For the dry fibers, the material cost of the infused resin is also calculated in the oven cure task by determining the volume of resin needed, calculating the weight used given the chosen resin's density, and multiplying it by the resin's cost per pound. Cycom resin was used with a density of 0.04 lbs./in^3 and a cost of $13.9 per lbs. and a material utilization factor of 1.5.

For the metallic pre-form tasks, the sheets are roughly the size of the final part. Given that the metallic parts' thicknesses change, just like with the composite parts, the average thickness is used for the metallic tasks as well. The stringers are made in one batch and so estimated with just one task, so the combined size of all the stringers is used. For the CNC mill tasks, the specific machining operations must be described as well as how much material is removed in each operation in order to obtain estimates for the tasks. High speed roughing and finishing operations are used for the machining. More details on the specific operations are provided in Appendix D. SEER uses these machining operations along with how much material there was to start and how much remains at the end to estimate how much material ended up being used.

The fasteners are the next most significant contributor to material costs. Three types of fasteners are used: hi-lock bolts, regular lock bolts, and solid rivets. Solid rivets are used to connect all internal joints that do not contact the skin because the primary loads in those situations are shear loads that rivets are better able to handle and do so at a low price. Based

on previous MInD works, 0.25” diameter solid rivets are appropriate for the typical loads in the wingbox and are $2.25 per rivet. Lock bolts are used to connect almost all joints that involve the wingskin because those joints are susceptible to tension loads that lock bolts are more rated for. Based on previous MInD works, 0.375” diameter lock bolts are appropriate for the typical loads encountered and are $3.80 per lock bolt. The only joints involving the wingskin that do not use lock bolts are the joints involving the wingskin that are also involved with the last assembly operation that “closes out” the wingbox and so are likely to require the usage of access holes. hi-lock bolts are used for these joints because they have similar strength properties to lock bolts but are easier to install; rivets and lock bolts require easy access to both sides of the joining location whereas hi-lock bolts do not need as much access. To compensate for this, hi-lock bolts are much more expensive. hi-lock bolts are assumed to have roughly the same properties as lock bolts, and so 0.375” diameter hi-lock bolts are used. Based on comparisons using online fastener suppliers such as [267], hi-lock bolts are roughly 1.6 times more expensive than the same size lock bolts and so each 0.375” diameter hi-lock bolts costs about $6.00.

The number of fasteners used depends on the joint length between two parts. The joint length is the length for which two parts contact each other. As an example, the joint length of the front spar to the upper skin is the length that the spar’s upper cap contacts the upper skin, which is equal to the front spar’s length. And another example, the joint length between the front spar and its stiffeners is the length along which the spar stiffeners contact the spar, which is the sum of the lengths of all the front spar stiffeners. After the joint length is determined, the minimum edge distance must be accounted for so that fasteners do not sit too close to the edge of the part. The minimum edge distance to be structurally sound was determined in previous MInD works as being 2 times the fastener diameter and so, to account for both edges, a length amount equal to 4 times the fastener diameter is subtracted from the joint length to give the fastener joint length. The fastener joint length is used to determine the number of fasteners. Based on previous MInD works, a single row

of fasteners with a pitch of 6 times the fastener diameter is sufficient to meet the structural needs of the F-86, and so the number of fasteners used is equal to the fastener joint length divided by 6 times the fastener diameter. Combined with the determination of whether a solid rivet, lock bolt, or hi-lock bolt is used, as well as a material utilization factor of 1.15 for spare fasteners, this sets the material cost due to fasteners.

The bagging material costs are the last type of material cost and are simple in comparison to the other costs. They are dependent on the size of the bag used, itself dependent on the size of the part being cured. These secondary costs are rather insignificant compared to raw material and fastener costs.

The tooling costs are discussed next. SEER is able to estimate tooling costs by first applying a size factor to the task's part to estimate how large the tooling needs to be. It then applies a complexity factor to the tooling depending on how complex the tooling has to be, which is dependent on the types of parts the tooling needs to hold. SEER uses both of these factors to scale up how much time it takes to fabricate the tooling, for the size factor, and how long it takes to design it, for the complexity factor. This is multiplied by the default tooling design and fabrication wage rate of $100 an hour and the default material utilization factor of 5 to obtain the tooling cost. The actual un-scaled time and size of the tooling is dependent on what type of tooling is selected. SEER allows the user to either select the specific types of tooling to be used or to defer the selection to SEER's own internal database for each task type, which is called "Aero Default Tool Set." The size factor is set to be 1.25 times the part's length and width for all tasks to ensure adequately large tooling is used. The complexity of the tooling depends on the parts being held, the specific task being modeled, and the type of task and its context. Tasks where only singular parts are involved such as spars or wingskins tend to have lower complexity while ones dealing with larger numbers of parts such as stringers or ribs tend to have higher complexity. Major part assembly tasks, due to the usage of sophisticated pulsable jigs described in the previous section, have high complexity tooling. And autoclave and oven curing tasks have variable

complexity depending on how many tasks, and thus parts, are in the same curing station and have to be cured at the same time. The tooling complexity in regard to curing tasks is covered in greater detail in section A.5. The Aero Default Tool Set is used whenever it is available except for the assembly tasks, where it estimates exorbitantly high costs, and for the autoclave cure tasks, where it is unavailable. For the assembly tasks a user-defined, representative, and comprehensive collection of jigs and fixtures is chosen instead. Tool sets and tooling most representative of a mandrel are used for the autoclave cure tasks.

Every fabrication task is assumed to have its own set of tooling, even if it is not the only task in its station, because each task's tooling specifically accommodates the task's part and no others. It is thus assumed that the part in its tooling is rolled up to the equipment in the station it is in, where the equipment operates on it to carry out the part's task. A conservative assumption is made that all composite fabrication tasks use a generalized mandrel for their tooling, with the cost of a task's mandrel equal to the tooling cost for that part's curing task. Mandrels are large pieces of tooling that parts sit on to be cured for their curing task and, due to the size of the parts in aerospace industry, it is common to perform many fabrication tasks on the part while it is on its mandrel. This is because the parts must either be carted around the factory in an automated ground vehicle on a common piece of tooling or be loaded and unloaded onto their station-specific tooling by some other device, generally a crane. The latter essentially imposes an omnipresent crane loading and unloading task in all stations and so the common tooling option is preferable. A generalized mandrel is the only common piece of tooling that is able to withstand the curing process with the part during its cure task as well as the only one able to accommodate the part before and after its cured state, which is why it is used.

A similar assumption is used for metallic fabrication tasks where they all use the tooling and tooling costs from their parts' dye penetrant tasks. The dye penetrant task requires the most flexibility of all the metallic fabrication tasks since the tooling must be able to hold the parts before, during, and after they are dipped into dye penetrant tanks and must also hold

them properly for inspection. This flexibility is why it is most suitable to be the generalized tool set for all the metallic tasks. It also means that the most expensive tooling set is being used for all the metallic fabrication tasks because the complexity of the dye penetrant task's tooling is high due to it needing to be so flexible.

Not all assembly tasks have their own sets of tooling, unlike with fabrication tasks, because multiple assembly tasks in the same station do not necessarily mean there are multiple assemblies in the same station, unlike with fabrication tasks where multiple such tasks in the same station mean there are multiple different parts in the same station. As an example, if all the tasks in an assembly task set are in the same station, it means the same parts are being fitted up, drilled, and fastened; only one set of tooling is needed for all that because the tasks only focus on one assembly and sequentially flow through its different assembly states. Different sets of assembly tooling are only needed when there is more than one assembly and more than one assembly state in a station, such as when a minor assembly task set is in the same station as a major assembly task set. This is delved into in greater detail in section A.5.

The last cost item to go over is the labor cost. SEER estimates labor cost by first estimating a labor time and multiplying it by a labor rate. Based on past MInD works, a burdened labor rate of $200 per hour is used for both the fabrication and assembly tasks. A burdened rate means it accounts for the labor rates of all the people who indirectly support the worker such as the people in the engineering department, the HR department, etc. SEER splits its labor time into three different categories: direct, setup, and inspection and rework. Direct labor is the time spent actually performing the task in question, such as performing the hand layup in a hand layup task or trimming the part with a water jet in a water jet trim task. Setup labor is all the time preparing the part, the equipment, and the tooling to get it ready for direct labor. This includes the time spent cleaning and preparing the tools and tooling, loading and unloading the part, and working on secondary activities such as getting the bag ready or rotating the part or debulking it. Setup time is thus a significant

contributor to total labor time. Inspection and rework time is distinct from the NDI tasks because it refers to in-process inspection done by the workers as opposed to a dedicated NDI machine. SEER calculates it by taking the total labor time and applying percentages to it to obtain the inspection and rework times. Based on past MInD works, percentages of 5% for in-process and quality assurance inspection and 2% for in-process rework and rework are used for almost all tasks except for VARTM cure ones. VARTM cure tasks have percentages a 16% in-process inspection rate and 12% quality assurance inspection rate due to the process of infusion needing to be carefully monitored to proceed properly.

Although SEER is able to output the total labor time for each task, the total labor time is not necessarily the task time. This is because SEER's total labor time estimate assumes there is only one worker performing the task, an assumption which cannot be changed in SEER. The total labor time thus is only the task time if one worker is working on the task, which is a situation that almost never happens in real life aircraft manufacturing. Adding additional workers to a task reduces the task time, but not in a linear way because workers can get in each other's way and negate some of the benefits of having extra workers. According to Siedlak et al.'s work, the task time is 60% of the total labor time if 2 workers work on a task, 40% with 3 workers, and by extrapolation, 30% with 4 workers and 25% with 5 workers [29]. On top of this, adding workers to a task is only beneficial to the non-automated parts of a task. For a task like the ATL process's automated tape layup task, more workers will not help the tape layup machine lay down plies any more quickly. Workers thus only apply their labor time reduction potential to manual tasks and the non-direct labor components of automated tasks. The list of automated vs non-automated task types and the number of workers used for each in this work is shown in Figure A.11. The number of workers for each task type will not be changed or varied as a simplifying assumption to reduce the otherwise staggering number of line balances that would need to be done given the large number of added variables.

Although SEER calculates a labor cost, that labor cost is not actually used, only the

Number Workers \ Auto or Manual	Manual	Automated
2	Dye Penetrant, Chromate Spray	Automated Tape Layup, CNC Mill, Ultrasonic NDI, Water Jet Trim, Hot Drape Form, Stitching, Oven Cure, Autoclave Cure, Anodize, Stringer Pre-Form, Stringer Roll Form, Stringer After-Form
3	Oven Bag/Debag, Autoclave Bag/Debag, Skin Pre-Form, Skin After-Form, Manual Trim	Skin Stretch Form
4	Hand Layup	
5	Minor & Major Assembly	

Figure A.11: List of Task Types, Their Number of Workers, and Whether They Are Automated or Manual

task time resulting from scaling the total labor time given the number of workers. This is because SEER's labor cost estimate does not account for any inefficiencies or idle time in a workstation. The task time itself is used only for line balancing and station duplication purposes. The actual labor cost used for each instance of a station per wingbox produced is the product of the cycle time, the labor rate, and the number of workers at that station. This labor cost calculation works out to be the same as that shown in Equation 5.8. This is done because it is not realistic to expect a worker to leave during their idle time and not get paid and then come back during the next cycle. As discussed by many authors such as Kazemi and Sedighi, the time a worker spends working at their workstation is equal to the cycle time, regardless of the actual station time [169]. Workers are generally not allowed to leave to another workstation to carry out other tasks during the idle time in their own workstation because they will not be properly trained, because their help is not needed, or because their help could possibly result in further mishaps, especially in the highly specialized aerospace industry. Workers are thus paid even during their labor time, and SEER's labor cost is not actually used, only its labor time estimations that are used to obtain the task times.

A noticeable quirk from Figure A.11 is that the oven and autoclave cure tasks are split

into a bagging/ debagging portion and a cure portion. This is because the cure portion is entirely automated and so only requires two workers who just have to watch the cure process being carried out; they do not even have to operate much, as opposed to workers in other automated tasks like in the automated tape layup task, who will likely need to operate the machine throughout its tape laying. Additionally, the task time of this cure portion is entirely unaffected by the number of workers since it will always be 6 hours for the autoclave cure and 148 minutes for the oven cure. Meanwhile, the bagging and debagging portion requires a lot of manual labor and so its task time is affected by the number of workers, hence splitting the two up.

SEER also has a mechanization setting to determine how automated the tasks are, which reduces the total labor time. The tasks listed as being automated in Figure A.11 thus have a high mechanization setting while the other ones have a nominal or low setting due to being manual in nature. The assembly tests especially have a very low mechanization setting because they are entirely manual.

Of particular note is that SEER does not estimate the travel time between stations because that is completely dependent on the physical layout of the stations in the factory, which SEER does not simulate. This topic is currently beyond the scope of this work and so the simplifying assumption is made that the amount of time spent traveling between stations is negligible.

The fitup, drill, and fasten assembly tasks require a fair amount of detail to model. The planform size of the assembly is equal to the dimensions of the major part for minor-part-to-major-part tasks and is equal to the planform dimensions of the wingbox for all major part assembly tasks. One of the most important pieces of information is the joint length, which was described above. Another is the joint width, needed to calculate the joint area for the fitup and fasten tasks. A joint width of 1 inch is used for all the internal, non-skin contacting joints because they are not as critical as the joints that do contact the skin, which have a joint width of 1.5 inch, as used in previous MInD works. For the minor part

fitup tasks, a detail location operation is used to situate the minor parts with their major parts. For the major part fitup tasks, a subassembly location operation is used to situate the major parts with the other major parts. Additionally, fuel sealing is applied to seal the gaps in the parts to prevent fuel leakage and faye sealing and liquid shim are applied to fill in any gaps between the parts. For the drilling task, 2 pilots holes are first drilled for every joint being joined except for the rib shear tie-to-skin joint. Full sized holes are then drilled through all the pilot holes and additional holes are drilled until there is enough for the number of fasteners needed for that particular drilling task. All holes in contact with the skin are countersunk so that the fasteners will eventually sit flush with the wingskin and afterwards reamed to final to tighten the tolerances. It is assumed that one-up assembly is too advanced and automated to be used and so, when the drilling is finished, the assembly is disassembled and the holes deburred before they are reassembled. The drilling task is where the differences between material systems show up as it takes longer to prepare for and drill through one material as compared to another. The material being operated in the must thus be declared for the drilling task. And for the fastening task, all the fasteners are installed completely wet to protect against corrosion, with rivets taking two-thirds of a minute each, lock bolts taking half a minute each, and hi-lock bolts taking about a minute each.

The assembly tasks are where accessibility plays a significant role. All the minor assembly tasks have high accessibility because both sides of the joining locations can be easily accessed. All major assembly tasks involving only spars and ribs have nominal accessibility because it is simpler to duck and weave in between the spars and ribs to access the joining locations. All major assembly tasks involving any of the wingskins have low accessibility because the wingskins being such large parts can make it awkward to install the other major parts. And it is in the last major part assembly task set that the assembly's accessibility from Conjecture 1.2 appears in the SEER modeling. If only the ribs need the access holes to access both sides of their joining locations, the accessibility is quite low,

which is the highest accessibility level for the last major part assembly task set. If an additional set of joining locations on the spar needs access holes to access both sides, the accessibility level is very low. And if more than one additional set of spar joining locations needs the access holes, then the accessibility level is extremely low. Each successively lower of accessibility increases the total labor time by roughly a factor of 0.33 times.

In regard to assembly tooling complexity, the tooling for the minor assembly tasks is nominal due to it not needing to be too large or particularly complex. As mentioned above with the pulsable assembly jigs, the tooling for the major assembly tasks is complex due to it needing to support an entire wingbox assembly and be able to move around the factory floor. And the tooling for the major assembly tasks composed of only one spar and the ribs is very complex because that subassembly does not make a fully closed shape yet and is very weak torsionally, making it unstable and so requiring additional supports and complexity.

The effects of integral fabrication and subassemblies are captured in various ways in the SEER modeling. Subassemblies are the most straightforward as they just entail a larger number of holes to be drilled and fasteners installed in the associated major assembly task set due to more parts being assembled at the same time. For co-bonding minor parts, an extra step is added in the minor part's fitup task where an adhesive film is placed between the minor part and its major part's joints to aid their adhesion. Extra pleats in the bag then need to be made in the major part's autoclave cure task, equal to the number of individual minor parts being co-bonded, e.g., the same number of pleats as there are actual bottom stringers for the bottom skin. Since this can result in an excessive number of pleats for the ribs and their minor parts, the ribs are arranged such that the pleats for a single rib's rib stiffeners also suffice for all the other ribs. Additionally, the pleats for the shear ties are done such that each individual rib only needs two pleats for its shear ties.

The VARTM minor part co-cure is simple to model since the only difference after the major part and its minor parts are fitted up is that the minor parts are trimmed before they

are both oven cured. The SRI minor part co-cure requires the minor parts be stitched to the major part after they are fitted up. The minor parts are first tack stitched to attach them to the major part before fill stitching is done to fill up the rest of the stitch space. The stitching length per row is equal to the joint length between the minor and major parts. The stitching width is equal to the joint width of the minor part specifically for the SRI process as described in the previous section, which is 2 inches for spar stiffeners, rib stiffeners, and rib shear ties, and 3 inches for stringers, which accounts for both of its 1.5 inch wide flanges. Based on NASA's advanced stitching machine, it is assumed for the stitching task that the stitching machine has 4 stitch heads that each stitch at 800 stitches a minute, with 8 stitches per inch and a row spacing of 0.2 inches, meaning 5 rows per inch of stitch width [268]. After they are stitched together, the minor and major parts are oven cured together in the same way as with the VARTM oven co-cure.

The SRI major part co-cure requires the major parts to be stitched together in the same manner as the minor parts being stitched to the major parts after the major parts are fitted up, which itself occurs after the major parts are stitched to their minor parts. Major assembly fitup tasks typically involve fuel sealing, faye sealing, and liquid shim. However, as the infused resin afterwards will perform all these roles by itself, major part co-cure fitup tasks do not involve these operations. The differences between major-to-major and major-to-minor part stitching primarily lie in the stitch widths being 3 inches between all of the major parts. The major co-cure parts are oven cured in the same way as the minor oven co-cure process except that the part size is updated to reflect the much larger integrally fabricated part and that the major parts are trimmed as well. The proceeding water jet trim and ultrasonic ndi tasks have a similarly updated part size.

Everything described in this section is implementable in the SEER program and is how the processes are modeled to obtain the task times and labor, tooling, and material costs. SEER is, however, unable to account for equipment cost or equipment size, which must be done as an add-on. This is covered in the next section as well as how the space required for

each station is actually estimated.

A.5 Equipment Costs and Space Modeling

SEER is able to provide tooling costs, material costs, and labor costs, the latter via providing labor times. It is not, however, able to provide equipment costs, which are needed for the APUC metric, nor the space taken up by said equipment, nor the space taken up by the workstation the equipment is in, both of which are needed for station duplication and estimation of the maximum feasible throughput. Other means must thus be used to obtain those values. The equipment costs and their spatial requirements are intricately linked and a major part of the overall station space used and so are focused on first.

The actual sets of equipment needed to operate on each part in each task for each process in real life are very specifically tailored such that certain tolerances, operational speeds, and other requirements are met. This means there are numerous factors and variables that play into exactly how much each piece of equipment costs as well as how large it is. These relations unfortunately are almost always proprietary in nature and, as such, public sources for the exact relations are basically nonexistent. This means exact, directly implementable-to-real-life equipment costs and space requirements are nigh impossible to obtain. Despite that, it is not desirable to simply make up that data out of thin air and then use it to demonstrate this work's tradeoff capabilities. This is because it brings up questions of validity regarding those tradeoff capabilities, which are this work's main focus. Therefore, equipment cost and space data will be estimated from as many publicly available sources as possible to show that the data has a basis in reality and that, despite the data not being directly translatable to real life, it is still representative, meaning any tradeoff capabilities demonstrated using such data is valid.

To be able to estimate equipment costs and space, a few key assumptions are required. The first was already mentioned above and is that each task type's equipment set is capable and flexible enough to be able to operate on all the parts equally. This means that, within a

specified task type, there are no equipment-based zoning constraints. This leads directly to the second assumption, which is that the equipment's cost and size scale only as a function of the size of the part. Other factors such as specific tolerances and part type are not accounted for since relations on how they affect cost are prohibitively difficult to obtain. And the third assumption is that the equipment at each station is sized to be able to accommodate the task with the largest part in that station. Although this means that the equipment will be over-sized and not cost-optimized for the smaller parts, it is a line balancing tradeoff that must be made to be able to operate on the largest part at all.

When sizing the equipment, the part's average length and width cannot be used like is done in SEER. Doing so is likely to result in some sections of the equipment being oversized and some being undersized for the part getting operated on. Instead, the lowest common denominator in the form of the part's maximum length and width are used. This is most noticeable for the width dimension because of the strong taper of the wingbox causing its root width to be double, sometimes almost triple, its tip width. This means all the widths for the equipment sizing will be done with each part's largest widths, which tends to lie at the root for all the parts. This results in the additional benefit of the equipment being sized more conservatively than liberally, which is desired when the equipment data is only representative instead of being exactly realistic. From this point on for the rest of this section, the length and width terms are assumed to be the maximum of the part's lengths and widths. However, it necessitates further explanation of what is referred to by widths and lengths because it is not necessarily the same between the SEER modeling and equipment sizing.

For wingskins, spars, and ribs, their definitions remain the same as before except that the largest widths are used. For the ribs this means a width equal to the length of the longest rib and a length equal to the average height of the ribs multiplied by the number of ribs. For the spars this means their widths are their heights at the root combined with their flange lengths while their lengths remain the same. The size determination of part types

that have many individual parts, such as all the minor parts, for the purpose of equipment sizing is slightly less standard. When the minor parts are manufactured on their respective equipment sets, they are almost always laid flat or are supported by fixtures or tooling. In this laid-flat or supported configuration, they are assumed to be approximately the same size as their major parts: front spar stiffeners have the same width and length as the front spars, rear spar stiffeners the same as rear spars, upper and lower stringers the same as upper and lower wingskins, and rib shear ties, spar joints, and stiffeners combined the same length and width as ribs. When both the front and rear spar stiffeners are fabricated together, their length is approximately the same as the longest spar's length, typically the front spar, and their width is approximately the same as double the widest spar's width, also typically the front spar. When both the stringers are fabricated together, their length is the same as the longest wingskin length and their width is double the widest wingskin width. When the rib shear ties and spar joints are not made with the rib stiffeners, both of their lengths are half the rib's length and their widths are the same as the rib's width. Equipment-sizing part dimensions as a result are overall larger than their SEER modeling counterparts.

Based on the available data, many equipment sets determine their size and cost not on the size of the part but on a platform or table that the part sits on or is supported on. These platforms or tables tend to be rectangular in shape, whereas most tapered and swept wingboxes are not. This causes it to be impractical to size the equipment based on just the length and width descriptions described above; it is likely to result in a part of the wingbox sticking beyond the table or support. The table or platform or support must therefore be sized up to fit an area larger than the wingbox. Based on tests examining various wingbox shapes, it was determined that the equipment should be able to accommodate lengths of 1.25 times the part length and widths of 1.25 times the part width, resulting in the equipment support or platform being able to handle any wingbox within a reasonable range of aspect ratios, taper ratios, and sweep angles. These parts in question are assumed to be the largest in the station the equipment is in.

The terminologies and assumptions just explained are sufficient to determine the equipment cost as a function of the part's size. However, a few more assumptions and clarifications must be mentioned before the equipment's size can be determined. The first is that information on equipment spatial requirements and its scaling is even more difficult to find than information on how its cost scales. This means that a not insignificant amount of estimation on the space required is dependent on what can be ascertained from pictures and key details in them, such as the height of OSHA-compliant handrails being used to scale the rest of the dimensions of the equipment set. The second is that the handling space from Equation 5.1, which is space for the workers to move around in within the station to operate the equipment and load and unload the parts, when it is not accounted for with the equipment itself using the picture scaling described above is accounted for by adding 3-6 feet of clearance on all sides around the equipment, based on Ventura's lecture notes [269]. The specific amount of clearance space used is dependent on the size of the equipment. And the third is that the spatial requirements cannot be accounted for by directly operating on the area; each equipment set is different, with some having a set length but a variable width or vice versa. Because of this length and width requirements are accounted separately according to the different needs of each equipment set, with the area only calculated as the very last step, of which the equipment space estimation is not the actual last step–the queue and aisle space calculations are the last step.

The costs and space requirements of all the different sets of equipment for all the tasks modeled in SEER are shown in Appendix E. Included with the equipment space requirements are the handling and loading/unloading space requirements as well. As not all of the cost data sources are from the same year or even sometimes the same currency, the final costs are all updated to January 2022 US Dollars, though the ones presented are from their original source and year.

