Multi-stage Stochastic Programming Models in Production Planning

A Thesis
Presented to
The Academic Faculty

by

Kai Huang

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

School of Industrial and Systems Engineering
Georgia Institute of Technology
August 2005

Multi-stage Stochastic Programming Models in Production Planning

Date Approved: July 11, 2005

To my dear wife Ying Luo, my dad Naiyu Huang, my mom Yan Li, and my whole family, your love is always the source of my strength.

ACKNOWLEDGEMENTS

First, I want to thank my advisor Dr. Shabbir Ahmed, without whose guidance I could not have finished this exciting journey of discovery and exploration. He taught me how to select a valuable research topic, to conduct high quality research, and to properly present that research. Also my friend, he is always willing to listen to me and to offer advice in every aspect of life. Second, I want to thank Dr. Marc Goetschalckx and Dr. Leon McGinnis, who taught me how to think like an industrial engineer. Third, I thank all my committee members whose patient and considerate advice provided significant insights for this research. Fourth, I thank all the professors in Georgia Institute of Technology who taught me in the past five years. Their high quality courses will always be my treasure. Finally, I thank all my colleagues. I have had great pleasure in working with them, learning from them, and living with them.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

In this thesis, we study a series of closely related multi-stage stochastic programming models in production planning, from both a modelling and an algorithmic point of view. We first consider a very simple multi-stage stochastic lot-sizing problem, involving a single item with no fixed charge and capacity constraint. Although a multi-stage stochastic integer program, this problem can be shown to have a totally unimodular constraint matrix. We develop primal and dual algorithms by exploiting the problem structure. Both algorithms are strongly polynomial, and therefore much more efficient than the Simplex method. Next, motivated by applications in semiconductor tool planning, we develop a general capacity planning problem under uncertainty. Using a scenario tree to model the evolution of the uncertainties, we present a multi-stage stochastic integer programming formulation for the problem. In contrast to earlier two-stage approaches, the multi-stage model allows for revision of the capacity expansion plan as more information regarding the uncertainties is revealed. We provide analytical bounds for the value of multi-stage stochastic programming over the two-stage approach. By exploiting the special simple stochastic lot-sizing substructure inherent in the problem, we design an efficient approximation scheme and show that the proposed scheme is asymptotically optimal. We conduct a computational study with respect to a semiconductor-tool-planning problem. Numerical results indicate that even an approximate solution to the multi-stage model is far superior to any optimal solution to the two-stage model. These results show that the value of multi-stage stochastic programming for this class of problem is extremely high. Next, we extend the simple stochastic lot-sizing model to an infinite horizon problem to study the planning horizon of this problem. We show that an optimal solution of the infinite horizon problem can be approximated by optimal solutions of a series of finite horizon problems, which implies the existence of a planning horizon. We also provide a useful upper bound for the planning horizon.

CHAPTER I

INTRODUCTION

1.1 Background and motivation

Production planning is a key area of operations management. An important methodology for production planning is mathematical programming. Traditional mathematical programming models for production planning are deterministic, and cannot provide robust production plans in the presence of uncertainty. As such, deterministic planning models may yield unsatisfactory decisions. To motivate our research, consider the following capacity and production planning problem arising in the semiconductor industry. In a semiconductor wafer fab, one must determine the number of machine tools required for manufacturing the appropriate product mix. However, the tools are highly customized and expensive, and the lead time for tool manufacturing can be up to 18 months. Moreover, the demand for the products is highly volatile. In this application, there is a clear need for explicitly addressing uncertainty, and optimization models dealing with uncertainty are greatly needed.

Stochastic programming [14, 31, 49], an active branch of mathematical programming dealing with optimization problems involving uncertain data, has seen several successful applications in production planning [10, 22, 23, 55]. Unlike alternative approaches to decision making under uncertainty, such as Markov decision processes, stochastic programming requires few assumptions on the underlying stochastic processes and allows for modelling of complicated decision structures. On the other hand, stochastic programming assumes a finite number of stages and exogenous uncertainties. With recent increase in computational power and algorithmic developments, the limitations of stochastic programming arising from computational difficulties have been relieved to a large extent. In recent years, a variety of algorithms for various classes of stochastic programs have been developed and used for solving realistic problems [49]. In multi-period production planning, most existing stochastic programming models adopt a two-stage approach. For the semiconductor tool planning example discussed above, a number of such two-stage stochastic programming models have been proposed [5, 22, 28, 55, 45]. In a two-stage approach, the plan for the entire multi-period planning horizon is determined before the uncertainty is realized, and only a limited number of recourse actions can be taken afterwards. In contrast, a multi-stage approach allows revision of the planning decisions as more information regarding the uncertainties is revealed. Consequently, the multi-stage model is a better characterization of the dynamic planning process, and provides more flexibility than does the two-stage model. However, very few multi-stage stochastic programming models have been developed and actually solved in applications [2, 3, 32, 35, 44, 57]. Two reasons can be identified. First, multi-stage models are much more difficult to solve than two-stage models. This is particularly true when integer variables are involved [47, 58]. Second, even though multi-stage models allow for superior solutions, the actual improvement in quality may be small and may not justify the added computational difficulties. Unfortunately, no quantitative analysis of solution quality of multi-stage models has been reported in the literature.

The goal of this thesis is to study multi-stage stochastic programming models in production planning applications with a view to exploit special structures and develop efficient solution algorithms. A related goal is to understand the value of multi-stage models over two-stage models in certain applications. Finally, we investigate the approximation of a certain class of infinite-horizon stochastic production planning problems using finite-horizon problems.

1.2 Outline of the thesis

The lot-sizing problem is a basic component of many production planning models. As in the deterministic case, to study production planning problems under uncertainty, we first study a simple stochastic lot-sizing problem. In Chapter II, we study a stochastic lot-sizing problem involving a single item, linear cost structures, and finite distributions for the stochastic cost and demand parameters. We represent the stochastic data as a *scenario tree.* In a scenario tree, each node n represents a possible state of the world, associated with a set of data (stochastic demand, stochastic cost, probability, etc.). The root node of the tree represents the current state of the world. Node $a(n)$ is the direct ancestor of node n. The direct descendants of node n are called the children of node n. The subtree with root node n is denoted by $\mathcal{T}(n)$. Since the root node is usually indexed as node 1, the whole scenario tree can be denoted as $\mathcal{T}(1)$. A path from the root node to a node n describes one realization of the stochastic process from the present to the period where node n appears. The set of all the nodes on this path is denoted as $\mathcal{P}(n)$. A full evolvement of the stochastic process over the entire planning horizon, i.e., the path from the root node to a leaf node, is called a scenario. Note the cardinality of $\mathcal{P}(n)$ represents the time period when node n appears, which is denoted by $t(n)$; and the set of all the nodes in a period t is represented by $\mathcal{S}_t$. An illustration of the scenario tree model is provided in Figure 1:

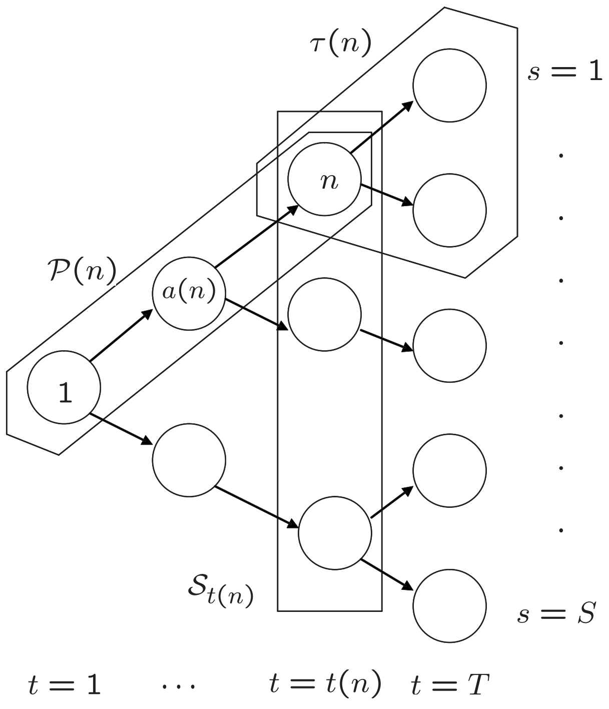

Figure 1: The scenario tree model

The scenario tree model provides an explicit description of the stochastic data. It has few assumptions on the underlying stochastic process. However, it does simplify the stochastic process by assuming a finite number of possible states in each period. On the other hand, if each non-leaf node has a fixed number of children, or a fixed number of branches, then the scale of scenario tree increases exponentially with increasing time horizon, which is a disadvantage. With the scenario tree model, we formulate a multi-stage stochastic linear program of the simple stochastic lot-sizing problem. We develop primal and dual algorithms for the problem. We prove that the complexity of the primal algorithm is $\mathcal{O}(N^2)$, while the complexity of the dual algorithm is $\mathcal{O}(N \log N \log(\log N))$, where N is the number of nodes in the scenario tree used to model the stochastic parameters. Numerical experiences further show the computational superiority of the two algorithms.

In Chapter III, we study a general class of capacity planning problems under uncertainty, where the simple stochastic lot-sizing problem is an inherent substructure. The model proposed is a multi-stage stochastic integer program. To evaluate the advantage of this model over two-stage models, the concept of *value of multi-stage stochastic programming* over the two-stage approach is introduced in [1]. For the general model, we provide useful analytical bounds for the value of multi-stage stochastic programming. Furthermore, by exploiting the simple stochastic lot-sizing substructure, we design an efficient approximation scheme, which is proven to produce asymptotically optimal solutions.