The autoclaves and ovens for the autoclave and oven cure tasks are discussed here instead of in Appendix E because they are significantly more complex than the other equip-

ment sets. This is because not only does their cost depend on two different variables, length and diameter for the autoclave and length and width for the oven, but those variables change as a function of all the parts in the same station, not just a function of the largest part. Additionally, how the cost changes as a function of the diameter is different from the change as a function of length for the autoclave because autoclaves with larger diameters require thicker walls to handle the pressure, further increasing their cost and introducing even more complexity. The raw cost and size data used to generate the cost expressions for both the autoclaves and ovens are shown in Figure A.12 and Figure A.13. The data for the autoclave is sourced from Heckwolf [270], Goel [271], and Hagnell and Akermo [272], while the data for the oven is sourced from previous MInD works as well as the online used equipment market. Based on experience from comparing the costs of new equipment and used equipment from the online market, used equipment is roughly 4-5 times cheaper than new equipment, which was used to scale up the cost of the used equipment as though it were new.

The length and diameter of the autoclave and the internal volume of the oven are determined by the tasks, and therefore the parts, in the same curing stations. The autoclave is the more complex of the two. Based on a picture and description of the large autoclave at Boeing's Composite Wing Center in Seattle in Gates's article [273], which is stated to be around 28 feet in diameter and 110 feet in length, the sized autoclave is assumed to be able to fit two F-86 wingboxes down its length and have a large enough diameter to have another row of two F-86 wingboxes stacked on top of the first. As the wingbox's length and width are about the same as its wingskin's, this means an autoclave is assumed to be able to hold four wingskin-sized parts. Comparing the relative size of the 777x lower skin with the inner diameter of the autoclave in Gates's article, and using that relative size to scale the F-86 experimental platform's autoclave, it is estimated that, with only one row of wingskins, the diameter of the autoclave is roughly 1.5 times that of the width of the wingskin. If two stacked rows of wingskin were placed in the autoclave, the tooling needed

Length (ft)	Diameter (ft)	Cost (Thousands 2022 US Dollars)
10.0	10.0	737
20.0	10.0	1,169
40.0	10.0	1,601
10.0	12.0	974
20.0	12.0	1,770
40.0	12.0	2,567
8.2	3.9	778
16.4	5.9	1,166
32.8	9.8	2,916
19.7	16.4	3,499
75.5	23.0	12,636
40.0	15.0	4,050
20.0	3.0	810
20.0	4.0	972
20.0	6.0	1,215
10.0	3.0	972
10.0	4.0	1,134
10.0	6.0	1,296

Figure A.12: Data on Autoclave Cost as a Function of Diameter and Length, Adapted from Heckwolf [270], Goel [271], and Hagnell and Akermo [272]

Internal Volume (ft^3)	Cost (Thousands 2022 US Dollars)
162	47.5
4320	1,400.0
3500	1,180.0

Figure A.13: Data on Oven Cost as a Function of Internal Volume, Obtained from Previous MInD Works and Online Used Equipment Market, Cost Adjusted as Though New

to suspend one set of wingskins over the other would take up more space and is assumed to force the diameter to be 2 times that of the width of the wingskin. In this situation the complexity of the cure task's tooling is increased to be high as opposed to nominal due to the increased complexity of stacked tooling. This is all used to estimate the autoclave's internal diameter and length and from there estimate its cost given the number of parts cured in the autoclave simultaneously.

Based on the aforementioned assumptions, the autoclave is essentially composed of 4 wingskin "units," with each unit capable of holding a wingskin-sized part. Unit 1 is the one closest to the door, unit 2 is behind unit 1, unit 3 is above unit 1, and unit 4 is above unit 2. If two units or less are used, the diameter is equal to 1.5 times the width of the widest unit among the 2 wingskin units, and its length equal to the length of the longest part in the first wingskin unit and the longest part in the second wingskin unit, if any. The width of the units is determined via the sum of the widths of the parts in that unit. If 3 units or more are used, the diameter is equal to 2 times the width of the widest unit, while the length is determined the same way as before. In cases where more than 4 wingskin units are needed, a second autoclave in the station is required. The second autoclave is sized in the same way as the first in the sense of using wingskin units to fill it up. The upper wingskin, lower wingskin, ribs, upper stringers, and lower stringers each take up 1 wingskin unit of space and, when filling up wingskin units, are assigned first. The front spar, rear spar, front spar stiffeners, and rear spar stiffeners each take up 0.25 wingskin units of space. This means they can all fit in one wingskin unit when laid down flat due to having much lower widths than the wingskins. This group of parts is assigned wingskin units after the previous group. The rib shear ties and spar joints and the rib stiffeners each take up 0.5 wingskin units of space and are assigned last. There cannot be a non-integer number of wingskin units and so the number of wingskin units is rounded up after each group is finished being assigned.

Due to the assumption of how parts can be stacked and inserted, no more than 2 autoclaves are ever needed in a single station instance, a situation that itself only arises if all the

cure tasks are also in the same station. After the initial diameter and length is determined from filling up the wingskin units, the final diameter and length for the cost is calculated. The initial diameter already accounts for space to move around and handle the parts and so is also the final diameter. The initial length only accounts for the parts being in contact end-to-end and does not include any space for handling. As such, the initial length is multiplied by a factor of 1.25 to account for handling and positioning space inside the autoclave. The final cost $C_{autoclave}$ for each autoclave given a final length l and final diameter d using the data in Figure A.12 was obtained by finding the best fitting regression that includes both the diameter and length and is as follows, with dimension units in feet:

$$C_{autoclave} = 251311 + 2966.1 * d^2 + 5902.2 * d * l \tag{A.1}$$

The ovens are filled up via the wingskin units method in the same was as it was done with the autoclaves. However, as ovens do not have curved walls and are rectangular, they use an internal width as opposed to a diameter and so also have a lower internal width penalty in general. If two units or less are used, the internal width is equal to 1.25 times the width of the widest unit among the 2 wingskin units, and if 3 units or more are used, the internal width is equal to 1.75 times the width of the widest unit. If more than 4 wingskin units are used, the wingskin unit accounting is still done in the same way as with the autoclave, but instead of having a second oven, the second internal width is added with the first to have a total internal width. The internal length is determined in the same way as for the autoclaves. The internal height, based on the pictures of the online used equipment market ovens, is estimated to be 13 feet. The cost-determining final height and width are the same as the initial ones, while the initial length is multiplied by a factor of 1.25 to obtain the final internal length in the same way as the autoclave's. The internal volume is then calculated as the product of the final internal height, width, and length. A linear relationship between the internal volume and cost is assumed, resulting in the following final oven cost C_{oven} based on the data from Figure A.13 and given an internal length l,

internal width w, and assumed 13 feet height :

$$C_{oven} = -333.825 + 329.309 * l * w * 13 \tag{A.2}$$

The total equipment, handling, and loading space for the autoclave is based on the estimated dimensions of an ASC autoclave and is dependent on the final diameters and lengths [274]. Based on the pictures, the door is 3 feet in front of the edge of the usable space while the back is 6 feet behind the edge of the usable space. Behind the back of the autoclave are its motors, which extend back another 9 feet. If there is another autoclave, it is placed adjacent to the first autoclave. The parts are loaded into the autoclave directly from the aisle, so no loading or unloading space is accounted for. 5 feet of clearance is added in all directions to account for handling space. The preliminary length and width space requirements $L_{prelim-autoclave}$ and $W_{prelim-autoclave}$ for the autoclave given the autoclave length l, first autoclave diameter d_1, n total number of autoclaves in the station, and second autoclave diameter d_2 if it exists, are shown below. The lengths of the autoclaves are assumed to be the same. The 5 feet of clearance on all sides for the second autoclave only exists if the second autoclave itself exists. These preliminary length and space requirements do not account for the queue and aisle space yet and so are not the final space requirements for the station.

$$L_{prelim-autoclave} = l + 28 \tag{A.3}$$

$$W_{prelim-autoclave} = d_1 + 10 + (n-1) * (d_2 + 10) \tag{A.4}$$

The total equipment, handling, and loading space for the oven is based on the schematics of an ITS oven similar to one of the ovens in the online used equipment market [275]. Based on the schematic, the back of the oven is 8 feet beyond its usable internal length and the sides of the oven extend 6 feet beyond its usable internal width on both sides. 5

feet of clearance on all sides is added for handling space. Like the autoclave, the oven is loaded and unloaded directly from the aisleway and so no loading and unloading space is needed. Its preliminary length and space requirements $L_{prelim-oven}$ and $W_{prelim-oven}$ given its internal length l and internal width w are shown below, represented in feet:

$$L_{prelim-oven} = l + 18 \tag{A.5}$$

$$W_{prelim-oven} = w + 22 \tag{A.6}$$

The oven and curing tasks are also unique in that they are essentially composed of two different types of activities: the actual curing task, and the setup for that curing task that includes the distinct activities of bagging and debagging the parts. These two activities are relatively independent from each other, with the curing activity always taking the same amount of time depending on what process is used and the bagging and debagging total labor times being variable. Due to the autoclaves and ovens being some of the single most expensive pieces of equipment in the entire factory, it is not desired to have to duplicate them and their station if it is the bagging and debagging activities that are making the curing task station a bottleneck station. If it is the bagging and debagging portion causing the bottleneck, it is better to only duplicate that part of the station instead of the whole station, including the expensive equipment. For this reason, all stations with curing tasks in them have two distinct effective station times and two distinct number of duplicates: an effective station time composed of the task times of all the bagging and debagging activities and the number of duplicates of those combined bagging and debagging activities, and an effective station time composed of the curing itself and the number of duplicates of either the autoclave or oven. This means the effective station time for a station with curing tasks is the greater of either the curing effective station time, or the bagging/debagging effective station time. If the station with curing tasks is subsequently identified as a bottleneck

station, the greater of the two effective station times has its number of instances increased, which will also increase its area usage. The area usage of just the ovens and autoclaves was already discussed above. The area usage of the bagging and debagging activities is discussed below.

Because the bagging and debagging activities still occur in the same station as the curing, the two must be located right next to each other. As the bagging and debagging activities result in the same stackup of parts that will eventually end up in the oven or autoclave, it makes sense that they will be similarly stacked up for the bagging and debagging. As a result, the length of the bagging/debagging area is the same as the final autoclave length or the oven's internal length, while the width of the bagging/debagging area is 1.5 times that of the widest part. The extra 0.5 factor is for handling space. Parts that would occupy wingskin units 5 and above that would thus require a second autoclave or stack of parts in the oven are stored in the part queue area, hence why the width is only based on the largest part's width. This is summarized in the equations below, where l is either the final autoclave length or the internal diameter and w in this case is the widest part's width. The preliminary dimension equations are the same for both autoclave and oven cure tasks.

$$L_{prelim-bag/debag} = l \tag{A.7}$$

$$W_{prelim-bag/debag} = 1.5w \tag{A.8}$$

To calculate the task times for the bagging and debagging activities, the total labor time of all the tasks in the curing station are examined. If it is an autoclave cure task, those total labor times for each task are for bagging and debagging activities because SEER does not account for autoclave cure times in its estimations. The task time is then calculated by scaling those total labor times based on the worker factor shown back with Figure A.11. If it is an oven cure task, 148 minutes of oven cure time is subtracted from each task's total

labor time because SEER does account for oven cure. The task times for these tasks are then obtained by scaling the total labor times according to the number of workers and the station time subsequently obtained by summing all the tasks' task times.

With the most complicated equipment set, its cost, and its equipment, loading and unloading, and handling space covered, the last topic to cover for cost and space estimation is the space required for aisles and part queues. As aisle space considerations determine the overall layout type desired in the factory and influence how the part queues are arranged, they are covered next. Aisle space is heavily dependent on how the factory is laid out. However, the factory layout problem is a completely different problem type from the ALBP and so trying to answer it is out of scope of this work. A compromise is made in that a relatively modular factory layout is envisioned and is composed of "layout units" of four stations positioned with their aisles and aisle space as shown in Figure A.14. These layout units can be rotated and placed in whichever way so long as the aisles of different layout units can connect with one another. The configuration itself is based on examining and taking inspiration from how most aircraft factories are laid out.

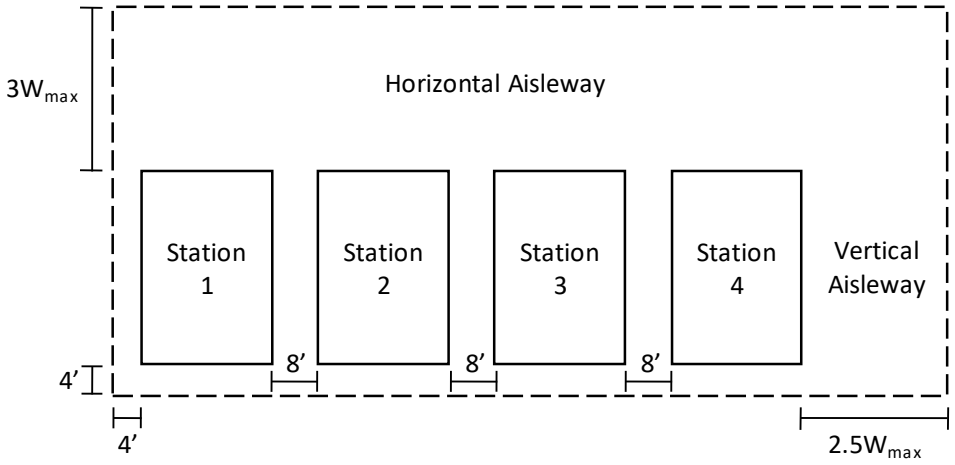


Figure A.14: Factory Modular Layout Unit with 4 Generic Stations, Aisleways, Dimensions. W_{max} is Width of Widest Part, Dashed Lines are Boundaries of Layout Unit. Station Relative Dimensions Notional and Not Necessarily Representative

The boundaries represent the walls of the factory or edges of other layout units. 8 feet

of clearance is provided for workers to walk between stations and accounts for Ventura's suggested worker aisle allowances should there be an opening door [269]. 4 feet of clearance is provided for workers to walk between the station and the edge of the layout unit; combined with an adjacent unit's 4 feet of clearance, it creates a total of 8 feet of clearance, the same as the clearance between stations. The aisleways are as wide as they are to allow the parts to be transported down their length while on their tooling, which itself is on top of automated ground vehicles. In particular, it is assumed the automated ground vehicles can move perfectly sideways, allowing parts to be loaded and unloaded from their stations by moving in their width direction. This significantly reduces the required size of the horizontal aisleways. Due to this, inside the stations themselves, the parts' long sides are not necessarily aligned with the stations' long sides depicted in Figure A.14. A factor of 3 times the width of the widest part is used both to accommodate the widest part and to allow workers to safely walk alongside the automated ground vehicle. Additionally, Ventura suggests 12 feet wide aisles to allow 3 ton tractors and forklifts to be able to move through; a factor of 2.5 and 3 times the widest width easily allows for this because it is not expected that the widest part width will fall significantly below 6 feet or, if it does, that the part is still heavy enough to need a 3 ton tractor. The horizontal aisleway specifically is larger than the vertical one to account for the extra space needed to turn the corner from the horizontal to the vertical aisleway.

The usage of this layout imposes additional aisle space needs on every station's length and width requirements. Specifically, assuming each layout unit is composed of four stations, each station's final width and length W_{final} and L_{final} is composed of its semi-final width and length w_{sfinal} and l_{sfinal} with additional add-ons to each, shown below. W_{max} is the width of the widest part. The semi-final widths and lengths are the preliminary widths and lengths with extra space accounting for part queue space. The preliminary widths and lengths are the widths and lengths accounting for the equipment space, handling space, and loading and unloading space, discussed above. All these dimensions are assumed to be in

feet.

$$L_{final} = l_{sfinal} + 4 + 3W_{max} \tag{A.9}$$

$$W_{final} = w_{sfinal} + 7 + \frac{5}{8}W_{max} \tag{A.10}$$

Part queues are the last type of space that must be considered. They exist because there is only one set of workers in each station and they can only work on one task at a time. Similarly, the equipment in each station can only work on one task and so one part, or part set for minor parts with many individual parts, at a time as well. All the other parts must be stored in a queue within the station and wait for their turn to be operated on. The part arrangement that optimizes space usage is dependent on the station's preliminary dimensions, which are the dimensions that already account for the equipment space, handling space, and loading and unloading space. In general, because of the addition of aisle allowances as the last step, the least amount of area used is obtained by arranging the parts in the queue such that the semifinal length and width are as close to each other as possible.

When arranging the parts and accounting for their queue space, only one of their dimensions is added to the semifinal dimensions. An example visualization is shown in Figure A.15 to help explain why.

The red dashed line is the area encapsulated by the parts in the queue and is essentially the desired area to add to the preliminary area of the station. Assume that only parts 2 and 3 in Figure A.16 are present, that part 4 is not there, and that the vertical direction represents width and the horizontal direction represents length. If part 5 is added to its Option 2 location, the area that would be encapsulated by the red dashed line increases only due to a change in length, not in width. Similarly, if part 5 is added to its Option 1 location, the encapsulated area would change only due to a change in its width, not its length. In the same way, when adding parts to the queue, either the part's width or length

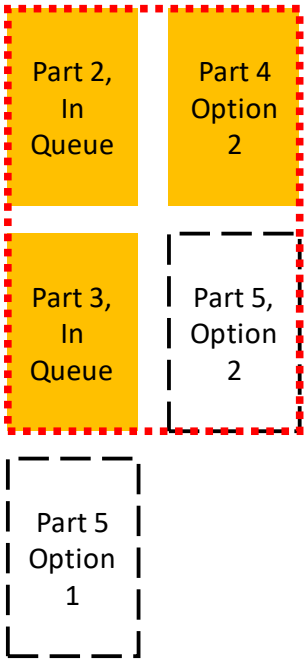


Figure A.15: Example Visualization of How Part Arrangement in Queue Can Affect Area Used To Store Queued Parts, Assume Vertical Direction is Width and Horizontal Direction is Length

dimension is added to the semifinal dimension, not both. Additionally, it is possible to add a part to the queue and not need to add any additional space requirements at all to the semifinal dimensions. If part 4 were present and part 5 placed in its Option 2 location, there would be no change in the area encapsulated by the queued parts. As such, depending on how the part queues are arranged to minimize space, it is possible for there to be situations where the semifinal dimensions only change for every second part added to the queue. The dimensions actually added, l_{queue} or w_{queue}, are equal to either the length l_{part} or width w_{part} of the part being added to the queue multiplied by a 1.25 factor to account for the tooling the part is on and an additional 3 feet of clearance on all sides to account for space for workers to walk between the parts. This is shown below, where l_{sfinal} is the semi-final length and w_{sfinal} is the semi-final width, PQ are the parts in the part queue that actually affect the queue space requirements and is not necessarily all of the parts, and l_{prelim} and w_{prelim} are the preliminary lengths and widths that account for equipment, handling, and

loading/unloading space. Since parts generally only add part queue space in one dimension, the summation portion of either Equation A.11 or Equation A.12 will generally be zero for most task types and the stations they are in. If only one of either Equation A.13 or Equation A.14 appears, the other can be assumed to be zero.

$$l_{sfinal} = l_{prelim} + \sum_{p=1}^{PQ} l_{queue_p} \tag{A.11}$$

$$w_{sfinal} = w_{prelim} + \sum_{p=1}^{PQ} w_{queue_p} \tag{A.12}$$

$$l_{queue} = 1.25 l_{part} + 6 \tag{A.13}$$

$$w_{queue} = 1.25 w_{part} + 6 \tag{A.14}$$

The parts in the queue are assigned part queue space in order of largest part to smallest part to minimize space usage. How the part queues are handled for each task type is covered in Appendix E along with each task type's respective equipment cost, equipment space, handling space, and loading/unloading space. A station's first task's part is not placed in the queue but rather directly loaded onto the equipment and so queue space for it is accounted for by including loading and unloading space. For both the oven and the autoclave, every other part in the queue adds a queue width w_{queue} listed above in Equation A.14 to the bag/debag preliminary width and turning it into the bag/debag semi-final width. The bag/debag preliminary length thus turns into the semi-final length by default.

Once the equipment space, handling space, loading/unloading space, and part queue space for a station are accounted for, the aisle space allowances are added to the respective dimensions. The station's final length and width are then multiplied to obtain its final area required given the tasks it must carry out. Whenever the station is duplicated, this

final area is duplicated as well. For the curing tasks and stations, the aisle allowances are added to the curing activity's preliminary length and width, which then become the final length and width. Part queues are not added to the curing activity's space, only the bagging/unbagging activity's preliminary dimensions to make them semifinal. The addition of the aisle allowances makes them final. The final area for the curing activity is added with the final area for the bagging and debagging activity, which becomes the area required for that station. When duplication is done, either the curing activity final area is duplicated or the bagging/debagging activity final area is duplicated, depending on which activity is the bottleneck, not both. This finishes spatial considerations for fabrication tasks and stations, with all the above steps summarized below. An example mock-up visualization of how a station looks given its space allocation is shown in Figure A.16.

1. Given cost relations, calculate equipment cost as a function of part's dimensions

2. Equipment space, handling space, and loading/unloading space relations are represented in terms of length and width and are functions of part's max length and max width. Given part dimensions, determine length and width needed for equipment and handling and loading/unloading space, which is dubbed preliminary length and width

3. Account for part queue space by adding either scaled part widths or scaled part lengths to appropriate preliminary length or width, depending on task type. Not all parts in part queue will add to part queue space, depending on task type and part arrangement. Result is semi-final length and width for station

4. Add aisle length and aisle width allowances to semi-final length and width to yield final length and final width for station. Multiply final length and final width to get area requirement for station

Up until now, the focus on area requirements was primarily directed at tasks and stations with equipment sets. The assembly tasks are unique in that they do not have any sets of

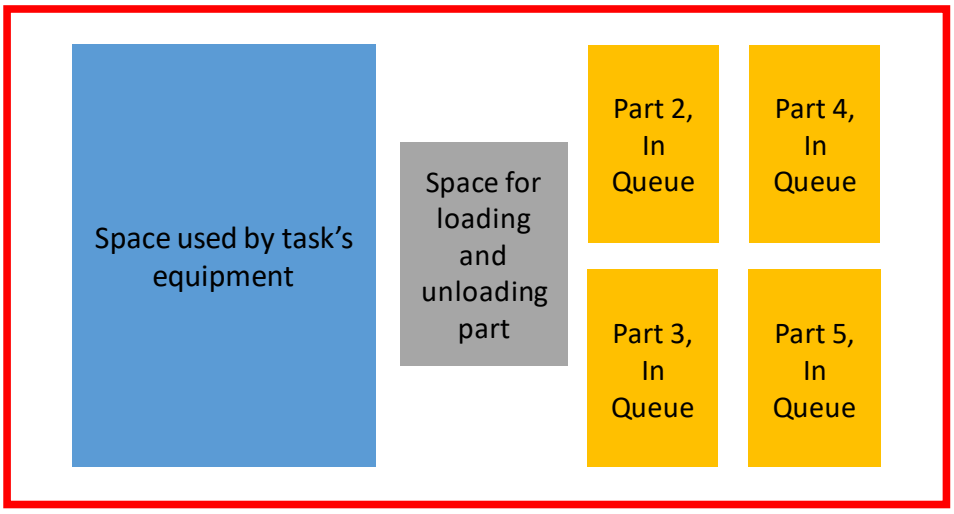


Figure A.16: Example Station and Resulting Space Allocation. Red Outline is Station Boundary, White Space is Aisle Space or Handling Space

equipment. But, as was mentioned in section A.4, all the different assembly tasks in the same station do not necessarily need their own tooling because some tasks work on the same set of assembled parts or add to them, meaning some tasks can share the same set of tooling. Due to this, for the stations with assembly and subassembly tasks, the assembly tooling acts as a sort of stand-in for the fabrication equipment. The cost of the assembly tooling can already be estimated by SEER. Subsequently, the number of tooling sets needed per station as well as how much space they take up, and the space requirements of assembly stations in general, need to be determined.

In SEER, the tooling size was set to be 1.25 times the part's size. However, this was for the cost of material to fabricate the tooling. The physical size of the tooling is expected to be much larger. To be conservative, the tooling is approximated to be 1.5 times the length and width of the parts and/or assembly it is supporting. Additionally, for the preliminary dimensions, an extra 10 feet of clearance is added to all sides to account for extra handling space and to make space for any extra tools needed. This is as opposed to the usual 5 feet used for fabrication because the assembly tasks are much more manual in nature and so

workers will need more room to move around. This results in the following preliminary tooling length $l_{tooling}$ and width $w_{tooling}$ as a function of the largest part length l_{part} and width w_{part} the tooling will support:

$$l_{prelim-tooling} = 1.5 l_{part} + 20 \tag{A.15}$$

$$w_{prelim-tooling} = 1.5 w_{part} + 20 \tag{A.16}$$

Loading and unloading space is not explicitly needed since parts will be directly loaded onto the tooling and not expected to be unloaded save for an exception to be discussed later. The space usually taken up by loading and unloading will instead be accounted for with the part queue space. This leaves the question of how many sets of tooling are needed per station given its task allocation. This is obtained by adhering to the following rules, which assumes all the referenced tasks are in the same station. Tooling sharing means that the applicable tasks only need one set of tooling among all of them.