Chapter IV describes a computational study of a semiconductor tool planning problem with industrial-size data. We test the value of multi-stage stochastic programming and heuristic solution quality, with respect to an increasing number of stages and an increasing number of branches in the scenario tree. The results show that the value of multi-stage stochastic programming increases with increasing variability in stochastic data, while the heuristic solution quality increases with an increasing number of stages. An exciting observation is that, in cases where the value of multi-stage stochastic programming is higher, the performance of the approximation scheme is better.

We study an extension of the simple stochastic lot-sizing problem to an infinite horizon in Chapter V. In the context of Markov decision processes, much research has focused on using a finite-horizon problem to approximate the infinite-horizon problem. However, in the field of stochastic programming, to the best of our knowledge, there is no such study. For the infinite-horizon stochastic lot-sizing problem, we analyze finite-horizon approximation problems. We show that one can find a length of planning horizon such that an optimal first-stage solution to the finite-horizon approximation is the same as that of the infinite-horizon problem. This length is called the *planning horizon.* We develop a useful formula for calculating the planning horizon.

We summarize the major contributions of the thesis in Chapter VI, along with a discussion of possible future research directions.

CHAPTER II

A SIMPLE STOCHASTIC LOT-SIZING PROBLEM

2.1 Introduction

This chapter considers a stochastic extension of the finite horizon, single item, uncapacitated, dynamic lot-sizing problem with linear costs:

$$\begin{aligned} \min \quad & \sum_{n=1}^{T}(\alpha_n x_n + \beta_n I_n) \\ \text{s.t.} \quad & I_{n-1} + x_n = I_n + \delta_n \quad && n = 1, \ldots, T \\ & x_n, I_n \in \mathbb{Z}_+ && n = 1, \ldots, T, \end{aligned} \tag{1}$$

where T is number of planning periods, x_n and I_n denote the production and ending inventory decisions for period n, respectively, and α_n, β_n and δ_n denote the per-unit production cost, holding cost and demand for the period n. Note I_0 is the initial inventory at the beginning of the time horizon. We usually assume $I_0 = 0$. It is well known that (1) can be easily solved using simple greedy schemes (cf. [30, 60]).

Beginning with the seminal work of Arrow et al. [4], stochastic inventory problems have been studied extensively (cf. [60]). Much of this work is based on specific assumptions on the underlying stochastic processes, to allow for elegant analytical solutions. Here we consider general, albeit finite, distributions for the stochastic parameters. In this situation, by using a scenario tree to model the evolution of the stochastic parameters, a stochastic extension of (1) can be formulated as a multi-stage stochastic program [14]. Such a stochastic programming formulation has been considered in [21], where the author shows that the problem can be transformed to a network flow problem by introducing additional variables. In this chapter, we consider a slightly different version of the stochastic inventory problem considered in [21], and develop a primal and a dual algorithm for the problem. We show that the complexity of the proposed algorithms is $\mathcal{O}(N^2)$, where N is the number of nodes in the scenario tree used to model the stochastic parameters.

2.2 Stochastic programming formulation

To extend (1) to a stochastic setting, we use a scenario tree with T stages to describe the evolution of the uncertain data over the planning horizon. In this tree, the nodes in stage (or level) t of the tree constitute the states of the world that can be distinguished by information available up to time stage t. The probability associated with the state of the world represented by node n is p_n, and the time period corresponding to node n is t_n. Each node n has a unique direct ancestor $a(n)$ except the root node. Each non-leaf node n is the root of a non-trivial subtree denoted by $\mathcal{T}(n)$. We use $\mathcal{T} = \{1, ..., N\}$ to denote the whole tree, where N is the total number of nodes in the tree. For node n, we define $\mathcal{P}(n)$ as the set of all nodes on the path from the root of $\mathcal{T}$ to node n (including node n and the root), and $\bar{\mathcal{P}}(n) = \mathcal{P}(n) \backslash \{n\}$. The stochastic problem parameters are then given by the sequence $\{\alpha_n, \beta_n, \delta_n\}_{n \in \mathcal{T}}$. With an objective of minimizing the *expected* total costs, a multi-stage stochastic programming extension of (1) is as follows:

$$\begin{aligned} \min \quad & \sum_{n \in \mathcal{T}} p_n(\alpha_n x_n + \beta_n I_n) & \\ \text{s.t.} \quad & I_{a(n)} + x_n = I_n + \delta_n & \forall\, n \in \mathcal{T} \\ & x_n, I_n \in \mathbb{Z}_+ & \forall\, n \in \mathcal{T}. \end{aligned} \tag{2}$$

We can reformulate (2) by introducing cumulative demand $d_n = \sum_{m \in \mathcal{P}(n)} \delta_m$ and eliminating the variables I_n using the identity $I_n = \sum_{m \in \mathcal{P}(n)} x_m - d_n$ (we assume the initial inventory $I_0 = 0$). The resulting formulation is:

$$\begin{aligned} \min \quad & \sum_{n \in \mathcal{T}} c_n x_n - \bar{c} & \\ \text{s.t.} \quad & \sum_{m \in \mathcal{P}(n)} x_m \geq d_n & \forall\, n \in \mathcal{T} \\ & x_n \in \mathbb{Z}_+ & \forall\, n \in \mathcal{T}, \end{aligned} \tag{3}$$

where $c_n = p_n(\alpha_n + \frac{\sum_{m \in \mathcal{T}(n)} p_m \beta_m}{p_n})$ and $\bar{c} = \sum_{n \in \mathcal{T}} p_n \beta_n d_n$. Note that c_n is actually the expected total cost of producing one unit at node n and carry it as inventory until the end of time horizon. For each n, note that $c_n = p_n(\alpha_n + \beta_n) + \sum_{m \in \mathcal{T}: a(m) = n}(c_m - p_m \alpha_m)$. So the computation of c_n is $\mathcal{O}(BN)$ where B is the maximum number of branches of non-leaf nodes. Since $B < N$ (in fact $B << N$), the complexity of the reformulation step is within $\mathcal{O}(N^2)$. The remainder of the chapter will be concerned with formulation (3). In the following developments we drop the constant term $\bar{c}$ from the objective function, and assume $c_n > 0$ and $d_n > 0$ for all $n \in \mathcal{T}$:

$$\begin{aligned} \min \quad & \sum_{n \in \mathcal{T}} c_n x_n \\ \text{s.t.} \quad & \sum_{m \in \mathcal{P}(n)} x_m \geq d_n \quad && \forall\, n \in \mathcal{T} \\ & x_n \in \mathbb{Z}_+ && \forall\, n \in \mathcal{T}. \end{aligned} \tag{4}$$

The first observation is that even though (4) is a multi-stage stochastic integer program, the following advantageous property holds.

THEOREM 1 *With integer demand parameters $\{d_n\}_{n \in \mathcal{T}}$, the LP relaxation of (4) yields integral solutions.*

PROOF. Note that the constraint matrix of (4) is a $N \times N$ 0-1 matrix. Let us denote this matrix as $U = [u_{ij}]$, where $u_{ij} = 1$ if $j \in \mathcal{P}(i)$ in the scenario tree $\mathcal{T}$, and $u_{ij} = 0$ otherwise. Let $\mathcal{J}$ be any subset of the columns of U, i.e., a subset of the nodes in $\mathcal{T}$. Let $\{t_1, \ldots, t_K\}$ be the indices of time stages corresponding to the nodes in $\mathcal{J}$, and suppose that $t_1 < t_2 < \cdots < t_K$. Let $\overline{\mathcal{S}}_t = \mathcal{S}_t \cap \mathcal{J}$, i.e., the set of nodes in time stage t included in $\mathcal{J}$. We can then create a partitioning of the nodes (columns) in $\mathcal{J}$ as follows $\mathcal{J}_1 = \cup_{i=1,\ i \text{ is odd}}^{K} \overline{\mathcal{S}}_{t_i}$ and $\mathcal{J}_2 = \cup_{i=1,\ i \text{ is even}}^{K} \overline{\mathcal{S}}_{t_i}$. It is immediately verified that

$$\Big| \sum_{j \in \mathcal{J}_1} u_{ij} - \sum_{j \in \mathcal{J}_2} u_{ij} \Big| \leq 1 \ \forall\ i = 1, \ldots, N_T.$$

Thus for any subset of the columns in U, we can create a bi-partition such that the difference in the sum of coefficients of each partition along every row of U is at most 1. It then follows (cf. [40]) that U is totally unimodular, and the claim holds. $\square$

According to Theorem 1, we can rewrite (4) as follows when the d_n's are integers:

$$\begin{aligned} \min \quad & \sum_{n \in \mathcal{T}} c_n x_n \\ \text{s.t.} \quad & \sum_{m \in \mathcal{P}(n)} x_m \geq d_n \quad && \forall\, n \in \mathcal{T} \\ & x_n \in \mathbb{R}_+ && \forall\, n \in \mathcal{T}. \end{aligned} \tag{5}$$

Although (4) is quite a simple problem, it often arises as the key substructure in more complicated planning problems, such as the capacity planning problem considered in Chapter III. This structure also arises in stochastic extensions of some classes of joint pricing-inventory problems (cf. [16]), whose deterministic versions involve the classical lot-sizing structure (1). The algorithms proposed next can be very effective within decomposition based methods for the problems mentioned above.

2.3 Algorithms

In this section, we propose a primal and a dual algorithm for the stochastic lot-sizing problem (5). Our exposition relies on two different indexing systems for the nodes in the scenario tree $\mathcal{T}$.

Indexing scheme 1. The nodes in $\mathcal{T}$ are indexed $1, 2, \ldots, N$ in increasing order of their time stage, i.e., $t_1 \leq t_2 \leq \cdots \leq t_N$. No particular ordering is imposed on the indices of the nodes in the same time stage. Thus the root node has an index of 1.

Indexing scheme 2. The nodes in $\mathcal{T}$ are indexed $1, 2, \ldots, N$ in decreasing order of the corresponding cumulative demand, i.e., $d_1 \geq d_2 \geq \cdots \geq d_N$. If $d_m = d_n$, then $m < n$ if $t_m < t_n$.

The two indexing schemes corresponding to an example scenario tree are illustrated in Figure 2.

2.3.1 The primal algorithm

We assume that the nodes are labelled according to indexing scheme 1. Note that we can construct a solution $x^0 = (x_1^0, x_2^0, ..., x_N^0)$ for (5) by setting

$$x_n^0 = \max\{0, d_n - \max_{m \in \bar{\mathcal{P}}(n)} d_m\}$$

for all $n = 1, \ldots, N$. It is easily verified that

$$\sum_{m \in \mathcal{P}(n)} x_m^0 = \max_{m \in \mathcal{P}(n)} d_m \geq d_n,$$

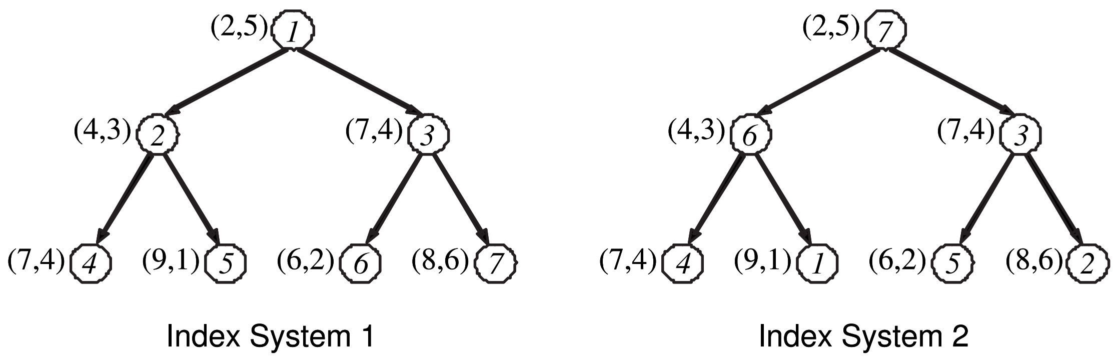

Figure 2: Indexing schemes (the numbers in parenthesis indicate (d_n, c_n))

and therefore x^0 is a feasible solution to (5). The key idea of our primal algorithm is to start from node production levels given by the solution x^0, and then to shift some production from a group of nodes to their common ancestor, whenever the sum of the unit costs of this node group is larger than that of the common ancestor. This operation will be called "shifting-up." In each shifting-up operation, we will shift as much as possible to make at least one variable (node production level) change from positive to zero. With respect to a certain solution $x = (x_1, x_2, ..., x_N)$, we shall need the following notations to describe the algorithm:

$$\begin{aligned} \mathcal{A}(n) &= \{m \in \mathcal{T}(n)\backslash\{n\} : x_m > 0,\ x_k = 0 \quad \forall\ k \in \bar{\mathcal{P}}(m)\backslash\mathcal{P}(n)\}, \\ s_n &= \sum_{m\in\mathcal{A}(n)} c_m, \qquad \text{and} \\ \Delta_n &= \min_{m\in\mathcal{A}(n)} x_m. \end{aligned}$$

Note that $\mathcal{A}(n)$ is the set of closest descendants of n with positive production levels. This is the set of nodes from which production may be shifted up to node n. The primal scheme is detailed in Algorithm 1.

The algorithm first initializes the solution to x^0. Then, starting from a non-leaf node k with the largest index, the algorithm first compares the total production cost s_k of the nodes in $\mathcal{A}(k)$ with the production cost c_k of node k; then, if $s_k > c_k$, the algorithm shifts the minimum production Δ_k amongst the nodes in $\mathcal{A}(k)$ to node k. Figure 3 illustrates the primal algorithm for the example scenario tree in Figure 2. The first scenario tree in

Algorithm 1 The primal algorithm

```
1:  set $x_n^* = \max\{0, d_n - \sum_{m \in \bar{\mathcal{P}}(n)} x_m^*\}$ for all $n = 1, \ldots, N$
2:  set $k = \max\{n \in \mathcal{T} : \mathcal{T}(n) \backslash \{n\} \neq \emptyset\}$.
3:  while $k \geq 1$ do
4:      compute $\mathcal{A}(k)$, $s_k$ and $\Delta_k$.
5:      if $c_k < s_k$ then
6:          update the solution corresponding to the nodes in $\mathcal{A}(k) \cup \{k\}$ as follows:
            $x_m^* = \begin{cases} x_k^* + \Delta_k & \text{if } m = k \\ x_m^* - \Delta_k & \text{for all } m \in \mathcal{A}(k) \end{cases}$
7:      else
8:          set $k = k - 1$
9:      end if
10: end while
11: return $x^*$
```

Figure 3 illustrates the initial solution. The next tree illustrates the iteration corresponding to node $k = 3$ (the non-leaf node with the largest index). Here $\mathcal{A}(3) = \{7\}$, $s_3 = 6$, $c_3 = 4$, and $\Delta_3 = 1$. Thus 1 unit of production is shifted up from node 7 to node 3. The next iteration considers node $k = 2$. Here $\mathcal{A}(2) = \{4, 5\}$, $s_2 = 5$, $c_2 = 3$, and $\Delta_2 = 3$. Thus 3 units of production are shifted from node 4 and node 5 to node 2. The remaining iterations are similar.

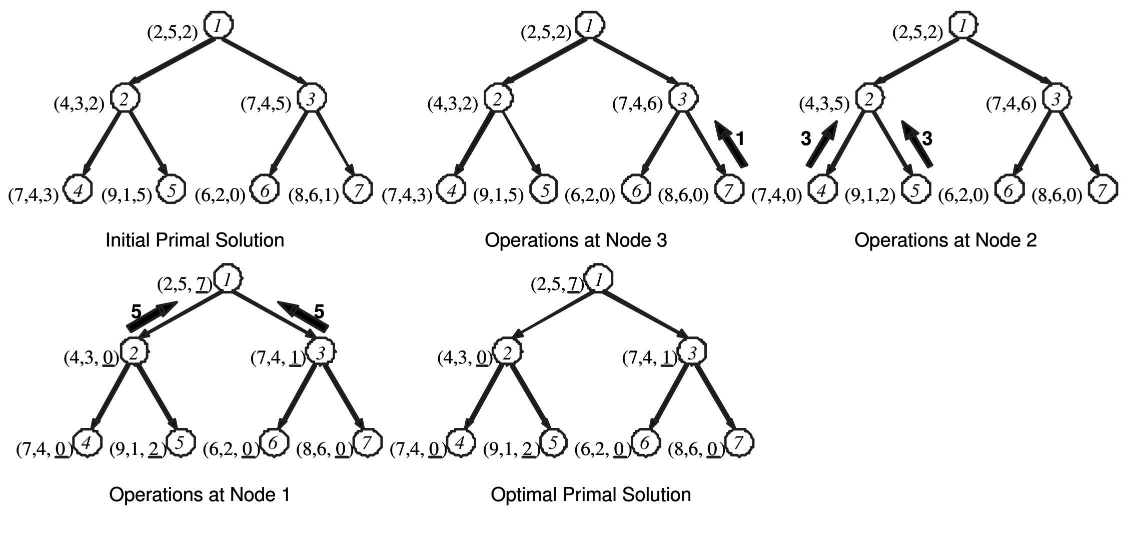

Figure 3: The primal algorithm (the numbers in parenthesis indicate (d_n, c_n, x_n))

2.3.2 The dual algorithm

Consider the dual of (5):

$$\begin{aligned} \max \quad & \sum_{n\in\mathcal{T}} d_n\pi_n \\ \text{s.t.} \quad & \sum_{m\in\mathcal{T}(n)} \pi_m \le c_n \quad \forall\, n\in\mathcal{T} \\ & \pi_n \in \mathbb{R}^+ \qquad\quad \forall\, n\in\mathcal{T}. \end{aligned} \tag{6}$$

Here we propose an algorithm for solving (6). The algorithm is based on the following observation.

LEMMA 1 *A solution $\pi = (\pi_1, \pi_2, ..., \pi_N)$ is feasible to (6) if and only if:*

$$0 \le \pi_n \le \min_{m\in\mathcal{P}(n)} \{c_m - \sum_{k\in\mathcal{T}(m)\setminus\{n\}} \pi_k\} \quad \forall\, n\in\mathcal{T}. \tag{7}$$

Furthermore, the second inequality is tight when π is optimal.

PROOF. Note that for every $m \in \mathcal{P}(n)$, the corresponding row in (6) is $\sum_{k\in\mathcal{T}(m)} \pi_k \le c_m$, i.e., $\pi_n + \sum_{k\in\mathcal{T}(m)\setminus\{n\}} \pi_k \le c_m$. So we require that $\pi_n \le c_m - \sum_{k\in\mathcal{T}(m)\setminus\{n\}} \pi_k$ for all $m \in \mathcal{P}(n)$. If we have a strict inequality for some n such that $\pi_n < \min_{m\in\mathcal{P}(n)}\{c_m - \sum_{k\in\mathcal{T}(m)\setminus\{n\}} \pi_k\}$, then all the constraints in which π_n appears are not tight and we can increase π_n to get a new feasible solution with greater objective value. $\square$

The dual algorithm is a greedy scheme, where we sort the dual variables according to decreasing objective function coefficients $\{d_n\}_{n\in\mathcal{T}}$, i.e., indexing scheme 2, and then set their values sequentially to make (7) tight. The scheme is detailed in Algorithm 2.