- Tasks from same major or minor part assembly task set can share tooling
- Tasks from different minor part assembly task sets need their own tooling
- Tasks from different major part assembly task sets can only share tooling if those task sets are not used for subassemblies or integral fabrication
- Tasks from major part assembly task sets that do not use subassemblies or integral fabrication can only share tooling with one major part assembly task set that does use subassemblies if: the major part subassembly's fasten task is present and the major part, non-subassembly, non-integral fabrication assembly task set's fitup task is also present; the major part subassembly's fasten task immediately precedes the major part, non-subassembly, non-integral fabrication assembly task set's fitup task in terms of both assembly sequence and precedence

When tooling is shared, the tooling uses the dimensions of the largest part among the tasks that share the tooling or, if major part assembly is concerned, it takes the planform dimensions of the wingbox, i.e., the root width for the width and the wingbox length for the length. It was mentioned before that there is only one exception where parts are unloaded after being loaded onto the tooling. The exception is for all drilling tasks. This is because part of the drilling involves disassembling all the parts that have been drilled to deburr them, after which they are reassembled. Due to this, the preliminary lengths of any sets of tooling are doubled if a drilling task uses that particular tooling set. This is to make space to store the parts as they are disassembled for deburring.

In regard to how different tooling sets interact in terms of space, different sets of tooling are placed adjacent to each other such that the station's preliminary length is equal to the sum of the tooling set's preliminary lengths. To minimize space usage the maximum preliminary length of an assembly station is the sum of the lengths of the 2 longest tooling sets. Any further tooling sets are placed below the longest tooling sets as part of an additional row of tooling and essentially add their tooling set's preliminary width to the station's preliminary width. This is only done for every second additional tooling set. If the longest tooling set's size has already been doubled due to it being used for a drilling task, then all subsequent tooling sets are placed in subsequent rows such that each row's length is no higher than double the largest tooling set's preliminary length. This is best shown in Figure A.17 where, if tooling set 1 supported a drilling task, tooling set 2 would be placed where the queued part 2 is in the second row.

The parts in the part queue are arranged in the same manner as the multiple tooling sets. The parts are ordered in size from largest to smallest and then sequentially assigned, as can be seen in Figure A.17 from left to right and going down different rows. Only every second part added to the part queue adds to the station's overall width. The amount added to width w_{queue} based on the part's width w_{part} is:

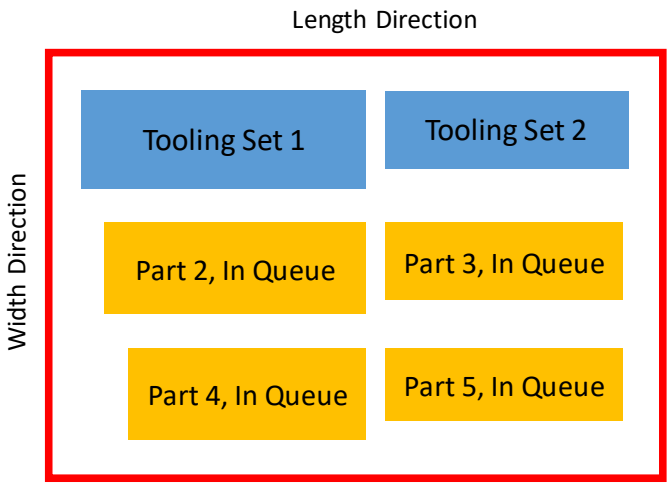


Figure A.17: Example Assembly Station and Resulting Space Allocation Showing How Different Tooling Sets and Part Queues are Arranged. Red Outline is Station Boundary, White Space is Aisle Space or Handling Space

$$w_{queue} = 1.25 w_{part} + 6 \tag{A.17}$$

The 6 feet is for clearance for the workers, 3 feet on both sides. Unlike with the queues for the fabrication tasks and stations, the first task's part is also placed in the part queue because it is first transferred into the station before being placed into its tooling. The parts in the queue are not actually on any tooling set. Because they are fully fabricated, they are stiff enough to be supported only by stands and so can be placed very closely to each other lengthwise. The only exceptions are with fitup tasks that involve that involve integral fabrication, where special tooling is used to support the parts.

A.6 Aircraft Design Considerations

All the sections covered thus far in this chapter of the appendix have been generally for the Experimental Platform for Experiment 1. In Experiment 2 and the Case Study, the aircraft design itself is varied, not just the assembly analysis variables. This warrants additional add-ons to some of the details already covered.

For Experiment 2 and the Case Study, the spar shape is one of the variables that is traded. It is chosen because it is a high level variable that affects the aircraft's design performance as well as its assembly performance. Specifically, it affects the assembly discipline by changing which sequences are feasible and which are not and is one of the highest-level variables that is able to do so. Five different shapes are studied in the Case Study: the default Out-C shape, the In-C shape, the Z shape, the I shape, and the J shape.

The spar shape affects the aircraft's performance in that it alters how much material is used by the spars by determining how efficiently the spar is able to handle loads during the structural sizing. The spar and the rest of the structure will weigh more if the shape is not efficient enough. How this is modeled is that during the Hypersizer structural sizing discussed in section A.1, the cross-sectional shape of the spar's bar elements that represent their spar caps are changed. The cross section for the default Out-C shape uses two L shapes. Using the front spar with an Out-C shape as the baseline, for the In-C shape the top bar element's shape is rotated 90° counterclockwise while the bottom bar element's shape is rotated 90° clockwise. For the Z spar the top bar element's shape is unchanged while the bottom bar's is rotated 90° clockwise. For the J spar, since the J is upside down, the top bar element remains unchanged while the bottom bar element used an upside T shape for its cross section. For the I spar, the top bar element's shape is an upright T and the bottom bar element's shape is an upside down T. After these changes for the various spar shapes, the sizing is carried out as normal. To prevent any spar shape from being artificially lighter or heavier simply due to the fact that the shape changed, as opposed to the shape's structural efficiency, the flanges are kept the same length regardless of spar shape, i.e., if the Out-C spar's top flange is 2.12 inches long, then the I spar's top flange is also 2.12 inches long. A visualization of what the spar shapes look like is in Figure A.18.

To account for the shape complexities associated with different spar shapes, all spar tasks with an Outwards-C or Inwards-C shape have a lower nominal shape complexity in SEER, J and Z spars have a nominal shape complexity, and I spars have a higher than nom-

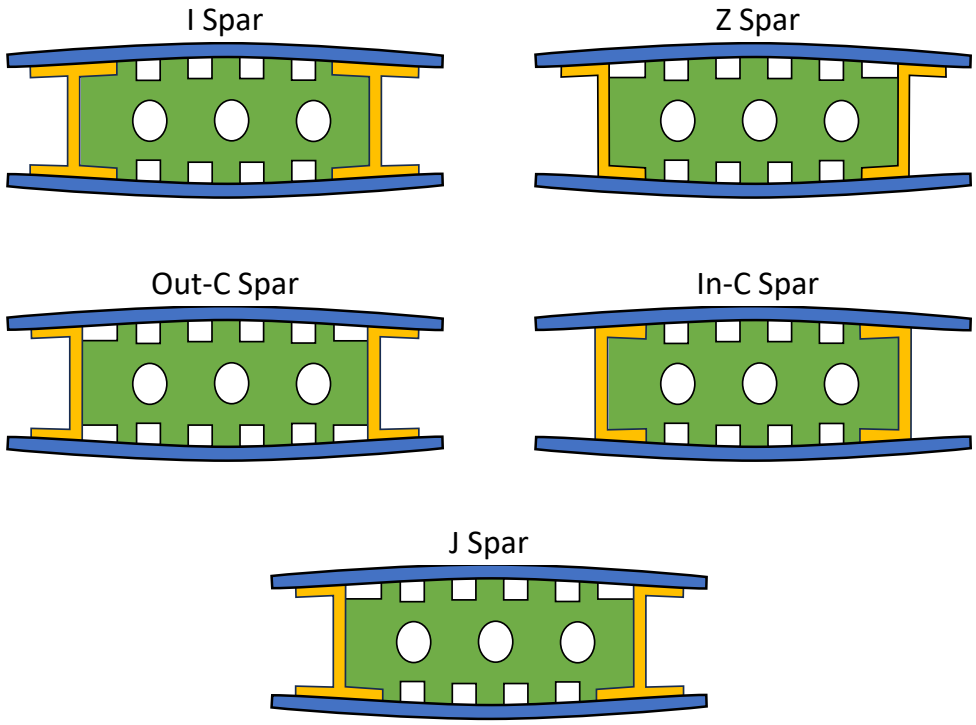


Figure A.18: Visualization of All Spar Shapes in 2D Wingbox Cross Section

inal shape complexity when shape complexity is an available parameter. Similarly, for the ATL process, Outwards-C and Inwards-C spars require only a single draping operation for the hot drape forming task while all of the other shapes require two hot draping operations.

The spar shape affects the assembly analysis in that different spar shapes make different sequences feasible or infeasible during the assembly sequencing. This can have notable downhill effects on the line balancing. Different spar shapes also have significant effects on an assembly sequence's accessibility. Since the accessibility changes how much time is spent on the assembly tasks in the SEER modeling, this has very large downhill effects on the line balancing as well. As a consequence of different spar shapes now affecting the assembly sequences, following the heuristics mentioned in section A.2 means that the interference matrices and accessibility analyses will need to be redone if a different spar shape is used. However, because the same spar shapes will yield the same matrices and accessibility regardless of the assembly's size, this also means that only several sets of

interference matrices need to be calculated, with the number of sets equal to the number of different spar shapes. The heuristics are updated as a result of this change and the ones applicable for the Case Study are shown below:

- All ribs and minor parts are treated as only a single part each during assembly sequencing
- Interference matrices do not have to be regenerated so long as the parts' shapes stay the same, regardless of their changes in size
- Interference matrices can be reused if the parts' shapes stay the same, even if their size changes
- Accessibility analysis only needs to be done on the last assembly step in an assembly sequence
- Accessibility is only dependent on the parts' shapes changing and the last assembly step in an assembly sequence, not on the size of the part

In the experimental platform, the F-86's design variables were kept constant. For the Case Study, which is focused on optimizing both aircraft and assembly design, the F-86's design variables are varied so that they and the assembly design can both be optimized. The specific variables being changed are the wingbox's aspect ratio, wing area, taper ratio, and range. The aspect ratio, wing area, and taper ratio are changed in OpenVSP at the very beginning of the sizing process so that their effects are felt all the way through to the structural optimization and so they can affect the aerodynamic characteristics, where they have some of the largest performance effects. The range is changed in FLOPS, which also subsequently requires that the mission be changed to allow the range to be variable. The new parametric mission profile is shown in Figure A.19.

Instead of the lengths and heights at each segment being preset, as was done for the design mission, FLOPS is allowed to vary those parameters itself such that the specified range

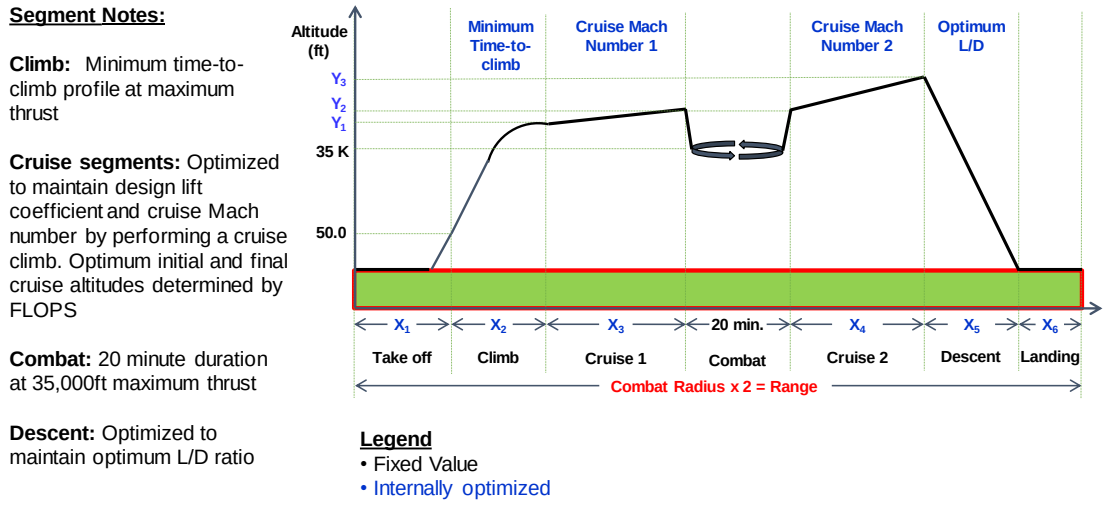


Figure A.19: F-86 Parametric Mission, Axes Are Not to Scale

is met while minimizing fuel weight and thus the total weight of the aircraft. Larger ranges require the aircraft to carry more fuel and thus handle additional weight and loads, increasing the required thicknesses of all the parts and ultimately their costs. Changing the design variables is thus a push-and-pull between desired aircraft performance and manufacturing performance. The same engine is used throughout this sizing.

It was mentioned at the end of section A.1 that the FLOPS program is rerun to obtain the design's updated characteristics and see if the design is able to meet the imposed performance constraints. For the Case Study, those constraints are the aircraft's: range, takeoff gross weight, fuel weight, fuel weight, landing approach velocity, takeoff field length, landing field length, climb rate, turn radius, and turn rate. The range has units of nautical miles, weights have units of pounds, the approach velocity has units of knots, the field lengths and turn radius have units of feet, the climb rate has units of feet per second, and the turn rate has units of degrees per second. These are all imposed by the user before the optimization begins, though care must be taken to not impose too many constraints that are too harsh lest there be no feasible designs.

There is a peculiarity with the field length constraints that must be noted. FLOPS only outputs the takeoff and landing field lengths in terms of Federal Aviation Regulations (FAR) takeoff and landing field lengths. However, the F-86 predates those regulations and it is desired to represent its constraints in terms of its rated characteristics to more easily compare with its baseline values. As such FLOPS's values must be converted to the F-86's rated values. The FAR landing fielding length in FAR Part 121.195 requires the actual landing length be divided by a factor of 0.6, meaning to obtain the F-86's rated landing length the FLOPS value must be multiplied by 0.6. For the takeoff field length, the FAR requirement in FAR Part 25.113 is that the length be 1.15 times the takeoff length with all engines operating, meaning FLOPS's value must be divided by 1.15. These conversions allow FLOPS's outputs to be directly compared with the F-86's stated characteristics.

During the optimization process, to meet the range constraint, the design is sized with the set range by default. The other constraints are handled in the genetic algorithm itself where if an individual's design is not able to meet the imposed performance constraints, a new individual is generated to replace it and the new performance constraints checked again. After enough new individuals have been generated and failed to meet the constraints, a penalty factor is applied to the remaining individual's production rate and cost that scales with how many constraints were violated and how much they were violated by. This is so that designs that barely miss the performance constraints are seen as more desirable than ones that completely miss the constraints. As barely infeasible designs are still infeasible, however, there is a minimum scaling to this penalty factor to ensure all but the poorest performing designs that meet the constraints are likely to have a higher fitness.

In Experiment 1, the RADE structural sizing framework, the FLOPS program, and the SEER program did not have to be called many times because only the baseline F-86 design was examined. In Experiment 2 and the Case Study, however, numerous F-86 designs are sifted through to search for both the optimal aircraft design along with its subsequent assembly design; every individual in the implemented genetic algorithm will be needing to

call the above programs or framework at least once. The RADE framework can take up to 90 minutes, sometimes more, to return optimized results for a single aircraft design. The SEER program requires 20 to 40 seconds to output all the desired task times and cost for a single assembly design. And the FLOPS program requires 1 to 2 seconds to output a sized aircraft. This is all infeasible in regard to time when the genetic algorithm optimization is likely to need to generate and evaluate tens of thousands of individual designs. Instead, as suggested by Mavris et al., surrogate models of those programs and framework should be made to significantly speed up the process [248]. Surrogate models are simple mathematical models able to approximate time-intensive computer functions or programs and can reduce their run times to just fractions of a second [276]. Artificial Neural Network (ANN) surrogate models specifically are made for RADE, SEER, and FLOPS because they can easily capture non-linear interactions and still produce accurate results [277]. However, because surrogate models, especially ANNs, can become wildly inaccurate when used outside of the variable ranges they were created with, the Design of Experiments used to create them must be carefully constructed.

The continuous input variables for the surrogate models and their ranges are shown in Table A.2 while the categorical input variables and their possible options are shown in Table A.3. These ranges are based off of those used in Siedlak et al.'s work [29]. Wing sweep is not varied because the design maximum Mach number for the aircraft is kept at 0.9, meaning greater sweep is not necessary and less sweep is not practical for aircraft performance. It is thus set at the F-86's baseline value.

Table A.2: Continuous Variables and Ranges for Design of Experiments for Surrogate Models for SEER, FLOPS, and RADE Framework

Input Variable	**Min**	**Max**
Wing Area (ft^2)	240	410
Aspect Ratio	3.84	6.24
Taper Ratio	0.36	0.66
Range (nmi)	600	1000

Table A.3: Categorical Variables and Options for Design of Experiments for Surrogate Models for SEER, FLOPS, and RADE Framework

Input Variable	**Variable Options**		
Spar Shape	Outwards C	Inwards C	Z
	I	J	
Manufacturing Process	HLU /Autoclave	VARTM	ATL
	SRI	CNC Machining	

The outputs from the RADE surrogate model are the optimized dimensions for all the parts, the outputs from the SEER surrogate model are all the times and costs described in section A.4, and the outputs from the FLOPS surrogate model are all the performance constraints listed above. Technically speaking, a surrogate model is needed for every individual output, necessitating a large number of surrogate models. For the experimental platform, RADE's results are used by both SEER and FLOPS to obtain their results, and so the same relationship is kept when their surrogate models are used. To maintain as much accuracy and consistency as possible, then, the cases used to make the SEER and FLOPS surrogate models must also be the same ones used to make the RADE ones; making the SEER and FLOPS models with results from the RADE surrogate model means that the former's inputs will already have some degree of inaccuracy due to their inputs being the output of a surrogate model which, while accurate, is not perfectly accurate. Because of this requirement, the RADE framework becomes a significant bottleneck due to requiring up to 90 minutes or more per case. Additionally, all the cases for the continuous variables must all be rerun for every combination of the categorical variables to ensure no unusual interactions are missed. This means the Design of Experiments used for the continuous variables has to cover as much of the design space as possible while requiring as few cases as possible.

With this in mind, a Box-Behnken Design of Experiments was used to create the initial cases for the continuous design variables and then Latin Hypercube sampling was used to

generate cases until a total of 180 continuous ones were made. These cases were then run through the RADE framework and the SEER and FLOPS programs for every combination of the categorical variables to obtain their associated outputs. That data was afterwards used to create the ANN surrogate models in the JMP software. Generally only one layer of 9 neurons using 3 each of the Tanh, Gaussian, and Radial activation functions were used, though some outputs required 2 layers of 6 neurons each, with each layer having 2 neurons each with the Tanh, Gaussian, and Radial activation functions. The learning rate was set to 0.2, 2 tours were carried out, and 33% of the cases were used for validation of the data. The resulting surrogate models had very high accuracy, with the lowest accuracy ones still having R^2 values of around 0.95 and the vast majority having training and validation R^2 values above 0.99 with no noticeable patterns in the residuals. With the surrogate models created, it takes a maximum of 0.04 seconds to query the all the SEER results with the SEER surrogate models, which are the most complex and take the longest to run. This represents a roughly 1000 fold speed increase for the SEER outputs and an 8 million fold minimum speed increase for the RADE framework for otherwise relatively accurate results, showing the necessity of surrogate models.

APPENDIX B

GEOMETRIC MODELING DETAILS

This section goes over the geometries of the parts used in the F-86 experimental platform's wingbox.

The following geometries for the F-86 wingbox's parts were taken from the various MInD works including those by Ceisel et al. [60], Sundaresan et al. [61], and Siedlak et al. [62, 63, 29].

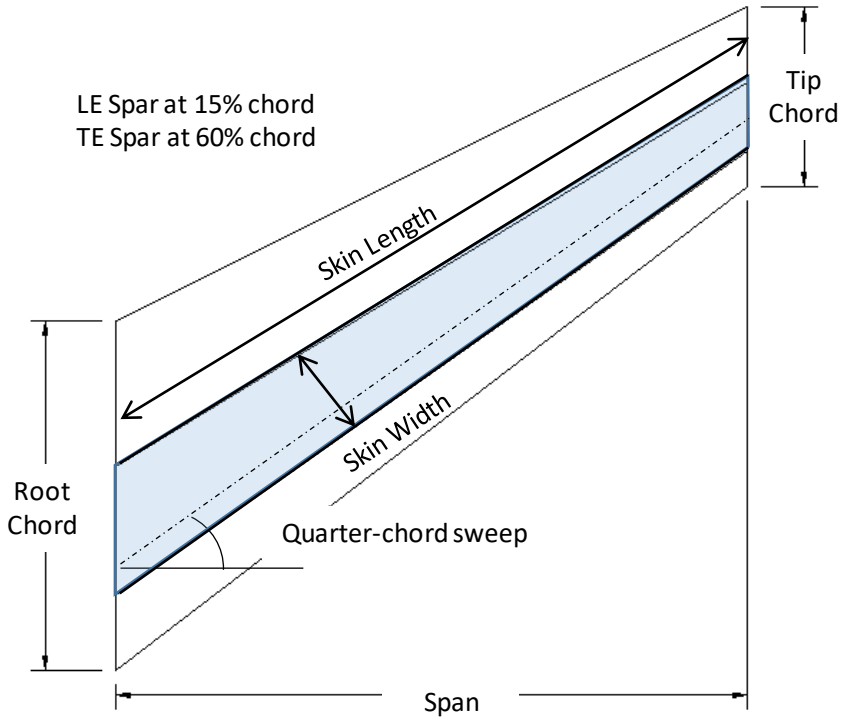
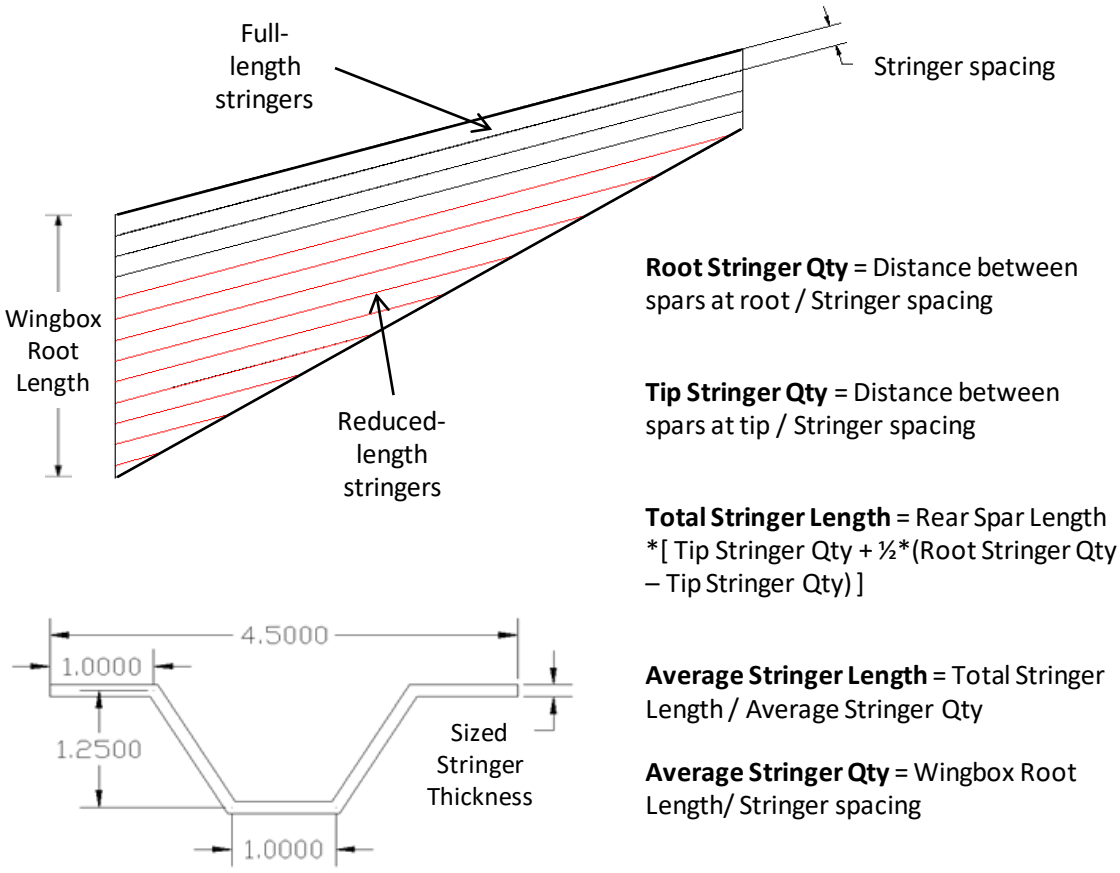