Figure 4 illustrates the dual algorithm for the example in Figure 2. Note that the nodes are indexed according to scheme 2. The first tree illustrates the initial dual solution. The next tree illustrates the first iteration, where $k = 1$, and we set $\pi_1^* = 1$. The costs on the nodes on $\bar{\mathcal{P}}(1) = \{6, 7\}$ are reduced to $c_6^1 = c_6^0 - \pi_1^* = 3 - 1 = 2$ and $c_7^1 = c_7^0 - \pi_1^* = 5 - 1 = 4$. The remaining iterations proceed similarly.

Algorithm 2 The dual algorithm

```
1: label the nodes of $\mathcal{T}$ according to the indexing scheme 2
2: set $\pi_n^* = 0$ for all $n = 1, \ldots, N$
3: set $c_n^0 = c_n$ for all $n = 1, \ldots, N$
4: for $k = 1, ..., N$ do
5:     set $\pi_k^* = \min_{n \in \mathcal{P}(k)} \{c_n^{k-1}\}$
6:     set
       $c_n^k = \begin{cases} c_n^{k-1} - \pi_k^* & \text{for all } n \in \mathcal{P}(k) \\ c_n^{k-1} & \text{otherwise} \end{cases}$
7: end for
8: return $\pi^*$
```

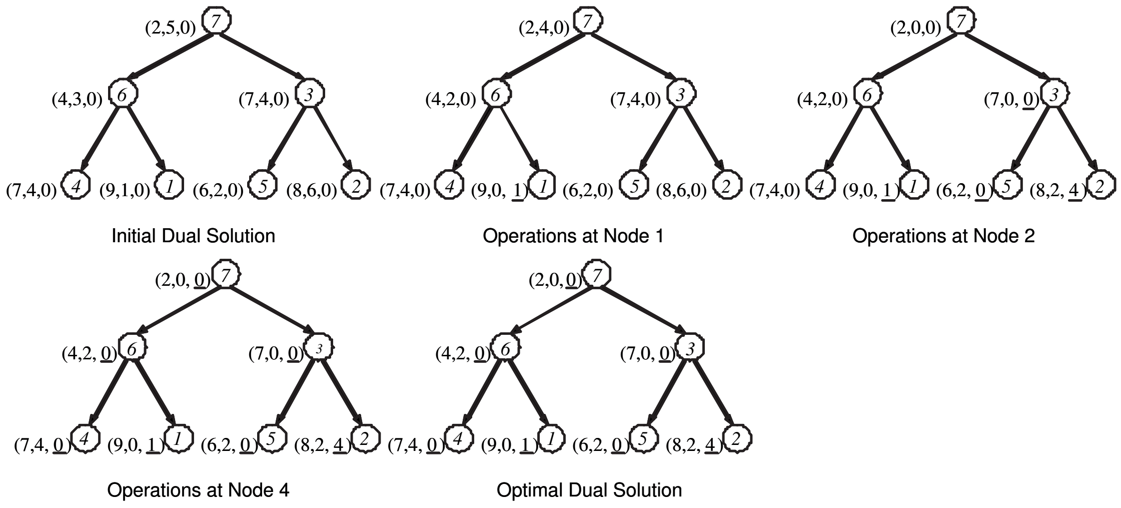

Figure 4: The dual algorithm (the numbers in parenthesis indicate (d_n, c_n, π_n))

2.4 Validity and complexity

2.4.1 Proof of validity

The following results establish the correctness of the proposed algorithms. We assume that the nodes are indexed according to scheme 2.

LEMMA 2 *At any iteration $k \in \{1, \ldots, N\}$ of the dual algorithm, the dual solution π^* satisfies*

$$\sum_{m \in \mathcal{T}^k(n)} \pi_m^* \leq c_n \qquad \forall\, n \in \mathcal{T}, \text{ and} \tag{8}$$

$$\sum_{m \in \mathcal{T}^k(n)} \pi_m^* = \sum_{m \in \mathcal{T}(n)} \pi_m^* = c_n \qquad \forall\, n \in \text{argmin}_{m \in \mathcal{P}(k)} \{c_m^{k-1}\}, \tag{9}$$

where $\mathcal{T}^k(n) = \mathcal{T}(n) \cap \{1, 2, ..., k\}$.

PROOF. By induction on k, it is easy to see that $c_n^k = c_n - \sum_{m\in\mathcal{T}^k(n)} \pi_m^*$ for all n. Also, since $n \in \mathcal{P}(k)$ if and only if $k \in \mathcal{T}(n)$, we have $\pi_k^* \leq c_n^{k-1}$ for all $n \in \mathcal{P}(k)$. Therefore, $c_n^k \geq 0$ for any $n \in \mathcal{T}$, and (8) follows. Furthermore, by construction, the algorithm ensures $\sum_{m\in\mathcal{T}^k(n)} \pi_m^* = c_n$ for all $n \in \text{argmin}_{m\in\mathcal{P}(k)}\{c_m^{k-1}\}$. Since the feasibility constraints demand $\sum_{m\in\mathcal{T}(n)} \pi_m^* \leq c_n$, equation (9) then follows. □

In the following result we shall make use of the following notation. At any iteration k of the dual algorithm, let

$$m_k \in \text{argmin}_{m\in\mathcal{P}(k)}\{c_m^{k-1}\}$$

such that

$$c_n^{k-1} > c_{m_k}^{k-1} \quad \forall\, n \in \mathcal{P}(k)\backslash\mathcal{P}(m_k).$$

That is, m_k is the closest node to k (on $\mathcal{P}(k)$) that minimizes c_m^{k-1}. Correspondingly, (9) holds for the node $n = m_k$.

THEOREM 2 *The solution $x^* = (x_1^*, x_2^*, ...x_N^*)$ returned by the primal algorithm and the solution $\pi^* = (\pi_1^*, \pi_2^*, ..., \pi_N^*)$ returned by the dual algorithm are optimal for (5) and (6), respectively.*

PROOF. First note that x^* is a feasible solution to (5). This is because the initial solution x^0 is feasible, and by construction, each shifting-up operation preserves feasibility by only shifting $\triangle_k = \min_{m\in\mathcal{A}(k)} x_m^*$ units. Lemma 2 guarantees the feasibility of the dual solution π^* (let $k = N$ in (8)). It remains to show that x^* and π^* satisfy complementary slackness, i.e., for all $n \in \mathcal{T}$:

$$\begin{aligned} \pi_n^* > 0 \quad &\Rightarrow \quad \textstyle\sum_{m\in\mathcal{P}(n)} x_m^* = d_n \\ \textstyle\sum_{m\in\mathcal{T}(n)} \pi_m^* < c_n \quad &\Rightarrow \quad x_n^* = 0 \end{aligned} \tag{10}$$

We shall show that the above conditions hold by induction. We assume that the nodes are indexed according to scheme 2.

The base case: Consider node 1. We show that (10) holds for all nodes $n \in \mathcal{T}(m_1)$. By the definition of m_1, we have that $\pi_1^* = c_{m_1}$. The dual constraint corresponding to node m_1 in (6) requires $\sum_{m\in\mathcal{T}(m_1)}\pi^*_m \leq c_{m_1}$. Therefore, for all $n \in \mathcal{T}(m_1)\backslash\{1\}$, we have $\pi^*_n = 0$ and $m_n = m_1$. Also, by our assumption of index system, we have $d_1 > d_n$ for all $n \in \bar{\mathcal{P}}(1)$ (otherwise n will be indexed 1). So in x^0 we must have $x^0_1 > 0$ and $\sum_{m\in\mathcal{P}(1)} x^0_m = d_1$. Furthermore, for all $n \in \mathcal{T}(1)\backslash\{1\}$, $x^0_n = 0$ since $d_n \leq d_1$. According to primal algorithm, we have $\sum_{m\in\mathcal{P}(1)} x^0_m = \sum_{m\in\mathcal{P}(1)} x^*_m$ since we only shift quantity along the tree towards the root. If $m_1 \neq 1$ (notice $m_1 \in \mathcal{P}(1)$), observe that $c_n > c_{m_1}$ for all $n \in \mathcal{P}(1)\backslash\mathcal{P}(m_1)$. So in the primal algorithm, when we process node m_1, all the productions of nodes $n \in \mathcal{P}(1)\backslash\mathcal{P}(m_1)$ will be shifted up, i.e., we must deplete any positive quantity along the path from 1 to m_1 (not including the latter). After we finish processing m_1, we have $x_n = 0$ for all $n \in \mathcal{P}(1)\backslash\mathcal{P}(m_1)$. Also, in the following iterations, the values of primal variables in $\mathcal{T}(m_1)\backslash\{m_1\}$ can not increase anymore (which are all zeroes). Therefore in final solution we have:

$$\begin{aligned} \textstyle\sum_{m\in\mathcal{P}(m_1)} x^*_m &= \textstyle\sum_{m\in\mathcal{P}(1)} x^0_m = d_1 \\ x^*_m &= 0 \quad \forall m \in \mathcal{T}(m_1)\backslash\{m_1\} \end{aligned}$$

The second equality comes from the first and the fact that $d_m \leq d_1$ for all $m \in \mathcal{T}(m_1)\backslash\{1\}$. Now we can check condition (10). Note that the only positive dual variable in $\mathcal{T}(m_1)$ is π^*_1 and we have shown that $\sum_{m\in\mathcal{P}(1)} x^*_m = \sum_{m\in\mathcal{P}(m_1)} x^*_m = d_1$. Since $\sum_{m\in\mathcal{T}(m_1)}\pi^*_m = c_{m_1}$, we claim that $\{n \in \mathcal{T}(m_1) : \sum_{m\in\mathcal{T}(n)}\pi^*_m < c_n\} \subseteq \mathcal{T}(m_1)\backslash\{m_1\}$. The conclusion holds because $x^*_n = 0$ for all $n \in \mathcal{T}(m_1)\backslash\{m_1\}$.