Figure B.1: General Geometry for Wingskins, Top View of Wingskin and Wingbox Superimposed on Full Wing



Root Stringer Qty = Distance between spars at root / Stringer spacing

Tip Stringer Qty = Distance between spars at tip / Stringer spacing

Total Stringer Length = Rear Spar Length *[Tip Stringer Qty + ½*(Root Stringer Qty – Tip Stringer Qty)]

Average Stringer Length = Total Stringer Length / Average Stringer Qty

Average Stringer Qty = Wingbox Root Length/ Stringer spacing

Figure B.2: General Geometry for Wing Stringers, Units in Inches

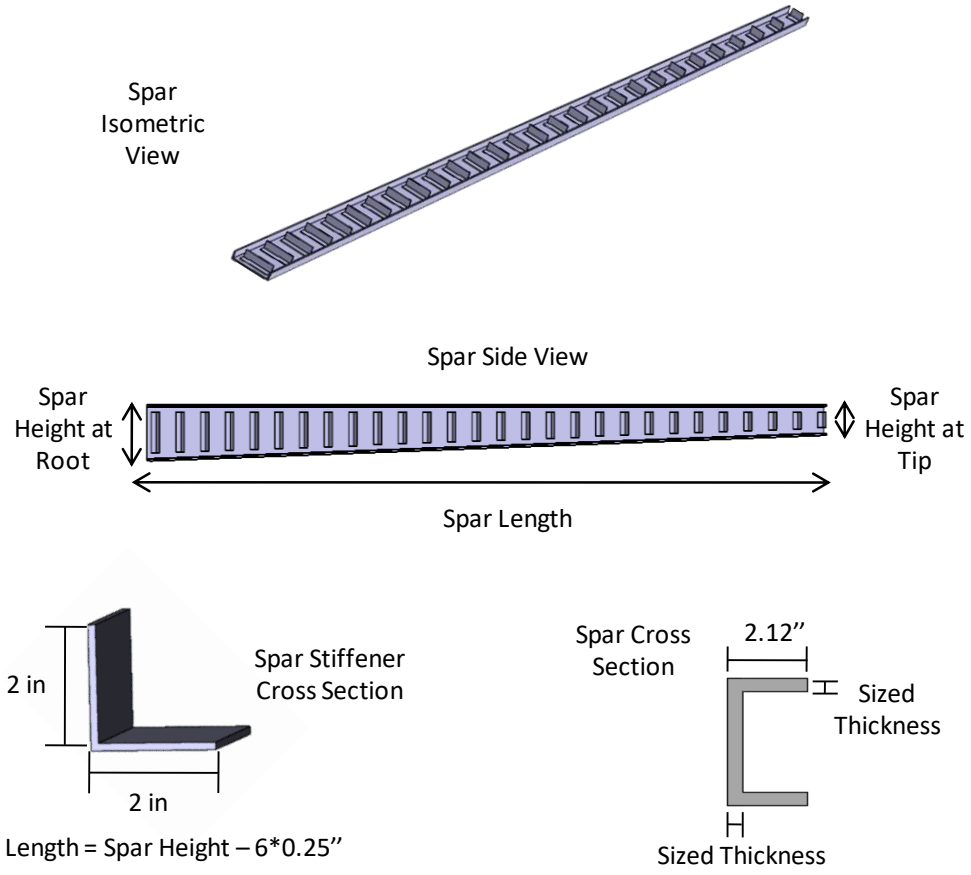


Figure B.3: General Geometry for Spar and Spar Stiffeners, Units in Inches

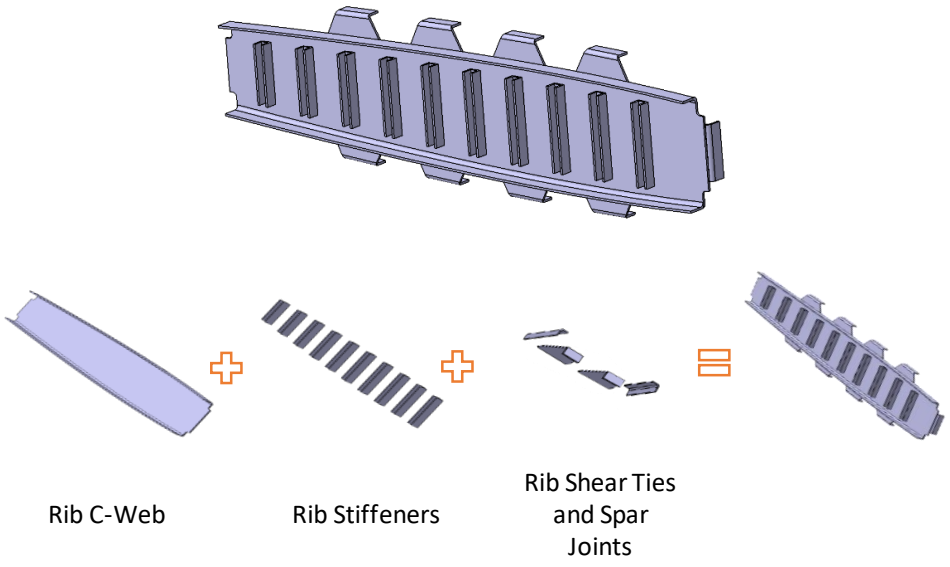


Figure B.4: Composite Rib and Part Decomposition

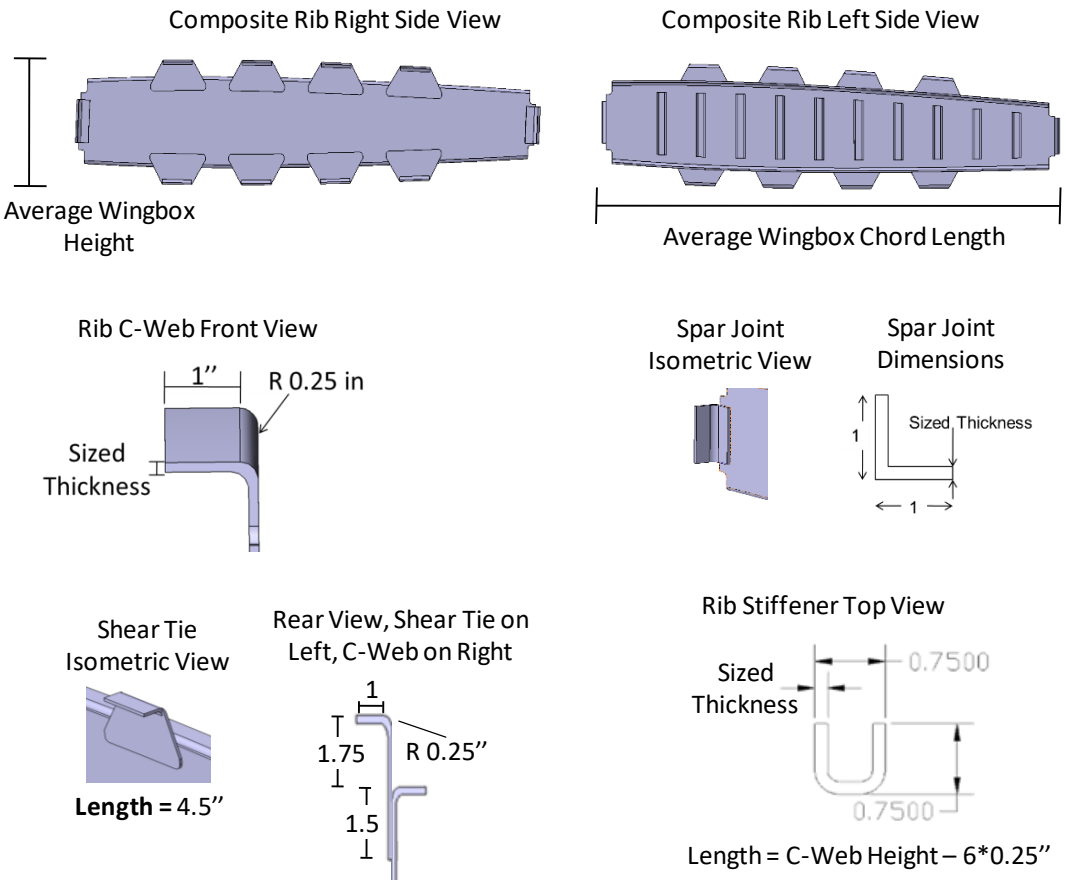


Figure B.5: General Geometry for Composite Rib Parts, Units in Inches

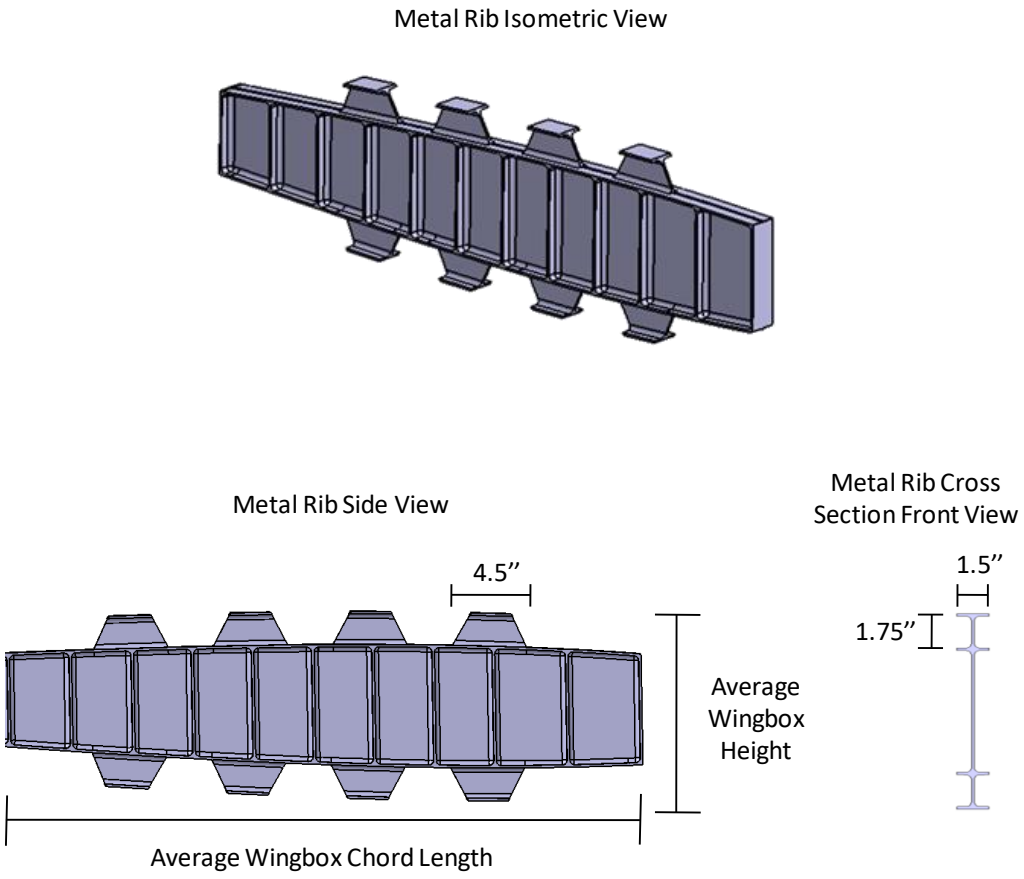


Figure B.6: General Geometry for Metal Rib, Units in Inches

APPENDIX C

IF-THEN PRECEDENCE RULES FOR TASK PRECEDENCE MATRIX

This section goes over the If-Then precedence rules used in the rule-based constraint modeling to define the possible integral fabrication part combinations as well as to generate the dynamic precedence graphs.

The following rules are for all processes.

- IF major part assembly operations are performed, THEN the maximum and default number of major part assembly task sets is equal to the number of major parts in the assembly sequence minus 1
- IF minor parts are integrally fabricated with a major part, THEN their fitup task with their major part must occur before the major part's cure task, whether it be autoclave or oven cure task
- IF minor parts are integrally fabricated with a major part, THEN their drill and fasten minor part-to-major part assembly tasks are removed from the list of available tasks
- IF any of the rib's minor parts are integrally fabricated with the ribs, THEN the other rib minor parts must also be integrally fabricated with the ribs
- IF minor parts are not integrally fabricated with their major parts, THEN their fasten parts task where they are fastened to their major parts must occur before any of their major parts' major part-to-major part assembly tasks
- IF major parts are part of a subassembly or are integrally fabricated, THEN remove major part assembly task sets equal to the number of parts in said subassembly or part of said integrally fabricated part minus 2, for each subassembly or integrally fabricated part

- IF major parts are part of a subassembly or are integrally fabricated, THEN each set of parts is assigned a major part assembly task set according to the order they are listed in the assembly sequence. Regular, non-subassembly and non-integral fabrication major assembly operations are only assigned assembly task sets after subassemblies and integrally fabricated parts are assigned their task sets

The following rules are for the HLU/Autoclave process.

- IF the HLU/Autoclave process is used, THEN the fabrication task ordering is hand layup, autoclave cure, part trim, and non-destructive inspection and the assembly task ordering is parts fitup, drill parts, and fasten parts, with all of a part's available fabrication tasks needing to be finished before it can start its assembly tasks unless otherwise noted

- IF the HLU/Autoclave process is used, THEN there is a fabrication task set for each of the following parts: spar stiffeners, stringers, rib shear ties and spar joints, rib stiffeners, front spar, rear spar, upper skin, lower skin, and ribs

- IF the HLU/Autoclave process is used, THEN there is a minor assembly task set for each of the following parts combining them to their associated major parts: front spar stiffeners, rear spar stiffeners, upper stringers, lower stringers, rib minor parts

The following rules are for the ATL process.

- IF the ATL process is used, THEN the fabrication task ordering is automated tape layup, hot drape form, autoclave cure, part trim, and non-destructive inspection and the assembly task ordering is parts fitup, drill parts, and fasten parts, with all of a part's available fabrication tasks needing to be finished before it can start its assembly tasks unless otherwise noted

- IF the ATL process is used, THEN there is a fabrication task set for each of the following parts: spar stiffeners, stringers, rib shear ties and spar joints, rib stiffeners, front spar, rear spar, upper skin, lower skin, and ribs

- IF the ATL process is used, THEN there is a minor assembly task set for each of the following parts combining them to their associated major parts: front spar stiffeners, rear spar stiffeners, upper stringers, lower stringers, rib minor parts

The following rules are for the VARTM process.

- IF the VARTM process is used, THEN the fabrication task ordering is hand layup, oven cure, part trim, and non-destructive inspection and the assembly task ordering is parts fitup, drill parts, and fasten parts, with all of a part's available fabrication tasks needing to be finished before it can start its assembly tasks unless otherwise noted

- IF the VARTM process is used, THEN there is a fabrication task set for each of the following parts: spar stiffeners, stringers, rib shear ties and spar joints, rib stiffeners, front spar, rear spar, upper skin, lower skin, and ribs

- IF the VARTM process is used, THEN there is a minor assembly task set for each of the following parts combining them to their associated major parts: front spar stiffeners, rear spar stiffeners, upper stringers, lower stringers, rib minor parts

- IF minor parts are integrally fabricated with a major part, THEN the minor part's oven cure, part trim, and non-destructive inspection fabrication tasks along with their drill and fasten minor part-to-major part assembly tasks are removed from the list of available tasks

The following rules are for the SRI process.

- IF the SRI process is used, THEN major part integral fabrication can be performed

- IF the SRI process is used, THEN the fabrication task ordering is hand layup, stitch, oven cure, part trim, and non-destructive inspection and the assembly task ordering is parts fitup, drill parts, and fasten parts, with all of a part's available fabrication tasks needing to be finished before it can start its assembly tasks unless otherwise noted
- IF the SRI process is used, THEN there is a fabrication task set for each of the following parts: spar stiffeners, stringers, rib shear ties and spar joints, rib stiffeners, front spar, rear spar, upper skin, lower skin, and ribs
- IF the SRI process is used, THEN there is a minor assembly task set for each of the following parts combining them to their associated major parts: front spar stiffeners, rear spar stiffeners, upper stringers, lower stringers, rib minor parts
- IF minor parts are integrally fabricated with a major part, THEN the minor part's oven cure, part trim, and non-destructive inspection fabrication tasks along with their drill and fasten minor part-to-major part assembly tasks are removed from the list of available tasks
- IF minor parts are integrally fabricated with a major part, THEN the minor part's stitching task must be performed after the minor part's fitup task and before the major part's oven cure task
- IF major parts are integrally fabricated with other major parts, THEN the major parts must all be fitted up using a major part assembly task set fitup task after all the major parts are stitched with their minor parts
- IF major parts are integrally fabricated with other major parts, THEN all their minor parts must also be integrally fabricated
- IF major parts are integrally fabricated with other major parts, THEN the major parts must all be fitted up using a major part assembly task set fitup task before all the major parts are stitched together using one of the major parts' stitching tasks

- IF major parts are integrally fabricated with other major parts, THEN the major parts that did not have their stitching task used have their stitching, oven cure, part trim, and non-destructive inspection fabrication tasks removed from the list of available tasks

- IF a part is not integrally fabricated, THEN its stitching task is not used and is eliminated from the list of available tasks

The following rules are for the CNC Machining process.

- IF the CNC Machining process is used, THEN integral fabrication cannot be used

- IF the CNC Machining process is used, THEN the fabrication task ordering starts with the CNC Mill task for the front spar, ribs, and rear spar, with the pre-form, stretch form, after-form, and CNC Mill tasks for the upper and lower wingskins, and with the pre-form, roll form, and after-form tasks for the stringers

- IF the CNC Machining process is used, THEN the fabrication task set for all the parts end with the dye penetrant, anodize, and chromate spray tasks

- IF the CNC Machining process is used, THEN there is a minor assembly task set for each of the following parts combining them to their associated major parts: upper stringers, lower stringers

APPENDIX D

SEER-MFG CNC MACHINING MODELING AND OTHER OPERATION DETAILS

This section goes over the details of the modeling used for the CNC Machining process in SEER-MFG.

Table D.1, Table D.3, and Table D.2 describe the specific machining operations, their purpose, and any additional notes used for the fabrication of the metallic spars and ribs and the trimming of the wingskins. These were obtained from previous MInD works.

The machining operations in Table D.1 can be used for the Out-C, In-C, and I spar shapes because the same amount of material is removed in all of them. However, the Z and J shapes require extra material to be removed, which requires additional machining operations. This is reflected in the operations done for Z and J spars shown in Table D.4. These spar shapes also subsequently need larger starting blocks of raw aluminum.

Table D.1: Spar Machining Operations, Purpose, Additional Details for Outwards-C Shape

Operation Type	**Purpose**	**Additional Notes**
High Speed Rough	Taper Top Surface	Includes Setup, Load, Unload
High Speed Rough	Taper Bottom Surface	
High Speed Rough	Remove Excess Material in Web Area	
High Speed Finish	Finish Top Surface	Includes Setup
High Speed Finish	Finish Bottom Surface	
High Speed Finish	Finish Excess Material in Web Area	
High Speed Rough	Remove Pockets for Stiffeners	Includes Setup, Load, Unload
High Speed Finish	Finish Pockets	Includes Setup

For the dye penetrant inspection task's inspection speed, it is assumed ultraviolet lights

Table D.2: Wingskin Machining Operations, Purpose, Additional Details

Operation Type	**Purpose**	**Additional Notes**
High Speed Rough	Trim IML Surface	Includes Setup, Load, Unload
High Speed Rough	Trim OML Surface	
High Speed Finish	Finish OML Surface	Includes Setup

similar to Magnaflux ultraviolet lamps [278] are used, which have a 9 inch wide beam. Given an assumed scan speed of 6 feet per minute, this means the dye penetrant inspection rate is about 648 in^2 per minute.

For the chromate spray task, it is assumed a single person can solvent wipe at a rate of 10 ft^2 per minute, meaning 2 people in the station results in a rate of 20 ft^2 per minute for the solvent wipe operation.

Table D.3: Rib Machining Operations, Purpose, Additional Details

Operation Type	**Purpose**	**Additional Notes**
High Speed Rough	Pockets for Stiffeners, Left Side	Includes Setup, Load, Unload
High Speed Rough	Profile Machine Rib Top	
High Speed Rough	Profile Machine Rib Bottom	
High Speed Rough	Profile Machine Left and Right Sides	
High Speed Rough	Shear Tie Top Slot Left Side	
High Speed Rough	Shear Tie Bottom Slot Left Side	
High Speed Rough	Remove Material Between Top Shear Ties	
High Speed Rough	Remove Material Between Bottom Shear Ties	
High Speed Finish	Finish Pockets for Stiffeners, Left Side	Includes Setup
High Speed Finish	Finish Rib Top	
High Speed Finish	Finish Rib Bottom	
High Speed Finish	Finish Left and Right Sides	
High Speed Finish	Finish Shear Tie Top Slot Left Side	
High Speed Finish	Finish Shear Tie Bottom Slot Left Side	
High Speed Finish	Finish Material Removal Between Top Shear Ties	
High Speed Finish	Finish Material Removal Between Bottom Shear Ties	
High Speed Rough	Pockets for Stiffeners, Right Side	Includes Setup, Load, Unload
High Speed Rough	Shear Tie Top Slot Right Side	
High Speed Rough	Shear Tie Bottom Slot Right Side	
High Speed Finish	Finish Pockets for Stiffeners, Right Side	Includes Setup
High Speed Finish	Finish Shear Tie Top Slot Right Side	
High Speed Finish	Finish Shear Tie Bottom Slot Right Side	

Table D.4: Spar Machining Operations, Purpose, Additional Details for Z and J Shapes

Operation Type	**Purpose**	**Additional Notes**
High Speed Rough	Taper Top Surface	Includes Setup, Load, Unload
High Speed Rough	Taper Bottom Surface	
High Speed Rough	Remove Excess Material in Web Area	
High Speed Finish	Finish Top Surface	Includes Setup
High Speed Finish	Finish Bottom Surface	
High Speed Finish	Finish Excess Material in Web Area	
High Speed Rough	Remove Pockets for Stiffeners	Includes Setup, Load, Unload
High Speed Finish	Finish Pockets	Includes Setup
High Speed Finish	Remove Extra Flanges	Includes Setup, Load, Unload
High Speed Rough	Remove Extra Excess Web Area Material	Includes Setup
High Speed Finish	Finish Extra Material Removal	Includes Setup

APPENDIX E

EQUIPMENT SPACE AND COST MODELING DETAILS

Appendix E goes over the estimation of the equipment cost, the estimation of the dimensions for the preliminary spatial requirements in the form of preliminary station lengths l_{prelim} and widths w_{prelim} that capture space needs for the equipment, handling, and loading/unloading, as well as how the parts are arranged in their queues for each task type and how they subsequently contribute to the semi-final lengths l_{sfinal} and widths w_{sfinal} for each station. How each part in the queue adds to the queue space dimensions, if it affects it at all, is generally based on either the length l_{part} or width w_{part} of the part being added to the queue multiplied by a 1.25 factor to account for the tooling the part is on and an additional 3 feet of clearance on all sides to account for space for workers to walk between the parts. This is shown below, where PQ are the parts in the part queue that actually affect the queue space requirements and is not necessarily all of the parts. Since parts generally only add part queue space in one dimension, the summation portion of either Equation E.1 or Equation E.2 will generally be zero for most task types and the stations they are in. Note, however, that different task types may affect this relationship. If only one of either Equation E.3 or Equation E.4 is present, the other can be assumed to be zero. Different task types may also have different versions of Equation E.3 or Equation E.4. As such, exceptions will be noted. Whenever the phrase "parts in the queue" is referred to, because the first task's part is already being loaded onto the equipment, it means the first task's part is not included in the part queue unless noted otherwise.

$$l_{sfinal} = l_{prelim} + \sum_{p=1}^{PQ} l_{queue_p} \tag{E.1}$$

$$w_{sfinal} = w_{prelim} + \sum_{p=1}^{PQ} w_{queue_p} \tag{E.2}$$

$$l_{queue} = 1.25 l_{part} + 6 \tag{E.3}$$

$$w_{queue} = 1.25 w_{part} + 6 \tag{E.4}$$

To create the final lengths and widths for each station, the aisle width and aisle length allowances described in section A.5 are added to each station's respective semi-final widths and lengths. These allowances are listed below for convenience, where W_{max} is the width of the widest part.

$$L_{final} = l_{sfinal} + 4 + 3W_{max} \tag{E.5}$$

$$W_{final} = w_{sfinal} + 7 + \frac{5}{8} W_{max} \tag{E.6}$$

The area requirement for a station is obtained by taking the product of its final length and final width with the exception of autoclave and oven cure stations, which are discussed in section A.5. All part lengths and widths referenced assume that the part's maximum widths and lengths are being used, and that the part itself being referenced is the largest part in that station. How this is defined for each part type is described at the beginning of section A.5.

All length and width dimensions are assumed to be in feet, with costs being in 2022 US Dollars.

E.1 Automated Tape Layup

The data for the automated tape layup machine is sourced from the works of Goel [271] and Hagnell and Akermo [272] and shown in Figure E.1. Hagnell and Akermo obtained their data from the MTorres company, but examination of a typical Torres Fiber Layup machine such as the one in [279] indicates that what Hagnell and Akermo call the gantry size is far too small to be the actual gantry size and is assumed to be the part size instead. This is reflected in the data in Figure E.1.

Part Area (ft^2)	Cost (Thousands 2022 US Dollars)
172	4,900
3165	11,700
3000	6,800
2240	7,800

Figure E.1: Data on Automated Tape Layup Machine Cost as a Function of Part Area, Adapted from Goel [271] and Hagnell and Akermo [272]

A linear relationship is assumed between the part size and the cost of the machine, resulting in the following cost estimating regression based on the part's length l and width w, represented in feet:

$$C_{atl} = 4418384 + 1562.4 * w * l \tag{E.7}$$

Based on the machine shown in [279], which is assumed to be representative, and using the man sitting at the machine as well as the stairs being 3 feet in width to meet OSHA compliance, it is estimated that the distance from the edge of the machine to the part is about 12 feet on both sides. There is similarly 8 feet of clearance in front of and behind the part to make room for the gantry. An additional 5 feet is added to all sides to account for handling room and room for the workers to safely walk around. The part is loaded directly under the machine, and so no loading/unloading space is needed. The part itself is

assumed to be resting on a platform that is 1.25 times its length and width so that the part does not stick beyond the platform, no matter its shape. The width direction is taken to be the direction the gantry moves back and forth in. The preliminary length and width space requirements $L_{prelim-atl}$ and $W_{prelim-atl}$ for the automated tape layup machine given the part's length l and width w are:

$$L_{prelim-atl} = 1.25l + 26 \tag{E.8}$$

$$W_{prelim-atl} = 1.25w + 34 \tag{E.9}$$

For the part queue, it is most efficient to stack the parts sideways as compared to how they are oriented when the machine is operating on them. The parts are arranged in front of the machine and affect its queue length $l_{queue-atl}$ via their part width w_{part}, show below. All parts in the queue add to this length instead of only every other part.

$$l_{queue-atl} = 1.25w_{part} + 6 \tag{E.10}$$

E.2 CNC Machine

The data for the CNC machine is sourced from [280] and shown in Table E.1. It is clear that the mentioned size of the smaller machine in the article refers to the size of the platform that the part sits on and that the mentioned size of the larger machine refers to the machine's actual size. Some conversions must be made so that a common reference size is used. Based on schematics of the C.R. Onsrud CNC machining center in [281], which [280] makes clear is representative of the larger machine, the total width of the machine trends to be 9 feet larger than its part-supporting platform width, or working width. Additionally, it is estimated there is about 4 feet of clearance in front of the platform's long direction, assumed to be the length direction, and about 12 feet of clearance behind the platform,

leading to the dimensions seen in Table E.1.

Table E.1: CNC Machine Cost, Adapted from [280] and Updated Based on [281]

Platform Size (ft^2)	**Cost (Thousands 2022 US Dollars)**
72 (12' $\times$ 6')	300
924 (84' $\times$ 11')	1,500

The platform size is approximated to be 1.25 times larger than the part it is supporting in both length and width so that awkwardly shaped parts do not stick out of it. This is represented via the following equations, where l is the part's length and w is the part's width.

$$l_{platform-cnc} = 1.25l \tag{E.11}$$

$$w_{platform-cnc} = 1.25w \tag{E.12}$$

A linear cost relationship is assumed between the CNC machine's cost C_{cnc} and its platform's length $l_{platform-cnc}$ and width $W_{platform-cnc}$, shown in the regression below.

$$C_{cnc} = 198592 + 1408.5 * w_{platform-cnc} * l_{platform-cnc} \tag{E.13}$$

For the CNC machine's preliminary dimensions, based on the observations made to generate Table E.1, the machine's length is 16 feet longer than its platform's length and the machine's width is 9 feet wider than its platform's width along with an extra 5 feet of clearance on both sides to account for the support struts seen in [281]. Based on those schematics, the inside of the machine center already includes handling space, and the part itself is loaded directly onto its support platform, so both types of space are already accounted for. The preliminary length and width space requirements $L_{prelim-cnc}$ and $W_{prelim-cnc}$ for the

CNC machine given the part's length l and width w are:

$$L_{prelim-cnc} = 1.25l + 19 \tag{E.14}$$

$$W_{prelim-cnc} = 1.25w + 16 \tag{E.15}$$

For the part queuing, parts are arranged outside of the machine center to its side to not obstruct the door. They are arranged such that their widths add to the preliminary widths, as indicated in Equation E.4. Every part in the queue adds to the queue width.

E.3 CNC Router

The CNC router is used for the pre-form and after-form tasks for the metallic stringers. Because it is a CNC machine in the same way the machine in the previous section, only that this one is meant for routing purposes as opposed to milling, it is assumed the cost relations are the same between the two, save that the router is much smaller. Thus, the same data shown in Table E.1 is applicable and the cost equation shown in Equation E.13 is used for the CNC Router as well. The main difference is in the size of the platform used to support the part. While the platform length is unchanged and is the same as Equation E.11, the platform width is set to a constant 5 feet. This is based on [282], which is a CNC router assumed to be representative of the type that would be used for metallic stringers. That router has a set working width of 5 feet regardless of length, meaning it is also set at 5 feet for the cost equation. The assumption is made that if the stringers that the router operates on are more than 5 feet in total width, some will just be stored in the loading area while the rest are being operated on.

The width of the router itself is always 11 feet. An additional width of 1.25 times the width of the stringers is added to that to allow for loading and unloading space, with 1.25 used so the stringers do not stick outside of the loading zone. An additional 5 feet is

added to the width to make room for any operating computers. The length is the length of the support platform, equal to 1.25 times the stringer length, plus an additional 8.5 feet to account for the rest of the router's parts, as is seen in the schematics in [282]. This is all topped off with an additional 3 feet of clearance on all sides for handling space and worker access. The resulting preliminary length and width space requirements $L_{prelim-cncr}$ and $W_{prelim-cncr}$ for the CNC router given the part's length l and width w are:

$$L_{prelim-cncr} = 1.25l + 14.5 \tag{E.16}$$

$$W_{prelim-cncr} = 1.25w + 22 \tag{E.17}$$

Given that the router is more likely to be wide than it is longer, the most space-efficient part queue arrangement is to orient the parts in the queue such that their widths w_{part} add to the queue length $l_{queue-cncr}$. All parts in the queue add to this length instead of only every other part.

$$l_{queue-cncr} = 1.25w_{part} + 6 \tag{E.18}$$

E.4 Manual Router

The manual router is used for the metallic wingskins' pre-form and after-form tasks. Because the HLU/Autoclave process has a completely manual trimming task that also uses a hand router, the space requirements for the manual router stations is used for those tasks too. As it is entirely manual in nature, no equipment and therefore no equipment costs are calculated for it. Its tooling costs are already estimated by SEER and so what is left is that its space requirements must be determined. The part and the tool it sits on are estimated to be the typical part length l and w multiplied by a factor of 1.25. Space is also needed to load and unload the part into the working area, increasing the preliminary width $W_{prelim-cncm}$

by another factor of 1.25 times the part's width. And finally, 8 feet of clearance is added to all sides to provide room to work and store all the necessary tools to perform the routing for the parts. This is shown by the following equations:

$$L_{prelim-cncm} = 1.25l + 16 \tag{E.19}$$

$$W_{prelim-cncm} = 2.5w + 16 \tag{E.20}$$

The optimal way to store the parts in queue is to add the part's width w_{part} to the queue length $l_{queue-cncm}$ since the station's width tends to be larger than its length. All parts in the queue add to the queue length.

$$l_{queue-cncm} = 1.25w_{part} + 6 \tag{E.21}$$

E.5 Stretch Forming Machine

The stretch forming machine is used for the metallic wingskins' stretch forming task. One of the only known public sources of both its cost and size is given by Schrader and Elshennawy in their book, where a machine able to stretch form an 8' by 40' aircraft panel costs about $3.24 million in 2022 [283]. It is assumed based on this that the cost scales purely linearly as a function of the product of the part's length l and the part's width w, shown below:

$$C_{stretch-form} = 3240000\frac{lw}{8 * 40} \tag{E.22}$$

ACB's stretch forming machine is taken to be representative of the stretch forming machines that will be used [284]. Based on their video demonstration of their equipment, the length between the machine's jaws is essentially the platform for the part and so is 1.25 times the part length plus an additional 3 feet for the grippers. The length of the two jaws

themselves are estimated to be about 20 feet each, with their ends each being about 4 feet away from the rails marking the end of the machine. There is roughly 14.5 feet of space from either side of the jaws to the edge of the station, with the width of the jaws themselves being essentially the width of the platform for the part, equal to 1.25 times the part width. An additional width of 1.25 times the part width is added to the side of the machine to make space for loading the part in. The machine itself includes a large amount of handling space and rails, and so handling space is already included. This results in the stretch forming stations' preliminary length $L_{prelim-stretch}$ and width $W_{prelim-stretch}$ as a function of the part length l and width w being:

$$L_{prelim-stretch} = 1.25l + 51 \tag{E.23}$$

$$W_{prelim-stretch} = 2.5w + 29 \tag{E.24}$$

Because there are only ever two stretch forming tasks, one for each wingskin, if the other wingskin is in the part queue, its part width w_{part} is added to the queue length $l_{queue-stretch}$ as described by the equation:

$$l_{queue-stretch} = 1.25w_{part} + 6 \tag{E.25}$$

E.6 Roll Forming Machine

The roll forming machine is used for the metallic stringers' roll forming task. The cost of a new machine is obtained from Schrader and Elshennawy in their book, where an integrated roll forming system estimated to be able to operate on parts 20 feet long costs about $1,620,000 in 2022 [283]. However, the integrated roll forming system in their book is composed of two separate roll forming machines, where only one is needed for a roll forming task, meaning each machine actually costs $810,000 in 2022. Additionally, roll

forming machines make stringers one by one and so do not need to be very wide at all to be able to accommodate every reasonable stringer size. As a result, it is assumed that the cost scales purely linearly as a function of the part's length l and, combined with Schrader and Elshennawy's data above, results in the cost regression shown below:

$$C_{roll-form} = 810000\frac{l}{20} \tag{E.26}$$

New Tech Machinery's roll forming machines are taken as being reasonably representative of stringer roll forming machines [285]. The width of the machine is always 5 feet since no single stringer when laid out will ever have a width of five feet or higher. Additional space is added to the preliminary width $W_{prelim-roll}$ in the form of the stringer width multiplied by a factor of 1.25 to account for space to store the stringers after they have been roll formed. For the preliminary length $L_{prelim-roll}$, the machine itself needs to be longer than the length of the part to properly roll it and so is approximated to need to be the part's length multiplied by a factor of 1.5. Another factor of 1.2 times the part length is added to be able to catch the part after it is done being rolled but before it is moved over to be stored. Additionally, an extra 8 feet is added to the length to account for the roll of material behind the machine used to create the stringer that can be seen in listings such as [286] but not in [285]. Finally, 3 feet of clearance is added on all sides to allow workers to safely traverse around the equipment. The resulting preliminary length and width given the part length l and the part width w is shown below:

$$L_{prelim-roll} = 2.7l + 14 \tag{E.27}$$

$$W_{prelim-roll} = 1.25w + 11 \tag{E.28}$$

Parts in the queue add their width to the queue width as described by Equation E.4 because the parts in the queue must be stored to the side of the machine lest they obstruct

it. This occurs for all parts in the queue.

E.7 Water Jet Trimming Machine

The data for water jet cutting machines shown in Table E.2 for some of the composite trimming tasks was obtained from the online used equipment market for Omax water jet cutters. As the cost of new equipment is desired, a way to scale up the cost of used equipment is needed. John in his article [287] shows that used CNC equipment is often 3 to 5 times less expensive than new ones, which is used to scale up the cost of the used water jet equipment. The data in Table E.2 assumes new equipment is being used.

Table E.2: Water Jet Trimming Machine Based on Data from Online Used Equipment Market for Omax Water Jet Cutters

Platform Size (ft^2)	**Cost (Thousands 2022 US Dollars)**
72 (12' $\times$ 6')	400
520 (40' $\times$ 13')	1,200

Because the equipment data is based off of platform size, the size of the platform given the part's dimensions is scaled using Equation E.11 and Equation E.12. A linear cost relationship is assumed between the water jet cutter's cost C_{wj} and its platform's length $l_{platform-wj}$ and width $W_{platform-wj}$, shown in the regression below.

$$C_{wj} = 271428.6 + 1785.7 * w_{platform-wj} * l_{platform-wj} \tag{E.29}$$

The water jet cutter's dimensions are based on the Omax 160X, which one of the online used equipment market's listings was for [288]. The 160X's cutting envelope, which is where the part can actually be cut, is assumed to be the effective platform size despite the official listing citing a larger number for the cutting table. Based on this, the water jet cutter's length and width are each 9 feet larger than the effective platform's size. An extra 4

feet of clearance is added on all sides for handling space. The water jet cutter's preliminary length $L_{prelim-wj}$ and width $W_{prelim-wj}$ as a function of its parts' length l and width w are thus:

$$L_{prelim-wj} = 1.25l + 17 \tag{E.30}$$

$$W_{prelim-wj} = 1.25w + 17 \tag{E.31}$$

For the part queue, it is most efficient to stack the parts sideways as compared to how they are oriented when the machine is operating on them. The parts are arranged in front of the machine and affect its queue length $l_{queue-wj}$ via their part width w_{part}, show below. All parts in the queue add to this length.

$$l_{queue-wj} = 1.25w_{part} + 6 \tag{E.32}$$

E.8 Anodizing

The anodizing equipment is used for the all the anodization tasks for the metallic parts. Korn states that large anodizing equipment sets cost about $500,000 in 2022 while some original equipment manufacturers have listings that go up to %1,500,000 in 2022 for enormous anodizing equipment that is 275 feet in length and 45 feet in width [289]. Based on the anodization process described in section A.3 and on schematics of anodizing equipment sets that can be found on the online used equipment market [290], the anodizing equipment for the F-86 will not exceed 100 feet in length. Based on all this, and because of the lack of being able to attach appropriately sized equipment to the associated costs, it is assumed that the baseline cost conservatively starts at $500,000 and scales up to be $1,000,000 for the system referenced in [290] that has 35 feet long anodizing tanks. The height of the anodic tanks, which determines the width of the parts that can be inserted, has no effect on

the size footprint of the anodizing equipment. The width of the anodic tanks, according to [290], is 3 feet, which is larger than the height or thickness dimensions of any of the parts for any reasonable configuration and so will not change. Due to this, it is assumed the cost of the anodizing equipment C_{anod} is determined primarily and linearly by its tanks' part length and, thus, the parts' length l. It is hazardous for the part to barely fit in the tank length-wise and so the length is multiplied by a factor of 1.25, resulting in the following cost regression:

$$C_{anod} = 500000 + 14285.7 * 1.25l \tag{E.33}$$

The anodizing system described in [290] is used as reference for the preliminary length and width. The length direction is initiated by a 25 feet long area to load and unload the part. In front of this is another area 5 feet long in length used to attach the part onto a hoist that will raise and lower it into the tanks. There are 3 sets of two tanks, each tank being 3 feet wide and separated by a 2 feet long gap, with there also being a 2.5 feet wide walkway in between each set of tanks. Each set of tanks totals up to being 10.5 feet in length. At the end of the tanks is a 16 feet long area used to unload the part. Due to the width of the tanks being set, the preliminary length $L_{prelim-anod}$ is set to being 77.5 feet long. In the width direction, there is a 12 feet wide aisle to shuttle the part from the unloading zone back into the queue space, a 4 feet wide walkway that separate the tanks and the aisle, the tanks themselves that are 1.25 times the part's length, a 3 feet wide trench, and a 13 feet wide area used to store the necessary boilers, rectifiers, and water conditioners for the anodization. The handling and loading/unloading space are included by default. This totals up into the preliminary length $L_{prelim-anod}$ and width $W_{prelim-anod}$ as a function of part length l shown below:

$$L_{prelim-anod} = 77.5 \tag{E.34}$$

$$W_{prelim-anod} = 1.25l + 32 \tag{E.35}$$

For the part queuing, the parts are arranged lengthwise down the length of the station. Because the parts are stored vertically on their side, they only add 8 feet of width to the queue width when they are in the queue, which is primarily to account for the space occupied by the tooling used to hold them along with some clearance room. Due to the very long length of the anodizing equipment, two parts can be stored down the length, meaning only every second part adds to the queue width $w_{queue-anod}$. Every part that does add to the queue width uses the following expression as a result:

$$w_{queue-anod} = 8 \tag{E.36}$$

E.9 Chromate Spray

Chromate spray equipment is used for the chromate spray tasks for the CNC Machining process. The primary cost of the chromate spray equipment is in the paint booth required to house the part while it is being sprayed. The data for the paint booth's cost as a function of its internal volume is gathered from various supplier listings [291, 292, 293] and is shown in Figure E.2.

To obtain a cost estimating regression for the paint booth, how the parts fit inside it must be determined. The parts are oriented vertically on their side, meaning the paint booths do not need to be excessively wide. 8 feet is estimated to be needed on both sides of the part, yielding a total, set width of 16 feet that should be sufficient for workers to move around the hanging part and also matches the widths of the booths in the supplier listings. The booth is estimated to need ten feet of clearance between its ceiling and the tip of the part's side to have space to hang hoists to hold up the part. An additional height of 1.25 times the part's width is added to have the parts fit properly regardless of their aspect ratios or sweep

Internal Volume (ft^3)	Cost (Thousands 2022 US Dollars)
2490	16.5
2530	17.1
3280	23.0
7680	54.0
10240	66.0
12800	80.6
7780	44.1
10370	51.7
12960	58.9
15550	65.9

Figure E.2: Data on Paint Booth Cost as a Function of Internal Volume, Obtained from Various Supplier Listings [291, 292, 293]

angles. For the booth's length, it is assumed that 10 feet of clearance in front of and behind the part is needed to store any additional tools as well as to have space to maneuver around the part, on top of the part's length multiplied by a factor of 1.25 to account for the part itself. A linear relationship between cost and internal volume is assumed, with the booth's cost C_{spray} as a function of the part's length l and width w shown below:

$$C_{spray} = 10172 + 4.39 * 16 * (1.25l + 20) * (1.25w + 10) \tag{E.37}$$

For the preliminary dimensions of the station, 3 feet of clearance is added to all sides of the booth to have additional handling space outside of the booth, resulting in the following preliminary length $L_{prelim-spray}$ and width $W_{prelim-spray}$ as a function of just part length l, since the part width only contributes to the height requirement:

$$L_{prelim-spray} = 1.25l + 26 \tag{E.38}$$

$$W_{prelim-spray} = 22 \tag{E.39}$$

The queued parts are stored lengthwise parallel to the booth but outside the booth and still hang vertically from their tooling. Due to this, they only add 8 feet of width to the queue width when they are in the queue. Every part in queue adds their 8 feet of required width to the queue width $w_{queue-spray}$ in the manner shown below:

$$w_{queue-spray} = 8 \tag{E.40}$$

E.10 Dye Penetrant Testing

The dye penetrant testing equipment is used for every NDI task for the CNC Machining process. The equipment type needed is of the variety where parts are dipped in tanks of dye and developer, akin to how the anodization is done and for which there are few cost estimations. Because the dye penetrant equipment needs only 3 tanks of solution as compared to the anodization's 6, and because the schematic for the listing shown in [290] also includes dye penetrant equipment, it is assumed its cost is half that of the anodizing equipment's, i.e., a base cost of $250,000 in 2022 and a cost of $500,000 for 35 feet long tanks. Similarly, the listing shown in [290] is assumed to have representative dimensions for dye penetrant equipment, meaning the part's width plays no role in determining the cost of the equipment C_{dye}, only the length l. A linear relationship is assumed, similar to the anodization equipment's cost but at half the rate, resulting in the below expression:

$$C_{dye} = 250000 + 7142.9 * 1.25l \tag{E.41}$$

For the preliminary length and width, the dye penetrant takes many of the same dimensional cues as the anodization equipment. The length starts out with a 25 feet long area to load the part and ends with a 16 feet long area to unload the part. In between, based on the schematic in [290], there is 44 feet of length between the beginning of the loading hoist to the end of the inspection room. This means the preliminary length of the penetrant testing

equipment is 85 feet long. For the preliminary width, there is 12 feet of aisle space, 4 feet of walkway adjacent to the tank, and 7 feet of dryer equipment that extends past the inspection room, which is estimated to be the length of the tank, itself the part's length multiplied by 1.25, multiplied by another factor of 1.4. This yields the following preliminary length $L_{prelim-dye}$ and width $W_{prelim-dye}$ as a function of just the part's length l since its width again plays no part:

$$L_{prelim-dye} = 85 \tag{E.42}$$

$$W_{prelim-dye} = 1.4 * 1.25 * l + 23 \tag{E.43}$$

The parts are queued in the exact same way as with the anodization, where they only contribute 8 feet of width to the queue width $w_{queue-dye}$ and only for every other part, resulting in:

$$w_{queue-dye} = 8 \tag{E.44}$$

E.11 Hand Layup

The only equipment needed for the hand layup task is an ultrasonic knife cutting machine for the more automated version of the hand layup task in the VARTM and SRI processes. The more manual layup task in the HLU/Autoclave does not need any equipment. Based on Verrey et al.'s work, an ultrasonic cutting machine 689 ft^2 in size costs about $1.79 million in 2022 [294]. As there are few other public sources of ultrasonic cutting machines' costs along with their correlating sizes, a purely linear relationship between the total size of the machine and the cost is assumed. The machine consists of a cutting table along with a 10 feet long area used to store the rolls of material. The cutting table is assumed to have a length and width 1.25 times greater than the part's length and width, respectively. This

results in the following cost of the ultrasonic cutting machine C_{hlu} as a function of the part's length l and width w:

$$l_{platform-hlu} = 1.25l \tag{E.45}$$

$$w_{platform-hlu} = 1.25w \tag{E.46}$$

$$C_{hlu} = 1790320\frac{w_{platform-hlu} * (l_{platform-hlu} + 10)}{689} \tag{E.47}$$

Though the HLU/Autoclave process's manual layup task does not use the VARTM and SRI process's task's equipment, all the hand layup tasks need the same amount of preliminary space. The layout shown in Verrey et al.'s work [294] is taken to be representative and essentially includes three different tables per hand layup station: one where the plies are cut out from their roll of raw material, one where the plies are set down and kitted after they are cut but before they are placed, and one where the plies are actually laid down for the part. The latter two tables each have lengths and widths that are 1.25 times the part's length and width, respectively, to allow the part to fit properly onto the table. The cutting table has the same size as the ultrasonic cutting equipment, even the more manual layup task, which needs the ten feet of length for its own rolls of materials. To save space, the lengths of the parts are oriented such that they are parallel to the station's width. 10 feet of clearance is added on both sides of the length dimension to provide working room for the workers. 5 feet of clearance is provided on both sides of the widths since the cutting table does not need as much room and the width provided by the cutting table's rolls of materials already provides an initial 5 feet of clearance for the other two tables that need the additional space. The loading and handling space are thus both already included. This results in the following preliminary width $W_{prelim-hlu}$ and length $L_{prelim-hlu}$ as a function of the part's width w and length l:

$$L_{prelim-hlu} = 3.75w + 20 \tag{E.48}$$

$$W_{prelim-hlu} = 1.25l + 20 \tag{E.49}$$

For the part queue, it is most convenient to stack the parts sideways. The parts are arranged such that they affect the queue length $l_{queue-hlu}$ via their part width w_{part}, show below. All parts in the queue add to this length.

$$l_{queue-hlu} = 1.25w_{part} + 6 \tag{E.50}$$

E.12 Hot Drape Form

The hot drape forming equipment is used for the hot draping forming tasks in the ATL process. The data for the hot drape forming equipment is sourced from Hagnell and Akermo [272] and shown in Table E.3, with the middle data point being an interpolation on the size of the machine.