The induction step: Assume that (10) holds for all nodes in $\mathcal{T}(m_1)\cup\mathcal{T}(m_2)\cup\cdots\cup\mathcal{T}(m_k)$. First, we define $\mathcal{H}(k) = \{1, 2, ..., k\}$, $\mathcal{R}(k) = \{m_n : n \in \mathcal{H}(k)\}$ and $\mathcal{F}(k) = \cup\{\mathcal{T}(m_n) : n \in \mathcal{H}(k)\}$. If $k+1 \in \mathcal{T}(m_n)$ for some $n \in \mathcal{H}(k)$, then there is no need to check $k+1$ since the corresponding conditions are already satisfied. If this is not the case, i.e., $k+1 \notin \mathcal{F}(k)$, we examine conditions (10) for nodes in $\mathcal{T}(m_{k+1})\backslash\mathcal{F}(k)$.

Note for any $\mathcal{T}(m)$ and $\mathcal{T}(n)$ $(m \neq n)$, there are only three exclusive cases: $\mathcal{T}(m) \subset \mathcal{T}(n)$, or $\mathcal{T}(m) \subset \mathcal{T}(n)$, or $\mathcal{T}(m)\cap\mathcal{T}(n) = \emptyset$. In the third case, $\mathcal{T}(m)$ and $\mathcal{T}(n)$ form an independent pair. Without loss of generality, we can assume the trees $\mathcal{T}(m_1)$, $\mathcal{T}(m_2)$, ..., $\mathcal{T}(m_k)$ are pairwise independent. Notice for all $m \in \mathcal{P}(k+1)$ and $n \in \mathcal{H}(k)$, either

$\mathcal{T}(m_n) \cap \mathcal{T}(m) = \emptyset$ or $\mathcal{T}(m) \supset \mathcal{T}(m_n)$ exclusively. Also, according to (8) and (9): $c_n \geq \sum_{m \in \mathcal{T}^k(n)} \pi^*_m$ for all $n \in \mathcal{P}(k+1)$, and $\sum_{m \in \mathcal{T}^k(n)} \pi^*_m = \sum_{m \in \mathcal{T}(n)} \pi^*_m = c_n$ for all $n \in \mathcal{R}(k)$. So:

$$\sum_{m \in \mathcal{T}^k(n)} \pi^*_m = \sum_{m \in \mathcal{T}(n) \cap \mathcal{F}(k)} \pi^*_m = \sum_{m \in \mathcal{T}(n) \cap \mathcal{R}(k)} c_m \leq c_n \quad \forall n \in \mathcal{P}(k+1) \tag{11}$$

On the other hand, according to index system 2, we have $d_{k+1} > d_n$ for all $n \in \bar{\mathcal{P}}(k+1)$. Therefore, $x^0_{k+1} > 0$ and $\sum_{m \in \mathcal{P}(k+1)} x^0_m = d_{k+1}$. Furthermore, note for all $n \in \mathcal{H}(k)$ such that $m_n \in \mathcal{T}(k+1)$, we have $k+1 \in \bar{\mathcal{P}}(n)$, so $d_n > d_{k+1}$ (otherwise $k+1$ will be checked before n). For all $m \in \mathcal{T}(k+1) \backslash \mathcal{F}(k) \backslash \{k+1\}$, $x^0_m = 0$ since $d_m \leq d_{k+1}$. Therefore, according to the primal algorithm, we also have $\sum_{m \in \mathcal{P}(k+1)} x^*_m = d_{k+1}$. If $m_{k+1} \neq k+1$ (note $m_{k+1} \in \mathcal{P}(k+1)$), then according to the definition of m_{k+1}, for all $n \in \mathcal{P}(k+1) \backslash \mathcal{P}(m_{k+1})$ we have $\pi^*_{k+1} = c_{m_{k+1}} - \sum_{m \in \mathcal{T}(m_{k+1}) \cap \mathcal{R}(k)} c_m < c_n - \sum_{m \in \mathcal{T}(n) \cap \mathcal{R}(k)} c_m$. Therefore:

$$c_{m_{k+1}} < \sum_{m \in (\mathcal{T}(m_{k+1}) \backslash \mathcal{T}(n)) \cap \mathcal{R}(k)} c_m + c_n \tag{12}$$

Combining (11) and (12), by the same reasoning as in the base case, we conclude that when we are processing node m_{k+1} in the primal algorithm, we must deplete any positive quantity along the path $\mathcal{P}(k+1) \backslash \mathcal{P}(m_{k+1})$. That is, after we finish processing node m_{k+1}, we have $x_m = 0$ for all $m \in \mathcal{P}(k+1) \backslash \mathcal{P}(m_{k+1})$. In the final solution x^*, we have:

$$\begin{aligned} \textstyle\sum_{m \in \mathcal{P}(k+1)} x^*_m &= \textstyle\sum_{m \in \mathcal{P}(m_{k+1})} x^*_m = d_{k+1} \\ x^*_m &= 0 \quad \forall m \in \mathcal{T}(m_{k+1}) \backslash \mathcal{F}(k) \backslash \{m_{k+1}\} \end{aligned}$$

The second equality comes from the first and the fact that $d_{k+1} \geq d_n$ for all node n in the set $\mathcal{T}(m_{k+1}) \backslash \mathcal{F}(k) \backslash \{m_{k+1}\}$.

Note that $\mathcal{T}(m_{k+1}) \backslash \mathcal{F}(k)$ is the collection of all nodes in $\mathcal{T}(m_{k+1})$ for which we need to check complementary slackness (all other nodes in it have been checked before). To verify condition (10), note that the only possible positive dual variable in this set is π^*_{k+1} and we have shown that $\sum_{m \in \mathcal{P}(k+1)} x^*_m = d_{k+1}$. Also note $\sum_{m \in \mathcal{T}(m_{k+1})} \pi^*_m = c_{m_{k+1}}$, so $\{n \in \mathcal{T}(m_{k+1}) \backslash \mathcal{F}(k) : \sum_{m \in \mathcal{T}(n)} \pi^*_m < c_n\} \subseteq \mathcal{T}(m_{k+1}) \backslash \mathcal{F}(k) \backslash \{m_{k+1}\}$. The conclusion then holds since $x^*_n = 0$ for all $n \in \mathcal{T}(m_{k+1}) \backslash \mathcal{F}(k) \backslash \{m_{k+1}\}$. □

2.4.2 Complexity

To compute the complexity of the proposed algorithms, we assume that the scenario tree is complete with T levels and B branches per non-leaf node.

THEOREM 3 *The complexity of the dual algorithm is $\mathcal{O}(N \log N \log(\log N))$.*

PROOF. First, we sort the demands of N nodes, whose complexity is $N \log N$. Second, we have at most N minimization procedures. Each minimization will concern at most T numbers, whose complexity is $T \log T$. The updating will cost us at most T operations. Therefore, the total number of operations will be no more than $N \log N + N(T \log T + T)$. Notice the total number of nodes in the tree is $N = \sum_{t=0}^{T-1} B^t = \frac{B^T-1}{B-1}$. So $T \sim \mathcal{O}(\log N)$ and $\mathcal{O}(N \log N) \leq \mathcal{O}(NT \log T)$. The complexity of the algorithm is no greater than $\mathcal{O}(NT \log T)$, or $\mathcal{O}(N \log N \log(\log N))$. $\square$

Before we can compute the complexity of the primal algorithm , we need to describe in detail how to compute and update $\mathcal{A}$, s and Δ. Given any solution x^* at a particular iteration, let

$$\begin{aligned} \mathcal{S}^0(n) &= \{m \in \mathcal{T} : a(m) = n, x_m^* = 0\} \\ \mathcal{S}^+(n) &= \{m \in \mathcal{T} : a(m) = n, x_m^* > 0\}. \end{aligned}$$

For the first time a node k is considered, $\mathcal{A}(k)$, s_k and Δ_k can be computed as follows:

$$\begin{aligned} \mathcal{A}(k) &= (\cup_{m \in \mathcal{S}^0(k)} \mathcal{A}(m)) \cup \mathcal{S}^+(k) \\ s_k &= \textstyle\sum_{m \in \mathcal{S}^+(k)} c_m + \sum_{m \in \mathcal{S}^0(k)} s_m \\ \Delta_k &= \min\{\min\{x_m : m \in \mathcal{S}^+(k)\}, \min\{\Delta_m : m \in \mathcal{S}^0(k)\}\} \end{aligned}$$

Each time after we update the solution in node k, for each n in $\mathcal{A}(k)$ such that x_n decreases to 0, $\mathcal{A}(k)$, s_k and Δ_l for all $l \in (\bar{\mathcal{P}}(n) \backslash \mathcal{P}(k)) \cup \{k\}$ can be updated as follows:

$$\begin{aligned} \mathcal{A}(k) &= (\mathcal{A}(k) \backslash \{n\}) \cup \mathcal{A}(n) \\ s_k &= s_k - c_n + s_n, \\ \Delta_l &= \min\{\min\{x_m : m \in \mathcal{S}^+(l)\}, \min\{\Delta_m : m \in \mathcal{S}^0(l)\}\}. \end{aligned}$$

Note that Δ_l is updated according to the order of decreasing index:

THEOREM 4 *The complexity of the primal algorithm is $\mathcal{O}(N^2)$.*

PROOF. First, to compute x^0, we need to consider N nodes, each of which needs at most T operations. So the complexity is $\mathcal{O}(NT)$. Second, in each iteration, the production in at least one node will be depleted, so the algorithm will end in N iterations. Third, for each iteration, if we compute $\mathcal{A}(k)$, s_k and Δ_k for the first time, the complexities are $\mathcal{O}(N)$, $\mathcal{O}(B)$ and $\mathcal{O}(B\log B)$, respectively. To update $\mathcal{A}(k)$ and s_k, we need $\mathcal{O}(N)$ and $\mathcal{O}(B)$ operations. To update the current solution, we need $\mathcal{O}(|\mathcal{A}(k)|)$ operations, which has an upper bound of $\mathcal{O}(N)$. Finally, we consider the effort in updating Δ. For each depleted node n in $\mathcal{A}(k)$, we need to update all Δ_l along the path from n to k, which has an upper bound of $\mathcal{O}(TBlogB)$. In each iteration, there can be several nodes in $\mathcal{A}(k)$ that are depleted, however, then the number of iterations will also decrease correspondingly. Therefore, the dominant operation is constructing and updating $\mathcal{A}(k)$, whose total complexity is $\mathcal{O}(N^2)$. $\square$

Therefore, the complexity of both algorithms is within $\mathcal{O}(N^2)$. Since (2) can be transformed to (3) in $\mathcal{O}(N^2)$ operations. So we can also solve (2) in $\mathcal{O}(N^2)$ operations (when we have integer demands).