Table E.3: Hot Drape Forming Machine Cost, Adapted from Hagnell and Akermo [272]

Equipment Main Body Size (ft^2)	**Cost (Thousands 2022 US Dollars)**
116	294
161	554
237	912

The data is in terms of the size of the main hot drape forming machine and so how that relates to the actual part size must be determined for the cost regression. Hagnell and Akermo obtained their data from the LT Machines company and it is assumed that the data is applicable to the HDF 6 machine, which is assumed to be representative of the machines used for the F-86 experimental platform. Based on the available schematics

for the HDF 6, the distance from the long direction edge of the supporting platform on either side to the edge of the machine is rounded up to be a conservative 3 feet and from the short direction edge to be 2 feet on either side. The long direction is taken to be the length direction and the short direction the width direction. This means the equipment's main body, ignoring the computer, is 6 feet longer than the platform and 4 feet wider. The platform itself is composed of the full platform area and the thermally stable platform area. The thermally stable platform length and width are taken to be 1.25 times the part length and width respectively, so that all portions of the part can be heated appropriately. The full platform size, based on the specs provided, is estimated to have a length and width 1.25 times the length and width of the thermally stable area. It is assumed there is a linear relationship between the cost of the hot drape forming equipment C_{hdf} and its full equipment size, which is represented as a function of its full platform length $l_{platform-hdf}$ and width $w_{platform-hdf}$, which themselves are functions of the part length l and width w. This is all shown below:

$$l_{platform-hdf} = 1.25 * 1.25l \tag{E.51}$$

$$w_{platform-hdf} = 1.25 * 1.25w \tag{E.52}$$

$$C_{hdf} = -283973.4 + 5074(w_{platform-hdf} + 4)(l_{platform-hdf} + 6) \tag{E.53}$$

For the preliminary length and width of the hot drape forming station, the video shown for another LT Machines hot drape former [295] indicates that the platform can extend out its full width plus an additional estimated 3 feet so that parts can be loaded onto it. An extra foot of width is added to the preliminary width to provide distance between the platform and the loading area the part waits in while the platform is being extended, which adds another width equal to 1.25 times the part width. On the length side, since the HDF 6 is

likely slightly smaller than the actual hot drape former to be used, the actual one to be used likely needs a computer similar to the HDF 9 in [295]. This adds an additional 10 feet to the preliminary length to account for the space for the computer and controls. 4 feet of clearance space is added on all sides for the workers to move about and handle all the parts. For more efficient space usage later when the aisle allowances are added, the preliminary length and width up until this point are flipped such that the preliminary length is now the preliminary width and vice versa. This results in the following new preliminary length $L_{prelim-hdf}$ and width $W_{prelim-hdf}$ as a function of the part's length l and width w:

$$W_{prelim-hdf} = 1.5625l + 6 + 10 + 8 \tag{E.54}$$

$$L_{prelim-hdf} = 2 * 1.5625w + 4 + 4 + 1.25w + 8 \tag{E.55}$$

The parts in the part queue, for efficient space utilization, are placed such that their widths w_{part} add to the queue length $l_{queue-hdf}$ as show below. All parts in the queue add to this length.

$$l_{queue-hdf} = 1.25w_{part} + 6 \tag{E.56}$$

E.13 Stitching

The stitching machine is only used for SRI stitching tasks and, even then, only when integral fabrication is used. The only well-documented publicly available cost and size information for stitching machines is from Dow's article [268]. The machine cost $10 million to develop back in the 1990s and was able to stitch 40 feet long by 8 feet wide wing panels. With the tremendous advances in technology since then, it is assumed it will cost about $5 million in 2022 to make the same machine today. Since the machine uses a gantry in a similar style to a CNC machine, it is assumed that its cost varies in the same manner in

that it is based on its platform size. Using the pictures in Dow's article [268] as reference, the width of the platform is approximately the part width multiplied by 1.25. It is similarly assumed the platform length is 1.25 times the part length, meaning a machine with a platform of size 50' x 10' costs $10 million in 2022. Using the same cost scaling relationship as the CNC milling machine in section E.2, the cost of the stitching machine C_{stitch} given a part's length l and width w is represented as:

$$l_{platform-stitch} = 1.25l \tag{E.57}$$

$$w_{platform-stitch} = 1.25w \tag{E.58}$$

$$C_{stitch} = 4295750 + 1408.5 * w_{platform-stitch} * l_{platform-stitch} \tag{E.59}$$

Based on Dow's article, the widest part of the stitching machine extends about 5 feet further out from the edge of the platform the part sits on both sides, meaning the preliminary width is 1.25 times the part width plus an additional 10 feet. An additional platform-sized width amount, or a width equal to 1.25 times the part width, is added to the preliminary width to make room to load and unload the part. For the part length, the original machine is 75 feet long with an assumed 50 feet long platform, so the preliminary length is 1.25 times the part length plus an additional 25 feet. Four feet of additional clearance is added on all sides to provide extra room for workers to move around. This causes the preliminary length $L_{prelim-stitch}$ and width $W_{prelim-stitch}$ as a function of the part's length l and width w to take the form of:

$$L_{prelim-stitch} = 1.25l + 33 \tag{E.60}$$

$$W_{prelim-stitch} = 1.25w + 18 \tag{E.61}$$

For the part queue, the parts are arranged such that they affect the queue length $l_{queue-stitch}$ via their part width w_{part}, show below. All parts in the queue add to this length.

$$l_{queue-stitch} = 1.25w_{part} + 6 \tag{E.62}$$

E.14 Ultrasonic Pulse Echo NDI

An ultrasonic pulse echo NDI machine is used for all the composite NDI tasks. A cost of \$4.13 million in 2022 for such a system is retrieved from Heckwolf's work [270]. The system is closest in nature to Par Systems's LaserUT laser ultrasonic NDI system [261]. It is assumed that Par Systems's NDI system is representative of the NDI machine used and, since there is no size provided for Heckwolf's data, the cost is for the estimated size of the platform underneath the system's gantry. That estimated size is roughly 20 feet by 35 feet, given that the inspection head itself is estimated to be about 7 feet long. In terms of determining the platform size with respect to the part, the platform's width is assumed to be fixed at 20 feet while its length is 1.25 times the part length with an additional 4 feet on both ends so that the inspection head can actually reach the ends of the part, since the inspection head is also limited to moving within the platform. The width is constant because the parts are assumed to be held on their side for the NDI process such that their widths are vertical, meaning a platform width larger than 20 feet is never needed because a 20 feet thick part will not exist. This is represented via the following equations, where l is the part's length and w is the part's width and a purely linear relationship is assumed:

$$l_{platform-ndi} = 1.25l + 8 \tag{E.63}$$

$$w_{platform-ndi} = 20 \tag{E.64}$$

$$C_{ndi} = 4130000 \frac{w_{platform-ndi} * l_{platform-ndi}}{20 * 35} \tag{E.65}$$

For the physical dimensions of the system, the two pillars on either side of the platforms supporting the gantry are estimated to be about 2 feet in width, and there must be a computer operating the system, which is assumed to require an extra 8 feet of width. An additional 3 feet of clearance in all directions is added for space for workers to move around. The NDI machine's preliminary length $L_{prelim-ndi}$ and width $W_{prelim-ndi}$ as a function of its parts' length l and width w are:

$$L_{prelim-ndi} = 1.25l + 14 \tag{E.66}$$

$$W_{prelim-ndi} = 38 \tag{E.67}$$

The parts are all assumed to be held in their tooling in a vertical position while they wait in the part queue. Because they are held in such a manner, they do not add much extra needed width, and so 8 feet is added to the queue width $w_{queue-ndi}$ for each part in the queue to be conservative.

$$w_{queue-ndi} = 8 \tag{E.68}$$

REFERENCES

[1] "Boeing Commercial Market Outlook 2019–2038," The Boeing Company, Tech. Rep., 2019.

[2] "Hexcel Corporation Annual Report 2011," Hexcel Corporation, Tech. Rep., 2011, https://web.archive.org/web/20230630054421/https://s22.q4cdn.com/602714005/files/doc_financials/annual/Hexcel-2011-Annual-Report.pdf.

[3] The Boeing Company, "Boeing commercial orders and deliveries," 2020, http://www.boeing.com/commercial/{#}/orders-deliveries.

[4] Airbus, "Airbus orders and deliveries," 2020, https://www.airbus.com/aircraft/market/orders-deliveries.html.

[5] "Boeing, airbus share honours in 2013 orders deliveries race - but it's not about winners and losers," CAPA - Center for Aviation, 2014.

[6] S. Yu and S. Qiu, "China certifies c919 jet to compete with airbus and boeing," *Reuters*, Sep. 2022, https://www.reuters.com/business/aerospace-defense/china-certifies-c919-jet-compete-with-airbus-boeing-photos-2022-09-29/.

[7] D. N. Mavris, D. A. DeLaurentis, O. Bandte, and M. A. Hale, "A stochastic approach to multi-disciplinary aircraft analysis and design," *36th AIAA Aerospace Sciences Meeting and Exhibit*, Jan. 1998.

[8] J. Sloan, "Airbus completes assembly of Wing of Tomorrow prototype," *Composites World*, 2022, https://web.archive.org/web/20230630052346/https://www.compositesworld.com/news/airbus-completes-assembly-of-wing-of-tomorrow-prototype, Accessed: 2022-09-30.

[9] R. Young, "Hi-Rate Composite Aircraft Manufacturing (HiCAM) Project Overview," *NASA*, 2022, https://web.archive.org/web/20230630052609/https://www.nasa.gov/sites/default/files/atoms/files/hicam-overview-apr-2022.pdf, Accessed: 2022-09-30.

[10] P. Diaz, "Airbus Wants to Increase A320 Production Rate; Its Suppliers, Not So Much," *Airline Geeks*, Apr. 2018, https://airlinegeeks.com/2018/04/27/airbus-wants-to-increase-a320-production-rate-its-suppliers-not-so-much/.

[11] The Boeing Company, "Boeing to increase 737 production rate to 52 per month in 2018," Boeing, 2014.

[12] T. Hepher, "Airbus confirms plans to raise A320 output to 63 a month," *Reuters*, Apr. 2018, https://www.reuters.com/article/us-airbus-production/airbus-confirms-plans-to-raise-a320-output-to-63-a-month-idUSKBN1HW1Z2.

[13] D. Royce, "Airbus A320 Family Production Increases, Driving Demand for APS 3200 APUs," *Defense and Security Monitor*, Aug. 2019, https://dsm.forecastinternational.com/wordpress/2019/08/22/airbus.

[14] T. A. Abdullah, K. Popplewell, and C. J. Page, "A review of the support tools for the process of assembly method selection and assembly planning," *International Journal of Production Research*, vol. 41, no. 11, pp. 2391–2410, Jul. 2003.

[15] R. Holt and C. Barnes, "Towards an integrated approach to "design for X": An agenda for decision-based DFX research," *Research in Engineering Design*, vol. 21, no. 2, pp. 123–136, Apr. 2010.

[16] G. Boothroyd, P. Dewhurst, and W. Knight, *Product Design for Manufacture and Assembly: Second Edition*. Marcel Dekker, Inc., 2002.

[17] P. G. Leaney, "CASE EXPERIENCE WITH HITACHI, LUCAS AND BOOTHROYD-DEWHURST DFA METHODS," in *Design for X: Concurrent Engineering Imperatives*, G. Q. Huang, Ed., Springer Netherlands, 1996, ch. 2, pp. 41–71.

[18] J. Fan and J. Dong, "Intelligent Virtual Assembly Planning with Integrated Assembly Model*," in *SMC'03 Conference Proceedings*, Washington D.C.: IEEE, 2003.

[19] S. Kalpakjian and S. R. Schmid, *MANUFACTURING ENGINEERING AND TECHNOLOGY: SEVENTH EDITION IN SI UNITS*. Pearson Education South Asia Pte Ltd, 2014.

[20] K. Swift, "Expert system aids design for assembly," *Assembly Automation*, vol. 9, no. 3, pp. 132–136, 1989.

[21] R. B. Stone, D. A. McAdams, and R. H. Crawford, "A heuristic method for identifying modules for product architectures," *Design Studies*, vol. 21, no. 1, pp. 5–31, Jan. 2000.

[22] R. B. Stone and D. A. McAdams, "A product architecture-based conceptual DFA technique," *Design Studies*, vol. 25, pp. 325–325, 2004.

[23] F. Bouissiere, C. Cuiller, P.-E. Dereux, C. Malchair, C. Favi, and G. Formentini, "Conceptual Design for Assembly in Aerospace Industry: A Method to Assess Manufacturing and Assembly Aspects of Product Architectures," *Proceedings of*

the Design Society: International Conference on Engineering Design, vol. 1, no. 1, pp. 2961–2970, Jul. 2019.

[24] N. Halfmann and D. Krause, "TOWARDS INNOVATIVE ASSEMBLY CONCEPTS: INTEGRAL PRODUCT-AND ASSEMBLY STRUCTURE," in *INTERNATIONAL DESIGN CONFERENCE - DESIGN 2010*, Dubrovnik - Croatia, 2010.

[25] A. Gómez, J. Ríos, F. Mas, and A. Vizán, "Method and software application to assist in the conceptual design of aircraft final assembly lines," *Journal of Manufacturing Systems*, vol. 40, pp. 37–53, Jul. 2016.

[26] K. Agyapong-Kodua, K. B. Asare, and D. J. Ceglarek, "Digital modelling methodology for effective cost assessment," in *Procedia CIRP*, vol. 17, Elsevier B.V., 2014, pp. 744–749.

[27] A. Caggiano, A. Marzano, and R. Teti, "Resource Efficient Configuration of an Aircraft Assembly Line," in *Procedia CIRP*, vol. 41, Elsevier B.V., 2016, pp. 236–241.

[28] W. J. Marx, D. N. Mavris, and D. P. Schrage, "Effects of alternative wing structural concepts on high speed civil transport life cycle costs," in *37th AIAA /ASME /ASCE /AHS /ASC Structure, Structural Dynamics and Materials Conference*, American Institute of Aeronautics and Astronautics Inc, AIAA, 1996, pp. 562–582.

[29] D. J. Siedlak, O. J. Pinon, P. R. Schlais, T. M. Schmidt, and D. N. Mavris, "A digital thread approach to support manufacturing-influenced conceptual aircraft design," *Research in Engineering Design*, vol. 29, no. 2, pp. 285–308, Apr. 2018.

[30] D. P. Raymer, *Aircraft Design: A Conceptual Approach, Fourth Edition*. American Institute of Aeronautics and Astronautics, Inc., 2006.

[31] C. Favi, M. Germani, and M. Mandolini, "Design for Manufacturing and Assembly vs. Design to Cost: Toward a Multi-objective Approach for Decision-making Strategies during Conceptual Design of Complex Products," in *Procedia CIRP*, vol. 50, Elsevier B.V., 2016, pp. 275–280.

[32] K. Ulrich, "The role of product architecture in the manufacturing firm," *Research Policy*, vol. 24, no. 3, pp. 419–440, May 1995.

[33] G. Pahl, W. Beitz, J. Feldhusen, and K. H. Grote, *Engineering Design: A Systematic Approach-Third Edition*. Springer, 1998.

[34] T. L. D. Fazio, S. J. Rhee, and D. E. Whitney, "Design-specific approach to design for assembly (dfa) for complex mechanical assemblies," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 5, pp. 869–881, 1999.

[35] N. P. Suh, *The Principles of Design*. Oxford University Press, 2007.

[36] C. Favi and M. Germani, "A method to optimize assemblability of industrial product in early design phase: From product architecture to assembly sequence," *International Journal on Interactive Design and Manufacturing*, vol. 6, pp. 155–169, 2012.

[37] S. Takai, "An approach to integrate product and process design using augmented liaison diagram, assembly sequencing, and assembly line balancing," *Journal of Mechanical Design, Transactions of the ASME*, vol. 143, no. 10, Oct. 2021.

[38] K. Amann, *Product lifecycle management: empowering the future of business*. CIM Data, Inc., 2002.

[39] R. Sudarsan, S. J. Fenves, R. Sriram, and F. Wang, "A product information modeling framework for product lifecycle management," *Computer-Aided Design*, vol. 37, pp. 1399–1411, Feb. 2005.

[40] F. Demoly, X. T. Yan, B. Eynard, L. Rivest, and S. Gomes, "An assembly oriented design framework for product structure engineering and assembly sequence planning," *Robotics and Computer-Integrated Manufacturing*, vol. 27, no. 1, pp. 33–46, Feb. 2011.

[41] F. Demoly, L. Toussaint, B. Eynard, D. Kiritsis, and S. Gomes, "Geometric skeleton computation enabling concurrent product engineering and assembly sequence planning," *CAD Computer Aided Design*, vol. 43, no. 12, pp. 1654–1673, Dec. 2011.

[42] F. Demoly, D. Monticolo, B. Eynard, L. Rivest, and S. Gomes, "Multiple viewpoint modelling framework enabling integrated product-process design," *International Journal on Interactive Design and Manufacturing*, vol. 4, no. 4, pp. 269–280, Nov. 2010.

[43] F. Demoly, N. Troussier, B. Eynard, H. Falgarone, B. Fricero, and S. Gomes, "Proactive Assembly Oriented Design Approach Based on the Deployment of Functional Requirements," *Journal of Computing and Information Science in Engineering*, vol. 11, no. 1, p. 014 501, Apr. 2011.

[44] F. Demoly, O. Dutartre, X. T. Yan, B. Eynard, D. Kiritsis, and S. Gomes, "Product relationships management enabler for concurrent engineering and product lifecycle management," *Computers in Industry*, vol. 64, no. 7, pp. 833–848, Sep. 2013.

[45] F. Mas, A. Gómez, J. Menéndez, and J. Ríos, "Proposal for the Conceptual Design of Aeronautical Final Assembly Lines Based on the Industrial Digital Mock-Up Concept," in *10th Product Lifecycle Management for Society (PLM)*, Nantes, France, Jul. 2013, pp. 10–19.

[46] J. Ortegón, F. Mas, A. Gomez-Parra, and M. Marcos, "Virtual improvement of a historical aircraft assembly line," *Procedia Manufacturing*, vol. 13, pp. 1312–1319, 2017.

[47] M. Schönemann, S. Thiede, and C. Herrmann, "Integrating product characteristics into extended value stream modeling," in *Procedia CIRP*, vol. 17, Elsevier B.V., 2014, pp. 368–373.

[48] *Systems Engineering and PLM Basics*, https://www.plmportal.org/en/systems- engineering -and-plm-basics.html, 2020.

[49] T. Li, H. Lockett, and C. Lawson, "Using requirement-functional-logical-physical models to support early assembly process planning for complex aircraft systems integration," *Journal of Manufacturing Systems*, vol. 54, pp. 242–257, Jan. 2020.

[50] T. Polacsek, S. Roussel, F. Bouissiere, C. Cuiller, P. E. Dereux, and S. Kersuzan, "Towards thinking manufacturing and design together: An aeronautical case study," in *International Conference on Conceptual Modeling*, Springer International Publishing, 2017, pp. 340–353, ISBN: 9783319699035.

[51] T. Polacsek *et al.*, "A Scheduling Tool for Bridging the Gap Between Aircraft Design and Aircraft Manufacturing," in *Twenty-Eighth International Conference on Automated Planning and Scheduling (ICAPS 2018)*, Delft, Netherlands, 2018.

[52] D. Schrage, D. Mavris, and J. Craig, "Nasa-cr-195511: Integrated design and manufacturing for the high speed civil transport," NASA, 1993.

[53] D. P. Schrage, "Systems Engineering Process (Boeing's 11 Step Process)," Notes for GT Class AE 6372: Aerospace Systems Engineering.

[54] D. P. Schrage, "Technology for Rotorcraft Affordability Through Integrated Product/Process Development (IPPD)," 1999, Technical Report.

[55] J. Cormey, L. Demory, J. Kenney, C. Reeder, Z. Schaffer, and L. Windhoffer, "Final project report for the f-86 stratosabre," 2007, Final Report for GT Class AE 6372: Aerospace Systems Engineering.

[56] D. P. Schrage, *Encyclopedia of Aerospace Engineering – Product Lifecycle Engineering (PLE): An Application*. John Wiley & Sons, Ltd, Dec. 2010.

[57] D. P. Schrage and A. Sirirojvisuth, "Boeing IDM through PLM Course 2: PLE through PLM Overview and NASA –Boeing Composite Wing Case Study," 2012, Notes for GT Class AE 6372: Aerospace Systems Engineering.

[58] A. Sirirojvisuth, "Development of hybrid lifecycle cost estimating tool (hlcet) for manufacturing influenced design tradeoff," Ph.D. dissertation, Georgia Institute of Technology, 2012.

[59] S. Phelan, A. Deschenes, T. Lee, and H. Koinuma, "Manufacturing Influenced Design (MInD) IPPD Process," 2011, Final Presentation for GT Class AE 6372: Aerospace Systems Engineering.

[60] J. Ceisel, T. Carr, S. Pogaru, and D. N. Mavris, "A NON-WEIGHT BASED, MANUFACTURING INFLUENCED DESIGN (MIND) METHODOLOGY FOR PRELIMINARY DESIGN," in *28TH INTERNATIONAL CONGRESS OF THE AERONAUTICAL SCIENCES*, Brisbane, Australia, 2012.

[61] C. Sundaresan, Z. Tang, J. Ceisel, and D. N. Mavris, "A Methodology For Parametric Production Planning in Preliminary Aircraft Design," in *28TH INTERNATIONAL CONGRESS OF THE AERONAUTICAL SCIENCES*, Brisbane, Australia, 2012.

[62] D. J. L. Siedlak, T. M. Schmidt, O. J. Pinon, and D. N. Mavris, "A Methodology for the Parametric Exploration of the Impact of Production Planning on the Early Stages of Design," in *Proceedings of the ASME 2014 International Manufacturing Science and Engineering Conference*, Detroit, Michigan: ASME International, Jun. 2014.

[63] D. J. L. Siedlak, P. R. Schlais, O. J. Pinon, and D. N. Mavris, "SUPPORTING AFFORDABILITY-BASED DESIGN DECISIONS IN THE PRESENCE OF DEMAND VARIABILITY," in *Proceedings of the ASME 2015 International Manufacturing Science and Engineering Conference*, Charlotte, North Carolina: ASME International, 2015.

[64] J. Butterfield *et al.*, "An Integrated Approach to the Conceptual Development of Aircraft Structures Focusing on Manufacturing Simulation and Cost," in *AIAA 5th Aviation, Technology, Integration, and Operations Conference*, Arlington, VA: American Institute of Aeronautics and Astronautics (AIAA), Sep. 2005.

[65] Y. Jin, R. Curran, J. Butterfield, R. Burke, and B. Welch, "Intelligent assembly time analysis using a digital knowledge-based approach," *Journal of Aerospace Computing, Information and Communication*, vol. 6, no. 8, pp. 506–522, Aug. 2009.

[66] I. Friel, A. Marzano, and J. Butterfield, "Developing an intelligent digital mock up to integrate design and manufacturing disciplines," in *32nd International Manufacturing Conference*, 2021.

[67] M. Price, S. Raghunathan, and R. Curran, "An integrated systems engineering approach to aircraft design," *Progress in Aerospace Sciences*, vol. 42, no. 4, pp. 331–376, Jun. 2006.

[68] A. Joneja, "Design For Manufacturing/Assembly Course Notes," Hong Kong University of Science and Technology, Hong Kong, Tech. Rep., 2017.

[69] R. Curran, A. Kundu, S. Raghunathan, D. Eakin, and R. McFadden, "Influence of manufacturing tolerance on aircraft direct operating cost (doc)," in *Journal of Materials Processing Technology*, vol. 138, Jul. 2003, pp. 208–213.

[70] K. S. Bhachu, G. Waycaster, R. T. Haftka, and N. H. Kim, "Aircraft tolerance optimization considering quality, manufacturing, and performance," in *54th AIAA /ASME /ASCE /AHS /ASC Structures, Structural Dynamics, and Materials Conference*, 2013, ISBN: 9781624102233.

[71] H.-G. R. Choi, M.-H. Park, and E. Salisbury, "Optimal tolerance allocation with loss functions," *Journal of Manufacturing Science and Engineering*, vol. 122, no. 3, pp. 529–535, Aug. 2000.

[72] Z. Zhao *et al.*, "Prediction of Assembly Variation During Early Design," in *Volume 4: ASME/IEEE International Conference on Mechatronic and Embedded Systems and Applications and the 19th Reliability, Stress Analysis, and Failure Prevention Conference*, ASME, 2007, pp. 891–900, ISBN: 0-7918-4805-1.

[73] L. Capacho, R. Pastor, A. Dolgui, and O. Guschinskaya, "An evaluation of constructive heuristic methods for solving the alternative subgraphs assembly line balancing problem," *Journal of Heuristics*, vol. 15, no. 2, pp. 109–132, Apr. 2009.

[74] J. Bukchin and M. Tzur, "Design of flexible assembly line to minimize equipment cost," *IIE Transactions (Institute of Industrial Engineers)*, vol. 32, no. 7, pp. 585–598, 2000.