2.5 Computational results

Finally, we present some computational results using the primal and the dual algorithms. We consider a total of 18 scenario trees with the number of stages (T) varying from 5 to 13, and the number of branches (B) for each non-leaf node varying from 2 to 9. We use uniform distributions to generate the parameters (δ_n, α_n, β_n, p_n) for (2) corresponding to each node of the tree. Then we transform these parameters to the parameters (d_n, c_n) for (3). We generate 10 data sets corresponding to each of the 18 scenario trees. We compare the running times of the proposed Primal and dual algorithms with that of the Simplex solver in CPLEX 8.1. All numerical experiments are conducted on an IBM PC with 1024 MB RAM and a PENTIUM4 1.6GHz processor.

Table 1 presents the results. The columns in the table are arranged according to the number of stages (T), the number of branches (B), total number of nodes (N), the actual running time of Simplex, the actual running time of the primal algorithm, the running time of the primal algorithm as a % of the running time of Simplex, the actual running time of the dual algorithm, the running time of the dual algorithm as a % of the running time of Simplex, respectively. The running times for a particular scenario tree are averages over 10 instances, and are in units of 10^{-3} CPU seconds. We can observe that the dual algorithm is the fastest algorithm, and the primal algorithm is the second. Both algorithms are much faster than the Simplex. In the worst case, the running times of Primal and dual algorithms are 18.8 % and 1.8 % of that of the Simplex algorithm, respectively. On average, the running times of primal and dual algorithms are 10.8% and 0.4 % percent, respectively, of that of the simplex algorithm.

Table 1: Comparison of CPU times (in seconds)

No.	T	B	N	Simplex	Primal	%	Dual	%
1	8	2	255	17.1	1.6	9.36	0	0
2	9	2	511	35.9	6.1	16.99	0	0
3	10	2	1023	114	14.1	12.37	0	0
4	11	2	2047	420.4	57.7	13.73	4.8	1.14
5	12	2	4095	1212.5	228.2	18.82	10.8	0.89
6	13	2	8191	8499.9	790.8	9.30	28.4	0.33
7	6	3	364	28	3.2	11.43	0	0
8	7	3	1093	186	20.1	10.81	1.6	0.86
9	8	3	3280	1161	154.5	13.31	6.3	0.54
10	5	4	341	26.5	0	0	0	0
11	6	4	1365	336.1	26.4	7.85	0	0
12	7	4	5461	5290.5	479.8	9.07	12.3	0.23
13	5	5	781	89	12.5	14.04	1.6	1.80
14	6	5	3906	3395.5	243.6	7.17	7.8	0.23
15	5	6	1555	318.8	35.8	11.23	1.6	0.50
16	5	7	2801	1173.2	117.4	10.01	4.7	0.40
17	5	8	4681	3159.2	368.8	11.67	11	0.35
18	5	9	7381	13000.1	925	7.12	23.6	0.18

2.6 Summary

In this chapter, we study a simple stochastic lot-sizing problem involving a single item, linear cost structures, and finite distributions for the stochastic cost and demand parameters. We develop a primal and a dual algorithm for a multi-stage stochastic linear programming formulation of the problem, both of which are much more efficient than the simplex method. This simple stochastic lot-sizing problem is a key substructure inherent in more general production planning problems dealing with uncertainty. Thus, the proposed algorithms can be used within solution approaches for such general problems. The next chapter explores this approach for a general capacity planning problem.

CHAPTER III

THE VALUE OF MULTI-STAGE STOCHASTIC PROGRAMMING IN CAPACITY PLANNING UNDER UNCERTAINTY

3.1 Introduction

Capacity planning, i.e., deciding the optimal timing and level of capacity acquisition and allocation, plays a crucial role in strategic level planning in a wide array of applications. This activity involves substantial commitment of capital resources and is marred by uncertainties in the long range forecasts, thereby making the associated decision problems very complex. For example, the initial investment in building a cutting edge semiconductor wafer fab of IBM is more than two and half billion dollars (cf. [42]), and every year the procurement of new tools to accommodate the high volatility in demand, product mix and technology could cost several million dollars (cf. [5, 28, 55, 56]).

Owing to the inherent complexities, quantitative models for economic capacity planning under uncertainty have been the subject of intense research since the early 1960s (cf. [36]). Early approaches for solving stochastic capacity expansion problems are restricted to a single resource and based on simplifying assumptions on the underlying stochastic processes to render analytical tractability (cf. [6, 19, 24, 37, 38]). More general stochastic programming based approaches that use scenarios to model the uncertain parameters within large-scale mathematical programs for multi-resource multi-item capacity planning have since been proposed (cf. [10, 22, 23]). Most of these stochastic programming approaches are based on the two-stage paradigm, wherein the capacity acquisition schedule for the entire (multi-period) planning horizon is decided "here-and-now," and capacity allocations are made on a period-by-period basis based on realized uncertainties and acquired capacities. In the context of semiconductor tool planning, such two-stage models are investigated in [5, 28, 33, 55, 56].

Multi-stage stochastic programming models extend the two-stage paradigm by allowing revised decisions in each time stage based upon the uncertainty realized so far (cf. [14, 49]). A multi-stage stochastic capacity planning model involving continuous capacity allocation decisions and fixed charge expansion costs is considered in [3]. The authors developed a LP-relaxation based heuristic for this problem and proved, via a probabilistic analysis, that the heuristic is asymptotically optimal in the number of planning stages.

Motivated by applications in semiconductor tool planning, we address a general multi-stage stochastic capacity planning model involving discrete capacity acquisition decisions. Our model generalizes earlier two-stage approaches considered in [5, 28, 55, 56], by allowing for revision of the capacity expansion plan as more information regarding the uncertainties is revealed. We provide analytical bounds for the *value of multi-stage stochastic programming* afforded over two-stage approaches. By exploiting a special lot-sizing substructure inherent in the problem, we develop an efficient approximation scheme for the multi-stage problem and prove that the proposed scheme is asymptotically optimal. Our asymptotic analysis is significantly different from that of [3], since we consider discrete capacity acquisition levels and do not make any assumptions regarding the distributions of the underlying stochastic parameters.

3.2 Model development

In this section we present a mathematical formulation for the stochastic capacity planning problem under consideration. We first describe a specific deterministic capacity planning formulation related to semiconductor tool planning, and then discuss deterministic and stochastic generalizations of this model.

3.2.1 A deterministic model for semiconductor tool planning

Consider a wafer fab consisting of M tool types, that can process N types of wafers. Each product (wafer type) goes through a subset of K processing steps, each of which can be performed on one or more tool types. Let h_{ijk} denote the time (in hours) required by processing step k $(1, \ldots, K)$ on wafer type j $(1, \ldots, N)$ on tool type i $(1, \ldots, M)$. We set $h_{ijk} = 0$ if step k is not needed for wafer type j, and $h_{ijk} = \infty$ if step k is required for wafer type j but cannot be performed on tool type i. Consider now a planning horizon of T periods. Let us use variables x_{it}, u_{jt}, v_{ijkt}, and w_{jt}, to denote the number of tools of type i purchased in period t $(1, \ldots, T)$, the shortage (in units of wafer starts) of wafer type j in period t, the allocation of processing step k of wafer type j to tool type i in period t, and the production of wafer type j in period t, respectively. In addition to h_{ijk}, let us also consider the problem parameters a_{it}, b_{jt}, c_i, and d_{jt} corresponding to the (discounted) cost of tool type i in period t, the penalty cost of unit shortage in wafer type j in period t, the per-period capacity (in hours) of one tool of type i, and the per-period demand (in wafer starts) of wafer type j in period t, respectively.

With the above notation, an optimization model for multi-period (deterministic) scheduling of tool purchases and the allocation of tool capacity to production so as to minimize total tool purchase costs and shortage penalties can be stated as follows:

$$\begin{array}{lll}
\min & \sum_{t=1}^{T} \Big[\sum_{i=1}^{M} a_{it} x_{it} + \sum_{j=1}^{N} b_{jt} u_{jt} \Big] & \\
\text{s.t.} & \sum_{j=1}^{N} \sum_{k=1}^{K} h_{ijk} v_{ijkt} \le c_i \Big(\sum_{\tau=1}^{t} x_{i\tau} \Big) & \forall\ i,\ t \\
& \sum_{i=1}^{M} v_{ijkt} \ge w_{jt} & \forall\ j,\ k,\ t \\
& w_{jt} + u_{jt} \ge d_{jt} & \forall\ j,\ t \\
& u_{jt},\ v_{ijkt},\ w_{jt} \in \mathbb{R}_+ & \forall\ i,\ j,\ k,\ t \\
& x_{it} \in \mathbb{Z}_+ & \forall\ i,\ t.
\end{array} \tag{13}$$

For any period t, the first constraint in model (13) assures that the total processing requirement (in hrs) allocated to tool i cannot exceed the installed capacity. The second constraint enforces that the actual production of wafer type j is equal to the number of wafers that has completed all of the required K processing steps. The third constraint enforces that the production and shortage together should exceed the demand. The fourth constraint enforces non-negativity of the production-allocation-shortage variables. The fifth constraint enforces the integrality of the tool purchase decisions. Model (13) is a multi-period extension of the tool planning model described by [55, 56].

3.2.2 A generic capacity planning model

The semiconductor tool planning model (13) is a special case of the following generic capacity acquisition and allocation model:

$$\begin{array}{lll} \min & \sum_{t=1}^{T} \Big[\alpha_t x_t + \beta_t y_t\Big] & \\ \text{s.t.} & A_t y_t \leq \sum_{\tau=1}^{t} x_\tau & \forall\ t \\ & B_t y_t \geq \delta_t & \forall\ t \\ & y_t \in \mathbb{R}_+^J,\ x_t \in \mathbb{Z}_+^I & \forall\ t. \end{array} \tag{14}$$

In (14), the vector of decision variables x_t represents the capacity acquisition decisions for a set of I resources, and the vector of decision variables y_t represents the operational level allocation of capacity to a set of J tasks, in period t. The vectors α_t, β_t and δ_t represent acquisition costs, allocation costs, and demands, respectively. The matrices A_t and B_t represent resource-task utilization coefficients.

To see the connection between models (13) and (14), note that x_t and y_t correspond to the tool purchase and production-allocation-shortage decision vectors for period t, respectively; α_t, β_t, and δ_t correspond to the tool cost, shortage penalty, and demand vectors for period t, respectively; the matrix A_t corresponds to the coefficients of the first set of constraint in (13), and the matrix B_t corresponds to the coefficients of the second and third set of constraint in (13).

3.2.3 Stochastic programming extensions

Let us now extend the deterministic capacity planning model (14) to a stochastic setting. We assume that the uncertain problem parameters $(\alpha_t, \beta_t, \delta_t, A_t, B_t)$ evolve as discrete time stochastic processes with a finite probability space. This information structure can be interpreted as a scenario tree where the nodes in stage (or level) t of the tree constitute the states of the world that can be distinguished by information available up to time stage t. There are a total T stages in the tree. Each node n of the scenario tree, except the root ($n = 1$), has a unique parent $a(n)$, and each non-terminal node n is the root of a sub-tree $\mathcal{T}(n)$. The probability associated with the state of the world in node n is p_n. The set $\mathcal{S}_t$ denotes the nodes corresponding to time stage t, and t_n is the time stage corresponding to node n. The path from the root node to a node n will be denoted by $\mathcal{P}(n)$. If n is a terminal (leaf) node, i.e., $n \in \mathcal{S}_T$, then $\mathcal{P}(n)$ corresponds to a *scenario*, and represents a joint realization of the problem parameters over all periods. There are S leaf nodes corresponding to S scenarios, i.e., $S = |\mathcal{S}_T|$. We denote the whole tree $\mathcal{T}(1)$ by $\mathcal{T}$. The number of nodes in the whole tree is N_T. In this chapter, we use the notation N_T to emphasize the fact that N_T is a function of T. Usually N_T increases exponentially with increasing T. The stochastic problem parameters are given by the sequence $\{\alpha_n, \beta_n, \delta_n, A_n, B_n\}_{n \in \mathcal{T}}$.

Let us first consider a two-stage model where the first-stage involves deciding the capacity acquisition plan for all periods, regardless of the state of the world, and the second-stage consists of deciding on the capacity allocation plan subject to available capacity and the realized state. Thus the capacity acquisition variables are only indexed by time periods (since these do not change with the realized state) while the allocation decisions are indexed by the nodes of the scenario tree. With an objective of minimizing the *expected* total costs, a two-stage stochastic programming extension of (14) is as follows:

$$
\begin{array}{lll}
\min & \sum_{t=1}^{T} \overline{\alpha}_t x_t + \sum_{n \in \mathcal{T}} p_n \beta_n y_n & \\
\text{s.t.} & A_n y_n \leq \sum_{\tau=1}^{t_n} x_\tau & \forall\, n \in \mathcal{T} \\
& B_n y_n \geq \delta_n & \forall\, n \in \mathcal{T} \\
& y_n \in \mathbb{R}_+^J & \forall\, n \in \mathcal{T} \\
& x_t \in \mathbb{Z}_+^I & \forall\, t,
\end{array}
\qquad (15)
$$

where $\overline{\alpha}_t = \sum_{n \in \mathcal{S}_t} p_n \alpha_n$, i.e., the average capacity acquisition cost in period t. The stochastic programming model considered in [55, 56] is a special case of the model (15) when the number of periods $T = 2$. The model presented in [5, 28] is similar to (15), however, there the uncertain parameters are defined over scenarios (paths in the scenario tree) rather than nodes of the scenario tree. Note that (15) is a two-stage stochastic program with integer variables in the first stage.

As mentioned earlier, the two-stage model (15) does not allow any flexibility in the capacity acquisition plan with respect to the realized state of the world. To formulate a multi-stage stochastic programming model, we need to have the capacity acquisition decisions to be dependent on the realized state, and hence the resulting model is as follows:

$$\begin{array}{lll} \min & \sum_{n\in\mathcal{T}} p_n\Big[\alpha_n x_n + \beta_n y_n\Big] & \\ \text{s.t.} & A_n y_n \leq \sum_{m\in\mathcal{P}(n)} x_m & \forall\ n\in\mathcal{T} \\ & B_n y_n \geq \delta_n & \forall\ n\in\mathcal{T} \\ & y_n \in \mathbb{R}^J_+ & \forall\ n\in\mathcal{T} \\ & x_n \in \mathbb{Z}^I_+ & \forall\ n\in\mathcal{T}. \end{array} \tag{16}$$

This is a multi-stage stochastic integer program.

3.2.4 Stochastic lot-sizing substructure

The multi-stage capacity planning problem (16) can be restated as follows:

$$\begin{array}{lll} \min & \sum_{n\in\mathcal{T}} p_n\beta_n y_n + \sum_{i=1}^{I} Q_i(y) & \\ \text{s.t.} & B_n y_n \geq \delta_n & \forall\ n\in\mathcal{T} \\ & y_n \in \mathbb{R}^J_+ & \forall\ n\in\mathcal{T}, \end{array} \tag{17}$$

where

$$\begin{array}{rlll} Q_i(y) = & \min & \sum_{n\in\mathcal{T}} p_n\alpha_{in} x_{in} & \\ & \text{s.t.} & \sum_{m\in\mathcal{P}(n)} x_{im} \geq (A_n y_n)_i & \forall\ n\in\mathcal{T} \\ & & x_{in}\in\mathbb{Z}_+ & \forall\ n\in\mathcal{T}. \end{array} \tag{18}$$

The above reformulation decomposes the problem into two separate problems, one (17) involving the capacity allocation decisions, and the other (18) involving the capacity acquisition decisions. Note that we have used y to collectively denote the capacity allocation sequence $\{y_n\}_{n\in\mathcal{T}}$, and $(A_n y_n)_i$ to denote the i-th component of the I dimensional vector $A_n y_n$.

Observe that for a fixed sequence of capacity allocation decisions, the optimal capacity acquisition decisions can be obtained via solving (18) independently for each resource i. Key to the further developments in this chapter is the study of (18). Clearly, (18) belongs to the type of simple stochastic lot-sizing problem we study in Chapter II. According to Theorem 1, we have the following equation:

$$\begin{array}{rlll} Q_i(y) = & \min & \sum_{n\in\mathcal{T}} p_n\alpha_{in} x_{in} & \\ & \text{s.t.} & \sum_{m\in\mathcal{P}(n)} x_{im} \geq \lceil (A_n y_n)_i \rceil & \forall\ n\in\mathcal{T} \\ & & x_{in}\in\mathbb{R}_+ & \forall\ n\in\mathcal{T}. \end{array} \tag{19}$$

3.3 Value of multi-stage stochastic programming

Let the v^{TS} and v^{MS} denote the optimal objective function values of the two-stage (15) and multi-stage (16) models, respectively. For a given set of problem parameters, it is easily verified that any solution to (15) is feasible to (16), and the objective function values corresponding to this solution are equal in both problems, thus

$$v^{TS} \geq v^{MS}.$$

That is, the overall cost of the multi-stage solution is smaller than that of the two-stage solution. This should come as no surprise, since, the multi-stage solution offers more flexibility in the capacity acquisition decisions with respect to the uncertain states of the world. We refer to the difference between the optimal objective values of the two-stage and multi-stage formulations as the *value of multi-stage stochastic programming* (VMS):

$$\text{VMS} = v^{TS} - v^{MS}.$$

Unfortunately, the value of multi-stage stochastic programming comes at the expense of solving a much larger and difficult optimization model. Both (15) and (16) are stochastic integer programs, and in general, can be extremely difficult to solve. For our particular case, both models have the property that by fixing the capacity acquisition decisions (the x variables), we can break the problem down to independent capacity allocation problems (in the y variables) corresponding to each node of the scenario tree. Owing to this structure, Benders decomposition (cf. [9]) is particularly attractive for these problems. In case of (15) and (16), this would require us to solve master problems involving the integer variables x. While the two-stage model (15) involves $I \times T$ integer variables, the multi-stage model (16) involves $I \times N_T$ integer variables (recall that $N_T = |\mathcal{T}|$), and for any non-trivial scenario tree $N_T >> T$. Consequently the computational difficulty of (16) is significantly more than that of (15). If the VMS is small, then this additional computational effort may not be worthwhile. However, we need *a priori* estimates of VMS to analyze this tradeoff. Next, we first describe simple bounds on VMS for the stochastic lot-sizing problem (19) and then use these to get bounds on the VMS for the capacity planning problem (16).