[75] M. Chica, Ó. Cordón, and S. Damas, "An advanced multiobjective genetic algorithm design for the time and space assembly line balancing problem," *Computers and Industrial Engineering*, vol. 61, no. 1, pp. 103–117, Aug. 2011.

[76] J. Oesterle, L. Amodeo, and F. Yalaoui, "A comparative study of Multi-Objective Algorithms for the Assembly Line Balancing and Equipment Selection Problem under consideration of Product Design Alternatives," *Journal of Intelligent Manufacturing*, vol. 30, no. 3, pp. 1021–1046, Mar. 2019.

[77] Y. J. Tseng, J. Y. Chen, and F. Y. Huang, "A multi-plant assembly sequence planning model with integrated assembly sequence planning and plant assignment using GA," *International Journal of Advanced Manufacturing Technology*, vol. 48, no. 1-4, pp. 333–345, Apr. 2010.

[78] Y. J. Tseng, J. Y. Chen, and F. Y. Huang, "A particle swarm optimisation algorithm for multi-plant assembly sequence planning with integrated assembly sequence planning and plant assignment," *International Journal of Production Research*, vol. 48, no. 10, pp. 2765–2791, Jan. 2010.

[79] O. Battaïa, X. Delorme, A. Dolgui, B. Finel, F. Metz, and F. Grimaud, "Flow line balancing problem: A survey," in *6th IESM Conference*, Oct. 2015.

[80] P. Jiménez, "Survey on assembly sequencing: A combinatorial and geometrical perspective," *Journal of Intelligent Manufacturing*, vol. 24, no. 2, pp. 235–250, Apr. 2013.

[81] M. A. Abdullah, M. F. F. Ab Rashid, and Z. Ghazalli, "Optimization of Assembly Sequence Planning Using Soft Computing Approaches: A Review," *Archives of Computational Methods in Engineering*, vol. 26, no. 2, pp. 461–474, Apr. 2019.

[82] L. S. H. de Mello and A. C. Sanderson, "Representation of Mechanical Assembly Sequences," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 2, 1991.

[83] A. Bourjault, "Contribution à une approche méthodologique de l'assemblage automatisé: élaboration automatique des séquences opératoire," Ph.D. dissertation, Universite de Franche-Compte, 1984.

[84] L. H. de Mello and A. C. Sanderson, "AND/OR Graph Representation of Assembly Plans," *IEEE Transactions on Robotics and Automation*, vol. 6, no. 2, 1990.

[85] T. L. De Fazio and D. Whitney, "Simplified Generation of All Mechanical Assembly Sequences," *IEEE JOURNAL OF ROBOTICS AND AUTOMATION*, vol. 3, no. 6, 1987.

[86] H.-J. Bullinger and E. D. Ammer, "COMPUTER-AIDED DEPICTING OF PRECEDENCE DIAGRAMS-A STEP TOWARDS EFFICIENT PLANNING IN ASSEMBLY," *Computing and Industrial Engineering*, vol. 8, no. 3, pp. 165–169, 1984.

[87] Y. Wang and J. H. Liu, "Chaotic particle swarm optimization for assembly sequence planning," *Robotics and Computer-Integrated Manufacturing*, vol. 26, no. 2, pp. 212–222, Apr. 2010.

[88] S. Ghandi and E. Masehian, "Review and taxonomies of assembly and disassembly path planning problems and approaches," vol. 67-68, pp. 58–86, Jun. 2015.

[89] A. J. Lambert, "Generating disassembly sequences using exact and heuristic methods applied to disassembly precedence graphs," in *The 6th IEEE International Symposium on Assembly and Task Planning: From Nano to Macro Assembly and Manufacturing, 2005*, 2005, pp. 47–52, ISBN: 0780390806.

[90] J. R. Li, L. P. Khoo, and S. B. Tor, "A Novel Representation Scheme for Disassembly Sequence Planning," *Int J Adv Manuf Technol*, vol. 20, pp. 621–630, 2002.

[91] G. Dini and M. Santochi, "Automated Sequencing and Subassembly Detection in Assembly Planning," *Annals of the CIRP*, vol. 41, no. 1, 1992.

[92] H.-E. Tseng and R.-K. Li, "A novel means of generating assembly sequences using the connector concept," *Journal of Intelligent Manufacturing*, vol. 10, pp. 423–435, 1999.

[93] L. H. de Mello and A. C. Sanderson, "A Correct and Complete Algorithm for the Generation of Mechanical Sequences," *IEEE Transactions an Robotics and Automation*, vol. 7, no. 2, 1991.

[94] R. Mantripragada and D. E. Whitney, "The Datum Flow Chain: A Systematic Approach to Assembly Design and Modeling," *Research in Engineering Design*, vol. 10, pp. 50–165, 1998.

[95] R. Mantripragada, T. W. Cunningham, D. E. Whitney, M. Cadlab, and M. Ave, "Assembly Oriented Design: A new approach to designing assemblies," in *Product Modeling for Computer Integrated Design and Manufacture*, M. Pratt, R. Sriram, and M. Wozny, Eds., Springer, Boston, MA, 1997, ch. 26, pp. 308–324.

[96] M. Goldwasser, "COMPLEXITY MEASURES FOR ASSEMBLY SEQUENCES," Ph.D. dissertation, Stanford University, 1997.

[97] M. F. F. Rashid, W. Hutabarat, and A. Tiwari, "A review on assembly sequence planning and assembly line balancing optimisation using soft computing approaches," *International Journal of Advanced Manufacturing Technology*, vol. 59, no. 1-4, pp. 335–349, Mar. 2012.

[98] J. Wolter, "On the Automatic Generation of Assembly Plans," in *Proceedings of the 1989 International Conference on Robotics and Automation*, Scottsdale, AZ: IEEE, 1989.

[99] D. Baldwin, T. Abell, M.-C. M. Lui, T. L. De Fazio, and D. E. Whitney, "An Integrated Computer Aid for Generating and Evaluating Assembly Sequences for Mechanical Products," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 1, 1991.

[100] L. Laperrire and H. ElMaraghy, "GAPP: A Generative Assembly Process Planner," *Journal of Manufacturing Systems*, vol. 15, no. 4, 1996.

[101] N. Panhalkar, R. Paul, and S. Anand, "Optimization of automobile assembly process to reduce assembly time," *Computer-Aided Design and Applications*, vol. 11, S54–S60, 2014.

[102] Y. Wang and De Tian, "A weighted assembly precedence graph for assembly sequence planning," *International Journal of Advanced Manufacturing Technology*, vol. 83, no. 1-4, pp. 99–115, 2016.

[103] R. Bahubalendruni, "Computer Aided Optimal Robotic Assembly Sequence Generation," Ph.D. dissertation, National Institute of Technology Rourkela, 2016.

[104] S. P. Leo Kumar, "Knowledge-based expert system in manufacturing planning: state-of-the-art review," *International Journal of Production Research*, vol. 57, no. 15-16, pp. 4766–4790, 2019.

[105] A. Kusiak, "Process Planning: A knowledge-Based and Optimization Approach," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 3, 1991.

[106] Z. Dong and W. Hu, "Candidate Machine Sequence Generation for Optimal Process Planning Using a Knowledge-Based System," in *IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing*, Victoria, BC, Canada, 1991.

[107] A. Swaminathan and K. S. Barber, "An Experience-Based Assembly Sequence Planner for Mechanical Assemblies," *IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION*, vol. 12, no. 2, 1996.

[108] M. Kashkoush and H. ElMaraghy, "Knowledge-based model for constructing master assembly sequence," *Journal of Manufacturing Systems*, vol. 34, no. C, pp. 43–52, 2015.

[109] X. F. Zha, H. J. Du, and J. H. Qiu, "Knowledge-based approach and system for assembly oriented design, Part I: the approach," *Engineering Applications of Artificial Intelligence*, vol. 14, pp. 61–75, 2001.

[110] Z. Yin, H. Ding, H. Li, and Y. Xiong, "A connector-based hierarchical approach to assembly sequence planning for mechanical assemblies," *Computer Aided Design*, vol. 35, pp. 37–56, 2003.

[111] J. Li, Q. Wang, P. Huang, and H. Shen, "A novel connector-knowledge-based approach for disassembly precedence constraint generation," *International Journal of Advanced Manufacturing Technology*, vol. 49, no. 1-4, pp. 293–304, Jul. 2010.

[112] Y. Y. Hsu, P. H. Tai, M. W. Wang, and W. C. Chen, "A knowledge-based engineering system for assembly sequence planning," *International Journal of Advanced Manufacturing Technology*, vol. 55, no. 5-8, pp. 763–782, Jul. 2011.

[113] A. Mishra and S. Deb, "An intelligent methodology for assembly tools selection and assembly sequence optimisation," *Lecture Notes in Mechanical Engineering*, pp. 323–333, 2016.

[114] L. Qiao, Y. Qie, Z. Zhu, Y. Zhu, U. K. uz Zaman, and N. Anwer, "An ontology-based modelling and reasoning framework for assembly sequence planning," *International Journal of Advanced Manufacturing Technology*, vol. 94, no. 9-12, pp. 4187–4197, Feb. 2018.

[115] Y. Zhong, C. Jiang, Y. Qin, G. Yang, M. Huang, and X. Luo, "Automatically generating assembly sequences with an ontology-based approach," *Assembly Automation*, vol. 40, no. 2, pp. 319–334, Nov. 2019.

[116] A. Neb, "Review on approaches to generate assembly sequences by extraction of assembly features from 3d models," in *Procedia CIRP*, vol. 81, Elsevier B.V., 2019, pp. 856–861.

[117] H. Ko and K. Lee, "Automatic assembling procedure generation from mating conditions," *Computer Aided Design*, vol. 19, no. 1, 1987.

[118] A. C. Lin and T.-C. Chang, "3D MAPS: Three-Dimensional Mechanical Assembly Planning System," *Journal of Manufacturing Systems*, vol. 12, no. 6, 1993.

[119] R. H. Wilson and J.-C. Latombe, "Geometric reasoning about mechanical assembly," *Artificial Intellgience*, vol. 71, pp. 371–396, 1994.

[120] B. Romney, C. Godard, M. Goldwasser, and G. Ramkumar, "AN EFFICIENT SYSTEM FOR GEOMETRIC ASSEMBLY SEQUENCE GENERATION AND EVALUATION," in *Proceedings of the 1995 ASME International Computers in Engineering Conference*, 1995, pp. 699–712.

[121] T.-H. Eng, Z.-K. Ling, W. Olson, and C. Mclean, "Feature-based assembly modeling and sequence generation," *Computers & Industrial Engineering*, vol. 36, pp. 17–33, 1999.

[122] D. Hu, Y. Hu, and C. Li, "Mechanical Product Disassembly Sequence and Path Planning Based on Knowledge and Geometric Reasoning," *Int J Adv Manuf Technol*, vol. 19, pp. 688–696, 2002.

[123] Q. Su, "Computer aided geometric feasible assembly sequence planning and optimizing," *International Journal of Advanced Manufacturing Technology*, vol. 33, no. 1-2, pp. 48–57, 2007.

[124] W. Zhang, M. Ma, H. Li, and J. Yu, "Generating interference matrices for automatic assembly sequence planning," *International Journal of Advanced Manufacturing Technology*, vol. 90, no. 1-4, pp. 1187–1201, Apr. 2017.

[125] W. Wan, K. Harada, and K. Nagata, "Assembly Sequence Planning for Motion Planning," *Assembly Automation*, vol. 38, no. 2, Sep. 2016. arXiv: 1609.03108.

[126] C. Pan, S. S. Smith, and G. C. Smith, "Determining interference between parts in CAD STEP files for automatic assembly planning," *Journal of Computing and Information Science in Engineering*, vol. 5, no. 1, pp. 56–62, Mar. 2005.

[127] C. Pan, S. S. Smith, and G. C. Smith, "Automatic assembly sequence planning from STEP CAD files," *International Journal of Computer Integrated Manufacturing*, vol. 19, no. 8, pp. 775–783, Dec. 2006.

[128] S. Zhanlei, H. Pengfei, and Z. Gang, "Research on aircraft assembly sequence planning technology based on key characteristics," in *Applied Mechanics and Materials*, vol. 328, 2013, pp. 9–16, ISBN: 9783037857120.

[129] C. L. P. Chen and Y.-H. Pao, "An Integration of Neural Network and Rule-Based Systems for Design and Planning of Mechanical Assemblies," in *IEEE TRANSACTIONS ON SYSTEMS. MAN. AND CYBERNETICS*, vol. 23, 1993.

[130] D. S. Hong and H. S. Cho, "A Neural-network-based Computational Scheme for Generating Optimized Robotic Assembly Sequences," *Engineering Applications of Artificial Intelligence*, vol. 8, no. 2, pp. 129–145, 1995.

[131] W. C. Chen, P. H. Tai, W. J. Deng, and L. F. Hsieh, "A three-stage integrated approach for assembly sequence planning using neural networks," *Expert Systems with Applications*, vol. 34, no. 3, pp. 1777–1786, Apr. 2008.

[132] J. M. Milner, S. C. Graves, and D. E. Whitney, "Using simulated annealing to select least-cost assembly sequences," in *Proceedings - IEEE International Conference on Robotics and Automation*, Publ by IEEE, 1994, pp. 2058–2062, ISBN: 0818653329.

[133] D. S. Hong and H. S. Cho, "Generation of robotic assembly sequences with consideration of line balancing using simulated annealing," *Robotica*, vol. 15, no. 6, pp. 663–673, 1997.

[134] S. Motavalli and A.-U. Islam, "MULTI-CRITERIA ASSEMBLY SEQUENCING," *Computers & Industrial Engineering*, vol. 32, no. 4, pp. 743–751, 1997.

[135] F. Bonneville, C. Perrard, and J. M. Henrioud, "A GENETIC ALGORITHM TO GENERATE AND EVALUATE ASSEMBLY PLANS," in *Proceedings 1995 INRIA/IEEE Symposium on Emerging Technologies and Factory Automation*, Paris, France: IEEE, 1995.

[136] D. S. Hong and H. S. Cho, "A genetic-algorithm-based approach to the generation of robotic assembly sequences," *Control Engineering Practice*, vol. 7, pp. 151–159, 1999.

[137] P. De Lit, P. Latinne, B. Rekiek, and A. Delchambre, "Assembly planning with an ordering genetic algorithm," *International Journal of Production Research*, vol. 39, no. 16, pp. 3623–3640, Nov. 2001.

[138] S. F. Chen and Y. J. Liu, "An adaptive genetic assembly-sequence planner," *International Journal of Computer Integrated Manufacturing*, vol. 14, no. 5, pp. 489–500, Sep. 2001.

[139] G. C. Smith and S. S.-F. Smith, "An enhanced genetic algorithm for automated assembly planning," *Robotics and Computer Integrated Manufacturing*, vol. 18, pp. 355–364, 2002.

[140] H. E. Tseng, J. D. Li, and Y. H. Chang, "Connector-based approach to assembly planning using a genetic algorithm," *International Journal of Production Research*, vol. 42, no. 11, pp. 2243–2261, Jun. 2004.

[141] R. M. Marian, L. H. Luong, and K. Abhary, "A genetic algorithm for the optimisation of assembly sequences," *Computers and Industrial Engineering*, vol. 50, no. 4, pp. 503–527, Aug. 2006.

[142] H. E. Tseng, M. H. Chen, C. C. Chang, and W. P. Wang, "Hybrid evolutionary multi-objective algorithms for integrating assembly sequence planning and assembly line balancing," *International Journal of Production Research*, vol. 46, no. 21, pp. 5951–5977, 2008.

[143] Y. K. Choi, D. M. Lee, and Y. B. Cho, "An approach to multi-criteria assembly sequence planning using genetic algorithms," *International Journal of Advanced Manufacturing Technology*, vol. 42, no. 1-2, pp. 180–188, May 2009.

[144] J. F. Wang, J. H. Liu, and Y. F. Zhong, "A novel ant colony algorithm for assembly sequence planning," *International Journal of Advanced Manufacturing Technology*, vol. 25, no. 11-12, pp. 1137–1143, Jun. 2005.

[145] C. Lu, H. Z. Huang, J. Y. Full, and Y. S. Wong, "A multi-objective disassembly planning approach with ant colony optimization algorithm," in *Proceedings of the*

Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, vol. 222, 2008, pp. 1465–1474.

[146] J. Zhang, J. Sun, and Q. He, "An approach to assembly sequence planning using ant colony optimization," in *Proceedings of 2010 International Conference on Intelligent Control and Information Processing, ICICIP 2010*, 2010, pp. 230–233, ISBN: 9781424470488.

[147] J. Yu and C. Wang, "A max-min ant colony system for assembly sequence planning," *International Journal of Advanced Manufacturing Technology*, vol. 67, no. 9-12, pp. 2819–2835, 2013.

[148] H. Wang, Y. Rong, and D. Xiang, "Mechanical assembly planning using ant colony optimization," *CAD Computer Aided Design*, vol. 47, pp. 59–71, 2014.

[149] C. Lu and Z. Yang, "Integrated assembly sequence planning and assembly line balancing with ant colony optimization approach," *International Journal of Advanced Manufacturing Technology*, vol. 83, no. 1-4, pp. 243–256, 2016.

[150] J. Liu, Y. Wang, and Z. Gu, "GENERATION OF OPTIMAL ASSEMBLY SEQUENCES USING PARTICLE SWARM OPTIMIZATION," in *ASME 2008 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Brooklyn, New York, 2008.

[151] H. G. Lv and C. Lu, "An assembly sequence planning approach with a discrete particle swarm optimization algorithm," *International Journal of Advanced Manufacturing Technology*, vol. 50, no. 5-8, pp. 761–770, Sep. 2010.

[152] J. A. Mukred *et al.*, "A Binary Particle Swarm Optimization Approach to Optimize assembly sequence planning," *Advanced Science Letters*, vol. 13, pp. 732–738, Jun. 2012.

[153] J. F. Wang, W. L. Kang, J. L. Zhao, and K. Y. Chu, "A simulation approach to the process planning problem using a modified particle swarm optimization," *Advances in Production Engineering And Management*, vol. 11, no. 2, pp. 77–92, 2016.

[154] M. F. F. A. Rashid, W. Hutabarat, and A. Tiwari, "Multi-objective discrete particle swarm optimisation algorithm for integrated assembly sequence planning and assembly line balancing," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol. 232, no. 8, pp. 1444–1459, Jun. 2018.

[155] C. Becker and A. Scholl, "A survey on problems and methods in generalized assembly line balancing," in *European Journal of Operational Research*, vol. 168, Feb. 2006, pp. 694–715.

[156] N. Boysen, A. Scholl, and M. Fliedner, "A classification of assembly line balancing problems," *European Journal of Operational Research*, vol. 183, no. 2, pp. 674–693, 2007.

[157] P. A. Pinto, D. G. Dannenbring, and B. M. Khumawala, "Assembly line balancing with processing alternatives: An application," *Management Science*, vol. 29, no. 7, pp. 817–830, 1983.

[158] A. Scholl, N. Boysen, and M. Fliedner, "The sequence-dependent assembly line balancing problem," *OR Spectrum*, vol. 30, no. 3, pp. 579–609, Jun. 2008.

[159] A. Scholl, N. Boysen, and M. Fliedner, "Optimally solving the alternative subgraphs assembly line balancing problem," *Annals of Operations Research*, vol. 172, no. 1, pp. 243–258, Jan. 2009.

[160] J. Bukchin and J. Rubinovitz, "A weighted approach for assembly line design with station paralleling and equipment selection," *IIE Transactions*, vol. 35, pp. 73–85, 2003.

[161] P. M. Vilarinho and A. S. Simaria, "ANTBAL: An ant colony optimization algorithm for balancing mixed-model assembly lines with parallel workstations," *International Journal of Production Research*, vol. 44, no. 2, pp. 291–303, Jan. 2006.

[162] S. Akpinar and G. Mirac Bayhan, "A hybrid genetic algorithm for mixed model assembly line balancing problem with parallel workstations and zoning constraints," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 3, pp. 449–457, Apr. 2011.

[163] S. Li, H. Wang, S. J. Hu, Y. T. Lin, and J. A. Abell, "Automatic generation of assembly system configuration with equipment selection for automotive battery manufacturing," *Journal of Manufacturing Systems*, vol. 30, no. 4, pp. 188–195, Oct. 2011.

[164] J. Bautista and J. Pereira, "Ant algorithms for a time and space constrained assembly line balancing problem," *European Journal of Operational Research*, vol. 177, no. 3, pp. 2016–2032, Mar. 2007.

[165] J. Bautista and J. Pereira, "Procedures for the Time and Space constrained Assembly Line Balancing Problem," *European Journal of Operational Research*, vol. 212, no. 3, pp. 473–481, Aug. 2011.

[166] M. Chica, Ó. Cordón, S. Damas, and J. Bautista, "Multiobjective constructive heuristics for the 1/3 variant of the time and space assembly line balancing problem: ACO and random greedy search," *Information Sciences*, vol. 180, no. 18, pp. 3465–3487, Sep. 2010.

[167] B. Zhou and Q. Wu, "An improved immune clonal selection algorithm for bi-objective robotic assemble line balancing problems considering time and space constraints," *Engineering Computations*, vol. 36, no. 6, pp. 1868–1892, Jul. 2019.

[168] Ö. Hazir, X. Delorme, and A. Dolgui, "A review of cost and profit oriented line design and balancing problems and solution approaches," *Annual Reviews in Control*, vol. 40, pp. 14–24, 2015.

[169] A. Kazemi and A. Sedighi, "A Cost-oriented Model for Multi-manned Assembly Line Balancing Problem," *Journal of Optimization in Industrial Engineering*, vol. 13, pp. 13–25, 2013.

[170] N. C. Wei and I. M. Chao, "A solution procedure for type e simple assembly line balancing problem," *Computers and Industrial Engineering*, vol. 61, no. 3, pp. 824–830, Oct. 2011.

[171] P. T. Zacharia and A. C. Nearchou, "A meta-heuristic algorithm for the fuzzy assembly line balancing type-E problem," *Computers and Operations Research*, vol. 40, no. 12, pp. 3033–3044, 2013.

[172] M. Jusop and M. F. F. A. Rashid, "Optimisation of assembly line balancing type-E with resource constraints using NSGA-II," in *Key Engineering Materials*, vol. 701, Trans Tech Publications Ltd, 2016, pp. 195–199, ISBN: 9783038356875.

[173] M. Jusop and M. F. A. Rashid, "A review on simple assembly line balancing type-e problem," in *IOP Conference Series: Materials Science and Engineering*, vol. 100, Institute of Physics Publishing, Dec. 2015.

[174] J. F. Bard, "Assembly line balancing with parallel workstations and dead time," *International Journal of Production Research*, vol. 27, no. 6, pp. 1005–1018, 1989.

[175] Y. Ege, M. Azizoglu, and N. E. Ozdemirel, "Assembly line balancing with station paralleling," *Computers and Industrial Engineering*, vol. 57, no. 4, pp. 1218–1225, Nov. 2009.

[176] D. Leiber, A. T. Vuong, and G. Reinhart, "Alternative subgraphs assembly line balancing problem with resource selection and parallel stations," *Engineering Optimization*, vol. 54, no. 11, 2021.

[177] P. R. McMullen and G. V. Frazier, "Using simulated annealing to solve a multiobjective assembly line balancing problem with parallel workstations," *International Journal of Production Research*, vol. 36, no. 10, pp. 2717–2741, 1998.

[178] O. Battaïa and A. Dolgui, "A taxonomy of line balancing problems and their solution approaches," *International Journal of Production Economics*, vol. 142, no. 2, pp. 259–277, Apr. 2013.

[179] E. Owensby, A. Shanthakumar, V. Rayate, E. Namouz, and J. Summers, "EVALUATION AND COMPARISON OF TWO DESIGN FOR ASSEMBLY METHODS: SUBJECTIVITY OF INFORMATION INPUTS," in *Proceedings of the ASME 2011 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, Washington, DC, USA, Aug. 2011.

[180] M. F. F. A. Rashid, N. M. Z. N. Mohamed, and A. N. M. Rose, "Multi-objective multi-verse optimiser for integrated two-sided assembly sequence planning and line balancing," *Journal of Combinatorial Optimization*, vol. 44, no. 1, pp. 850–876, Aug. 2022.

[181] A. Cox, "FIDELITY ASSESSMENT FOR MODEL SELECTION (FAMS): A FRAMEWORK FOR INITIAL COMPARISON OF MULTIFIDELITY MODELING OPTIONS," Ph.D. dissertation, Georgia Institute of Technology, 2019.

[182] S. Wang, J. Hong, Y. L. Li, Z. Yi, and B. Z. Li, "Assembly Planning of Aircraft Based on Polychromatic Sets," in *2010 IEEE International Conference on Industrial Engineering and Engineering Management*, Macao, China: IEEE, Dec. 2010, ISBN: 9781424485031.

[183] J. D. Anderson, *Aircraft Performance and Design*. The McGraw-Hill Companies, Inc., 1999.

[184] U. S. Navy, *Airman NAVEDTRA 14014 Nonresident Training Course*. Naval Education, Training Professional Development, and Technology Center, 2000.

[185] C. Kennedy, *Boeing 707 Owners' Workshop Manual: 1957 to Present*. Haynes Publishing UK, 2018.

[186] C. Collier, P. Yarrington, M. Pickenheim, and B. Bednarcyk, "An Approach to Preliminary Design and Analysis," in *48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Honolulu, Hawaii, Apr. 2007.

[187] J. A. Corman, N. Weston, C. Friedland, D. N. Mavris, and T. W. Laughlin, "A Parametric Multi-Fidelity Approach to Conceptual Airframe Design," in *2018 AIAA Modeling and Simulation Technologies Conference*, Kissimmee: American Institute of Aeronautics and Astronautics (AIAA), Jan. 2018.

[188] D. Sarojini, J. Xie, Y. Cai, J. A. Corman, and D. Mavris, "A Certification-Driven Platform for Multidisciplinary Design Space Exploration in Airframe Early Pre-

liminary Design," in *AIAA Aviation 2020 Forum*, American Institute of Aeronautics and Astronautics (AIAA), Jun. 2020.

[189] S. Rakshit and S. Akella, "The Influence of Motion Paths and Assembly Sequences on the Stability of Assemblies," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 2, pp. 615–627, Apr. 2015.

[190] D. Halperin, J. C. Latombe, and R. H. Wilson, "A general framework for assembly planning: The motion space approach," *Algorithmica (New York)*, vol. 26, no. 3-4, pp. 577–601, 2000.

[191] Z. Ding and B. Hon, "Constraints analysis and evaluation of manual assembly," *CIRP Annals - Manufacturing Technology*, vol. 62, no. 1, pp. 1–4, 2013.

[192] Q. Su and S. J. Lai, "3D geometric constraint analysis and its application on the spatial assembly sequence planning," *International Journal of Production Research*, vol. 48, no. 5, pp. 1395–1414, Mar. 2010.

[193] C. F. Li, Y. T. Feng, and D. R. Owen, "SMB: Collision detection based on temporal coherence," *Computer Methods in Applied Mechanics and Engineering*, vol. 195, no. 19-22, pp. 2252–2269, Apr. 2006.

[194] Christiand and J. Yoon, "Optimal assembly path planning algorithm for aircraft part maintenance," in *International Conference on Control, Automation and Systems 2007*, Oct. 2007, ISBN: 9788995003862.

[195] E. Masehian and S. Ghandi, "Assembly sequence and path planning for monotone and nonmonotone assemblies with rigid and flexible parts," *Robotics and Computer-Integrated Manufacturing*, vol. 72, Dec. 2021.

[196] R. M. V. Bahubalendruni, G. A. Kumar, and M. S. K. Sankar, "Practically Feasible Optimal Assembly Sequence Planning with Tool Accessibility," in *IOP Conference Series: Materials Science and Engineering*, vol. 390, Institute of Physics Publishing, Jul. 2018.

[197] M. C. Frank, R. A. Wysk, and S. B. Joshi, "Determining setup orientations from the visibility of slice geometry for rapid computer numerically controlled machining," *Journal of Manufacturing Science and Engineering, Transactions of the ASME*, vol. 128, no. 1, pp. 228–238, Feb. 2006.

[198] M. J. Hoefer and M. C. Frank, "Automated Manufacturing Process Selection During Conceptual Design," *Journal of Mechanical Design*, vol. 140, no. 3, p. 031 701, Jan. 2018.

[199] R. H. Wilson, "Geometric reasoning about assembly tools," *Artificial Intelligence*, vol. 98, pp. 237–279, Jan. 1998.

[200] P. K. Ghosh, "A Solution of Polygon Containment, Spatial Planning, and Other Related Problems Using Minkowski Operations," *COMPUTER VISION, GRAPHICS, AND IMAGE PROCESSING*, vol. 49, pp. 1–35, 1990.

[201] C. Chung and Q. Peng, "A novel approach to the geometric feasibility analysis for fast assembly tool reasoning," *International Journal of Advanced Manufacturing Technology*, vol. 31, no. 1-2, pp. 125–134, Nov. 2006.

[202] C. Hui, L. Yuan, and Z. Kai-Fu, "Efficient method of assembly sequence planning based on GAAA and optimizing by assembly path feedback for complex product," *International Journal of Advanced Manufacturing Technology*, vol. 42, no. 11-12, pp. 1187–1204, 2009.

[203] D. Dakdouk and F. Xi, "Tool Accessibility Analysis for Robotic Drilling and Fastening," *Journal of Manufacturing Science and Engineering, Transactions of the ASME*, vol. 139, no. 9, Sep. 2017.

[204] S. J. Fortune, "A fast algorithm for polygon containment by translation," in *ICALP 1985: Automata, Languages and Programming*, 1985, pp. 189–198.

[205] Y.-K. Chen, S.-Y. Chou, and T.-C. Wu, "An Efficient Method for Computing the Feasible Region with Translational Containment between two Convex Polygons," in *Proceedings 21st International Conference on Distributed Computing Systems Workshops*, 2001.

[206] R. H. Wilson, "A Framework for Geometric Reasoning About Tools in Assembly," in *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, Minneapolis, MN, Apr. 1996.

[207] S. Topaloglu, L. Salum, and A. A. Supciller, "Rule-based modeling and constraint programming based solution of the assembly line balancing problem," *Expert Systems with Applications*, vol. 39, no. 3, pp. 3484–3493, Feb. 2012.

[208] H. B. Maynard, G. J. Stegemertenv, and J. L. Schwab, *Methods Time Measurement*. McGraw-Hill Book Company, 1948.

[209] K. B. Zandin, *MOST Work Measurement Systems*. Marcel Dekker, 1989.

[210] M. Eigner, D. Roubanov, S. Sindermann, and J. Ernst, "Assembly time estimation based on product assembly information," in *INTERNATIONAL DESIGN CONFERENCE –DESIGN 2014*, 2014.

[211] H. Cho, S. Lee, and J. Park, "Time estimation method for manual assembly using modapts technique in the product design stage," *International Journal of Production Research*, vol. 52, no. 12, pp. 3595–3613, 2014.

[212] A. Layer, E. T. Brinke, F. Van Houten, H. Kals, and S. Haasis, "Recent and future trends in cost estimation," *International Journal of Computer Integrated Manufacturing*, vol. 15, no. 6, pp. 499–510, Nov. 2002.

[213] R. Curran, S. Raghunathan, and M. Price, "Review of aerospace engineering cost modelling: The genetic causal approach," *Progress in Aerospace Sciences*, vol. 40, no. 8, pp. 487–534, Nov. 2004.

[214] H. P. Bao and J. A. Samareh, "Affordable design: A methodology to implement process-based manufacturing cost models into the traditional performance-focused multidisciplinary design optimization," in *8th Symposium on Multidisciplinary Analysis and Optimization*, American Institute of Aeronautics and Astronautics Inc., 2000.

[215] Northrop Corporation, "Advanced composite cost estimating manual," Air Force Flight Dynamics Laboratory, 1976.

[216] M. Ashby *et al.*, *Engineering Materials and Processes Desk Reference*, First Edition. Butterworth-Heinemann, 2009, pp. 89–91.

[217] Y. Haimes, L. Lasdon, and D. Wismer, "On a bicriterion formulation of the problems of integrated system identification and system optimization," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-1, no. 3, pp. 296–297, 1971.

[218] J. Yu and C. Wang, "Method for discriminating geometric feasibility in assembly planning based on extended and turning interference matrix," *International Journal of Advanced Manufacturing Technology*, vol. 67, no. 5-8, pp. 1867–1882, Jul. 2013.

[219] M. Amen, "An exact method for cost-oriented assembly line balancing," *International Journal of Production Economics*, vol. 64, pp. 187–195, 2000.

[220] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, Apr. 2003.

[221] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, Ltd, 2001.

[222] E. Zitzler, "Evolutionary algorithms for multiobjective optimization: Methods and applications," Ph.D. dissertation, Swiss Federal Institute of Technology Zurich, 1999.

[223] N. Riquelme, C. V. Lücken, and B. Barán, "Performance metrics in multi-objective optimization," in *Proceedings - 2015 41st Latin American Computing Conference, CLEI 2015*, Dec. 2015, ISBN: 9781467391436.

[224] J. Knowles, "A summary-attainment-surface plotting method for visualizing the performance of stochastic multiobjective optimizers," in *Proceedings - 5th International Conference on Intelligent Systems Design and Applications 2005, ISDA '05*, 2005, pp. 552–557, ISBN: 0769522866.

[225] US Air Force, "F-86f -25 thru -40 sabre standard aircraft characteristics," North American, 1956, https://www.avialogs.com/aircraft-n/north-american-aviation/item/4689-f-86f-25-thru-40-sabre-standard-aircraft-characteristics.

[226] B. Boehm, C. Abts, and S. Chulani, "Software Development Cost Estimation Approaches -A Survey," *Annals of Software Engineering*, vol. 10, pp. 177–205, 2000.

[227] C. Hueber, K. Horejsi, and R. Schledjewski, "Review of cost estimation: methods and models for aerospace composite manufacturing," *Advanced Manufacturing: Polymer and Composites Science*, vol. 2, no. 1, pp. 1–13, Jan. 2016.

[228] *SEER for Manufacturing (SEER-MFG)*, 2020.

[229] R. Curran, M. Mullen, N. Brolly, M. Gilmour, P. Hawthorne, and S. Cowan, "Cost Modelling of Composite Aerospace Parts and Assemblies," in *Collaborative Product and Service Life Cycle Management for a Sustainable World*, Springer London, Aug. 2008, pp. 281–294.

[230] M. Kaufmann, D. Zenkert, and P. Wennhage, "Integrated cost/weight optimization of aircraft structures," *Structural and Multidisciplinary Optimization*, vol. 41, no. 2, pp. 325–334, Mar. 2010.

[231] Electroimpact, *Wingbox Assembly Lines*, https://web.archive.org/web/20230608224209/https://electroimpact.com/wingbox.aspx, Accessed: 2022-01-20, 2022.

[232] K. A. Linton, A. Velicki, K. Hoffman, P. Thrash, R. Pickell, and R. Turley, "Prseus panel fabrication final report nasa/cr–2014-218149," NASA, 2014.

[233] A. Velicki and D. Jegley, "Prseus structural concept development," in *AIAA 2014-0259. 52nd Aerospace Sciences Meeting*, 2014.

[234] D. Gates, "Boeing shows off new 777x wing center," *The Seattle Times*, May 2016, https://web.archive.org/web/20230609164746/https://www.seattletimes.com/business/boeing-aerospace/boeing-shows-off-new-777x-wing-center/, Accessed 2022-02-08.

[235] Shimizu Corporation, *Mitsubishi Heavy Industries,Ltd., Nagoya Aerospace Systems Works Composite Wing Center Fabrication Factory*, https://web.archive.org/web/20230609165741/https://www.shimz.co.jp/en/works/jp_fac_200512_mitsubishi_heavyIndustries_nagoya.html, Accessed: 2022-02-08.

[236] U.S. Centennial of Flight Commission, *North American Aviation*, https://web.archive.org/web/20230609163309/https://www.centennialofflight.net/essay/Aerospace/NorthAmerican/Aero37.htm, Accessed: 2023-01-12.

[237] Leagle, Inc., *NORTH AMERICAN AVIATION, INC. v. RENEGOTIATION BOARD*, https://web.archive.org/web/20230609163015/https://www.leagle.com/decision/196224639stc2071228, Accessed: 2022-02-08.

[238] The Boeing Company, *Boeing Backgrounder: Boeing South Carolina*, https://web.archive.org/web/20130927091407/http://www.boeing.com/assets/pdf/commercial/charleston/pdf/bkg_BoeingSC.pdf, Accessed: 2022-02-08, Aug. 2013.

[239] R. Ewing, "7 photos from inside boeing's south carolina facility," *Airline Geeks*, Mar. 2018, https://web.archive.org/web/20230609170936/https://airlinegeeks.com/2018/03/25/7-photos-from-inside-boeings-south-carolina-facility/, Accessed 2022-02-08.

[240] J. W. Sawyer, "Effect of stitching on the strength of bonded composite single lap joints," *AIAA Journal*, vol. 23, no. 11, pp. 1744–1748, 1985.

[241] S. Tang, Z. Cai, and J. Zheng, "A fast method of constructing the non-dominated set: Arena's principle," in *Proceedings - 4th International Conference on Natural Computation, ICNC 2008*, vol. 1, 2008, pp. 391–395, ISBN: 9780769533049.

[242] S. Mishra, S. Mondal, and S. Saha, "Fast implementation of steady-state NSGA-II," in *2016 IEEE Conference on Evolutionary Computation*, Vancouver, BC, Canada: IEEE, Nov. 2016, pp. 3777–3784, ISBN: 9781509006236.

[243] R. L. Becerra and C. A. C. Coello, "Solving hard multiobjective optimization problems using epsilon-constraint with cultured differential evolution," in *PPSN 2006: Parallel Problem Solving from Nature - PPSN IX*, 2006, pp. 543–552.

[244] R. Landa, G. Lárraga, and G. Toscano, "Use of a goal-constraint-based approach for finding the region of interest in multi-objective problems," *Journal of Heuristics*, vol. 25, no. 1, pp. 107–139, Feb. 2019.

[245] Z. Fan *et al.*, "An improved epsilon constraint handling method embedded in moea/d for constrained multi-objective optimization problems," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2016, pp. 1–8, ISBN: 9781509042401.

[246] C. A. C. Coello, "Multi-objective evolutionary algorithms in real-world applications: Some recent results and current challenges," in *Computational Methods in Applied Sciences*, vol. 36, Springer Science and Business Media B.V., 2015, pp. 3–18, ISBN: 9783319115405.

[247] A. Aguilar-Rivera, "A GPU fully vectorized approach to accelerate performance of NSGA-2 based on stochastic non-domination sorting and grid-crowding," *Applied Soft Computing Journal*, vol. 88, Mar. 2020.

[248] D. N. Mavris, O. P. Fischer, D. Fullmer, and O. J. Pinon, "Systems design and modeling: A visual analytics approach," in *ICAS 2010 - 27th International Conference of the Aeronautical Sciences*, reference for surrogates, along with siedlak myers source, 2010.

[249] S. Picek, D. Jakobovic, and M. Golub, "On the recombination operator in the real-coded genetic algorithms," in *2013 IEEE Congress on Evolutionary Computation*, Explains how to perform whole arithmetic crossover in GAs and shows that that particular crossover scheme is actually relatively effective. Not the most effective but extremely simple and efficient for how good it is., 2013.

[250] K. Deb and D. Deb, "Analysing mutation schemes for real-parameter genetic algorithms," *International Journal of Artificial Intelligence and Soft Computing*, vol. 4, no. 1, pp. 1–28, 2014, Goes over gaussian mutation scheme for real value genetic algorithms, goes over generally-good values for parameters such as for mutation strength.

[251] C. Coello, G. Lamont, and D. Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, 2nd. Springer, 2007, ISBN: 9780387310299.

[252] J. Knowles, "Parego: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 1, pp. 50–66, Feb. 2006.

[253] U.S. Air Force, *USAF Series F-86F Aircraft Flight Manual T.O. 1F-86F-1*. Secretary of the Air Force, 1960.

[254] R. Wagner, *North American Sabre Jet F-86D/K/L Part One Design/Structure/Testing* (Air Force Legends). Ginter Books, 1999.

[255] U.S. Air Force, *USAF Series F-86K Aircraft Handbook: Structural Repair Instructions T.O. 1F-86K-3*. Secretary of the Air Force, 1957.

[256] U.S. Air Force, *USAF Series F-86F Aircraft Handbook: Maintenance Instructions T.O. 1F-86F-2*. Secretary of the Air Force, 1954.

[257] M. Murata, "Manufacturing of f-86 jet fighter," *Journal of the Society of Mechanical Engineers*, vol. 60, no. 457, pp. 188–194, 1957, https://doi.org/10.1299/jsmemag.60.457_188.

[258] J. Markish, "Valuation Techniques for Commercial Aircraft Program Design," Ph.D. dissertation, Massachusetts Institute of Technology, 2000.

[259] M. C. Niu, *Airframe Structural Design*. Conmilit Press, Ltd, 1988.

[260] R. M. Ajaj, D. Smith, and A. T. Isikveren, "A conceptual wing-box weight estimation model for transport aircraft," *Aeronautical Journal*, vol. 177, no. 1191, pp. 533–551, 2013.

[261] Par Systems, Inc., *Laser Ultrasonic NDT System - LaserUT*, https://web.archive.org/web/20230607020254/https://www.dndkm.org/Technology/TechnologyFactSheet.aspx?TechnologyID=1876, Accessed: 2022-01-29.

[262] Norsk Hydro ASA, *What is Aluminium Anodizing and How Does It Work — Anodizing Process Overview*, https://web.archive.org/web/20230607033524/https://www.youtube.com/watch?v=0yl35W0o9S0, Accessed: 2022-01-29, 2021.

[263] A. Velicki, "Damage arresting composites for shaped vehicles phase 1 final report," NASA, 2009, https://web.archive.org/web/20230630055754/https://ntrs.nasa.gov/api/citations/20090034167/downloads/20090034167.pdf.

[264] M. Karal, "Ast composite wing program-executive summary," NASA, 2001, https://web.archive.org/web/20230630055627/https://ntrs.nasa.gov/api/citations/20010033249/downloads/20010033249.pdf.

[265] F. Aymerich, "Effect of stitching on the static and fatigue performance of co-cured composite single-lap joints," *Journal of Composite Materials*, vol. 38, no. 3, pp. 243–257, 2004.

[266] P. W. Boyd *et al.*, *High Rate Pulsing Wing Assembly Line*, US Patent 8,661,684 B1, 2014.

[267] *Genuine Aircraft Hardware Company*, https://www.gen-aircraft-hardware.com/, Accessed: 2022-01-31.

[268] M. Dow, "The advanced stitching machine: Making composite wing structures of the future," NASA Facts, Langley Research Center, Aug. 1997, https://web.archive.org/web/20230607185836/https://www.nasa.gov/centers/langley/news/factsheets/ASM.html, Accessed: 2022-01-28.

[269] J. Ventura, "Lecture Notes on Space Requirements," *Pennsylvania State University*, Feb. 2019.

[270] A. Heckwolf, "Integration of demand variability into aircraft factory cost predictions using manufacturing influenced design (MInD)," Georgia Institute of Technology, Tech. Rep., 2014.

[271] A. Goel, "Economics of composite material manufacturing equipment," Ph.D. dissertation,
Massachusetts Institute of Technology, Sep. 2000.

[272] M. K. Hagnell and M. Åkermo, "A composite cost model for the aeronautical industry: Methodology and case study," *Composites Part B: Engineering*, vol. 79, pp. 254–261, May 2015.

[273] D. Gates, "At boeing's 777x wing factory, robots get big jobs," *The Seattle Times*, Nov. 2016, https://web.archive.org/web/20211204091209/https://www.seattletimes.com/business/boeing-aerospace/at-boeings-777x-wing-factory-robots-get-big-jobs/, Accessed 2022-01-24.

[274] ASC Process Systems, *What's an Econoclave*, https://web.archive.org/web/20230605062905/http://www.aschome.com/index.php/en/products/autoclaves/whats-an-econoclave, Accessed: 2022-01-29.

[275] Ebay, *Used ITS Oven Schematic*, https://web.archive.org/web/20230605060640/https://i.ebayimg.com/images/g/klAAAOSwe~BbrkwJ/s-l1600.jpg, Accessed: 2022-01-29.

[276] G. G. Wang and S. Shan, "Review of metamodeling techniques in support of engineering design optimization," *Journal of Mechanical Design*, vol. 129, no. 4, pp. 370–380, Apr. 2007.

[277] R. H. Myers, D. C. Montgomery, and C. M. Anderson-Cook, *Response Surface Methodology*. John Wiley & Sons Inc., 2009.

[278] Woodward Co., *Magnaflux Black Lights*, https://www.ndtsupplies.com/blacklights1.html, Accessed: 2022-01-25.

[279] GH Cranes, *TORRESFIBERLAYUP: fabricación de componentes para aeronaves*, https://web.archive.org/web/20230606154204/https://www.ghcranes.com/blog/torresfiberlayup-fabricacion-de-componentes-para-aeronaves/, Accessed: 2022-01-27.

[280] Modern Machine Shop, *A More Affordable Approach to 5-Axis Large Part Machining*, https://web.archive.org/web/20210508185957/https://www.mmsonline.com/

articles/a-more-affordable-approach-to-5-axis-large-part-machining, Accessed: 2022-01-24, 2020.

[281] C.R. Onsrud, Inc., *5-Axis HR Series - High Rail CNC Machining Center*, https://web.archive.org/web/20230606185213/https://www.cronsrud.com/brochures/HR_Series_5-Axis_CROnsrud.pdf, Accessed: 2022-01-24.

[282] C.R. Onsrud, Inc., *5-Axis G Series CNC Router*, https://web.archive.org/web/20230606201015/https://cronsrud.com/brochures/G-Series_5-Axis_SellSheet_CROnsrud.pdf, Accessed: 2022-01-25.

[283] G. F. Schrader and A. K. Elshennawy, *Manufacturing Processes and Materials - Fourth Edition*. Society of Manufacturing Engineers, 2000.

[284] ACB, An Aries Industries Company, *Transverse/Longitudinal Sheet Stretch Forming Process (VTL)*, https://web.archive.org/web/20230606210529/https://www.acb-ps.com/en/transverse-longitudinal-sheet-stretch-forming-press-vtl, Accessed: 2022-01-25.

[285] New Tech Machinery, *WAV Wall Panel Machine*, https://web.archive.org/web/20230606222236/https://newtechmachinery.com/machines/roof-wall-panel-machines/wav-wall-panel-machine/, Accessed: 2022-01-24.

[286] Focus Technology Co., *Made-in-China Sales Listing: Wall Panel Roll Forming Machine and Metal Forming Machine*, https://web.archive.org/web/20230606222759/https://willingint.en.made-in-china.com/product/zXkxPUGrvCVf/China-Ce-Trapezoid-Wall-Roll-Forming-Machine-Roof-Rollformer.html, Accessed: 2022-01-24.

[287] John, "How Much Do CNC Machines Cost? [2023]-Every Type," *MellowPine*, 2022, https://web.archive.org/web/20230607002349/https://mellowpine.com/cnc/how-much-do-cnc-machines-cost/, Accessed: 2022-01-24.

[288] Omax Corporation, *OMAX 160X - PRECISION JETMACHINING CENTER*, https://web.archive.org/web/20230607004402/https://www.omax.com/sites/default/files/2021-08/OMAX%20160x-series.pdf, Accessed: 2022-01-25.

[289] Zhuozhou Yesheng Import & Export Co., Ltd., *Alibaba Listing: Automatic Aluminium Profile Anodizing Processing Line*, https://web.archive.org/web/20230607031901/https://www.alibaba.com/product-detail/Automatic-Aluminium-Profile-Anodizing-Processing-Line_1600624258158.html?spm=a2700.7724857.0.0.7958175etgwf17, Accessed: 2023-06-06.

[290] Baker Technology Associates Inc., *Used Plating Equipment Listing: 35' Long by 3' Wide by 6' Deep Anodizing System Item 331*, https://web.archive.org/web/20230607000729/http://web.archive.org/screenshot/https://www.flickr.com/photos/198501993@N06/52955826082/in/dateposted-public/lightbox/, Accessed: 2022-01-26.

[291] Home Depot, *IDEAL Paint Booths*, https://web.archive.org/web/20230607050331/https://www.homedepot.com/b/Paint-Paint-Supplies-Paint-Sprayers-Paint-Booths/IDEAL/N-5yc1vZciktZrj, Accessed: 2022-01-27.

[292] JMC Automotive Equipment, *RGI DELTA CARGO Truck Spray Booth*, https://web.archive.org/web/20230607045254/https://jmcautomotiveequipment.com/automotive-equipment/paint-booth/industrial/rgi-delta-cargo-truck-spray-booth/, Accessed: 2022-01-27.

[293] Standard Tools and Equipment Co., *Truck Paint Booths*, https://web.archive.org/web/20230607050126/https://www.paint-booths.com/category/truck-paint-booths.html, Accessed: 2022-01-27.

[294] J. Verrey, M. D. Wakeman, V. Michaud, and J. A. Månson, "Manufacturing cost comparison of thermoplastic and thermoset rtm for an automotive floor pan," *Composites Part A: Applied Science and Manufacturing*, vol. 37, no. 1, pp. 9–22, Jan. 2006.

[295] LT Machines, *Hot Drape Former HDF 9 (30')*, https://web.archive.org/web/20230607175122/https://www.ltmachines.com/products/hot-drape-formers/hdf-9-30, Accessed: 2022-01-26.