3.3.1 VMS for the stochastic lot-sizing problem

Consider the linear relaxation of the multi-stage stochastic lot-sizing problem (18) and let v^M denote its optimal objective function value, i.e.,

$$\begin{aligned} v^M = \ \min\ & \textstyle\sum_{n\in\mathcal{T}} p_n\alpha_n x_n \\ \text{s.t.}\ & \textstyle\sum_{m\in\mathcal{P}(n)} x_m \geq d_n \quad \forall\ n\in\mathcal{T} \\ & x_n \in \mathbb{R}_+ \qquad\qquad \forall\ n\in\mathcal{T}. \end{aligned} \tag{20}$$

Note that we have dropped the constant term C from the objective. A two-stage model for the stochastic lot-sizing problem would require that the production decisions for each time period be the same irrespective of the state realized, i.e.,

$$\begin{aligned} v^T = \ \min\ & \textstyle\sum_{n\in\mathcal{T}} p_n\alpha_n x_n \\ \text{s.t.}\ & \textstyle\sum_{m\in\mathcal{P}(n)} x_m \geq d_n \quad \forall\ n\in\mathcal{T} \\ & x_n \in \mathbb{R}_+ \qquad\qquad \forall\ n\in\mathcal{T} \\ & x_m = x_n \qquad\qquad \forall\ m,n\in\mathcal{S}_t,\ \forall\ t. \end{aligned} \tag{21}$$

THEOREM 5 *Let*

$$\begin{aligned} \alpha^* &= \textstyle\max_{n\in\mathcal{T}} \alpha_n \\ \alpha_* &= \textstyle\min_{n\in\mathcal{T}} \alpha_n \\ d_T^* &= \textstyle\max_{n\in\mathcal{S}_T}(\max_{m\in\mathcal{P}(n)} d_m) \\ \overline{d}_T &= \textstyle\sum_{n\in\mathcal{S}_T} p_n(\max_{m\in\mathcal{P}(n)} d_m), \end{aligned}$$

then

$$\alpha_* d_T^* - \alpha^*\overline{d}_T \leq \text{VMS} = v^T - v^M \leq \alpha^* d_T^* - \alpha_*\overline{d}_T.$$

PROOF. Note that any feasible solution x for (20) has to satisfy

$$\begin{aligned} & \textstyle\sum_{m\in\mathcal{P}(n)} x_m \geq \max_{m\in\mathcal{P}(n)} d_m \quad \forall\ n\in\mathcal{S}_T \\ \Rightarrow\ & \textstyle\sum_{n\in\mathcal{S}_T} p_n\Big(\sum_{m\in\mathcal{P}(n)} x_m\Big) \geq \sum_{n\in\mathcal{S}_T} p_n(\max_{m\in\mathcal{P}(n)} d_m) \\ \Leftrightarrow\ & \textstyle\sum_{t=1}^{T}\sum_{n\in\mathcal{S}_t}\Big(\sum_{m\in\mathcal{S}_T\cap\mathcal{T}(n)} p_m\Big)x_n \geq \overline{d}_T \\ \Leftrightarrow\ & \textstyle\sum_{n\in\mathcal{T}} p_n x_n \geq \overline{d}_T, \end{aligned}$$

where the last step follows from the fact that

$$\sum_{m\in\mathcal{S}_T\cap\mathcal{T}(n)} p_m = p_n \quad \forall\ n\in\mathcal{T}.$$

Then if x^* is an optimal solution for (20), we have

$$v^M = \sum_{n \in \mathcal{T}} p_n \alpha_n x_n^* \geq \alpha_* \sum_{n \in \mathcal{T}} p_n x_n^* \geq \alpha_* \bar{d}_T. \tag{22}$$

Next, consider a feasible solution $\widehat{x}$ to (20), such that $\widehat{x}_n = \max_{m \in \mathcal{P}(n)} d_m - \max_{m \in \mathcal{P}(a(n))} d_m$ for all $n \in \mathcal{T}$, and $\max_{m \in \mathcal{P}(a(1))} d_m = 0$. Then

$$\begin{array}{rcl} v^M & \leq & \sum_{n \in \mathcal{T}} p_n \alpha_n \widehat{x}_n \\ & \leq & \alpha^* \sum_{t=1}^{T} \sum_{n \in \mathcal{S}_t} p_n (\max_{m \in \mathcal{P}(n)} d_m - \max_{m \in \mathcal{P}(a(n))} d_m) \\ & = & \alpha^* \sum_{n \in \mathcal{S}_T} p_n (\max_{m \in \mathcal{P}(n)} d_m) \\ & = & \alpha^* \bar{d}_T, \end{array} \tag{23}$$

where the third step follows the fact that

$$\sum_{m \in \mathcal{S}_{t+1} \cap \mathcal{T}(n)} p_m = p_n \quad \forall\ t,\ n \in \mathcal{S}_t.$$

In the two-stage model (21), since the production decision is identical for all nodes in any stage, it has to satisfy the largest possible cumulative demand in that stage, i.e., d_n can be replaced with $\tilde{d}_n = \max_{m \in \mathcal{S}_{t_n}} d_m$ in (21). Then, by applying the same analysis used for problem (20) to problem (21) with $\tilde{d}_n$ replacing d_n, it can be shown that

$$\alpha_* d_T^* \leq v^T \leq \alpha^* d_T^*. \tag{24}$$

Combining (22), (23), and (24), the claim follows. □

In Theorem 5, α^* is maximum cost in the whole scenario tree; α_* is the minimum cost in the whole scenario tree; d_T^* is in fact the maximum demand in the whole scenario tree; and $\bar{d}_T$ is the expected value of maximum demands in all the scenarios. Suppose that the cost parameters α_n are nearly constant, i.e., $\alpha^* \approx \alpha_* \approx \alpha$, then by Theorem 5

$$\text{VMS} \approx \alpha(d_T^* - \bar{d}_T),$$

i.e., the value of the multi-stage model depends only on the difference between d_T^* and $\bar{d}_T$. Thus, if there is little variability in the demand data, then the multi-stage model has little value. On the other hand, if there is large variability in the demand data, then the multi-stage model has a high value, and the two-stage model may provide bad quality decisions.

3.3.2 VMS for the capacity planning problem

We shall now describe a lower bound on the VMS for the multi-stage capacity planning model (16) based on the analysis in the previous section and an optimal solution to the LP relaxation of the two-stage model (15). Since this LP relaxation can be solved fairly quickly, we can use this lower bound estimate to justify additional computational effort required to solve the difficult multi-stage model (16).

THEOREM 6 *Let $\{y_n^{TLP}\}_{n\in\mathcal{T}}$ be the capacity allocation decisions in an optimal solution to the linear relaxation of the two-stage model (15). Then for each resource $i = 1, \ldots, I$, let $d_{in} = (A_n y_n^{TLP})_i$, $d_{iT}^* = \max_{n\in\mathcal{S}_T}(\max_{m\in\mathcal{P}(n)} d_{im})$, $\bar{d}_{iT} = \sum_{n\in\mathcal{S}_T} p_n(\max_{m\in\mathcal{P}(n)} d_{im})$, $\alpha_i^* = \max_{n\in\mathcal{T}} \alpha_{in}$, and $\alpha_{i*} = \min_{n\in\mathcal{T}} \alpha_{in}$, we have*

$$\text{VMS} \geq \sum_{i=1}^{I} \left(\alpha_{i*} d_{iT}^* - \alpha_i^* \bar{d}_{iT}\right) - \sum_{i=1}^{I} \alpha_{i1}.$$

PROOF. Note that

$$v^{TS} \geq \sum_{n\in\mathcal{T}} p_n \beta_n y_n^{TLP} + \sum_{i=1}^{I} v_i^T$$

where

$$\begin{array}{rll} v_i^T = & \min & \sum_{n\in\mathcal{T}} p_n \alpha_{in} x_{in} \\ & \text{s.t.} & \sum_{m\in\mathcal{P}(n)} x_{im} \geq d_{in} \quad \forall\ n \in \mathcal{T} \\ & & x_{in} \in \mathbb{R}_+ \qquad\qquad\quad \forall\ n \in \mathcal{T} \\ & & x_{im} = x_{in} \qquad\qquad\quad \forall\ m, n \in \mathcal{S}_t,\ \forall\ t. \end{array}$$

Since $\{y_n^{TLP}\}_{n\in\mathcal{T}}$ is a feasible capacity allocation for the multi-stage problem (16), we have

$$v^{MS} \leq \sum_{n\in\mathcal{T}} p_n \beta_n y_n^{TLP} + \sum_{i=1}^{I} o_i^M$$

where

$$\begin{array}{rcll} o_i^M & = & \min & \sum_{n\in\mathcal{T}} p_n\alpha_{in}x_{in} & \\ & & \text{s.t.} & \sum_{m\in\mathcal{P}(n)} x_{im} \geq d_{in} & \forall\ n\in\mathcal{T} \\ & & & x_{in}\in\mathbb{Z}_+ & \forall\ n\in\mathcal{T} \\ & = & \min & \sum_{n\in\mathcal{T}} p_n\alpha_{in}x_{in} & \\ & & \text{s.t.} & \sum_{m\in\mathcal{P}(n)} x_{im} \geq \lceil d_{in}\rceil & \forall\ n\in\mathcal{T} \\ & & & x_{in}\in\mathbb{R}_+ & \forall\ n\in\mathcal{T} \\ & = & \max & \sum_{n\in\mathcal{T}}[d_{in}+(\lceil d_{in}\rceil-d_{in})]\pi_{in} & \\ & & \text{s.t.} & \sum_{m\in\mathcal{T}(n)} \pi_{im} \leq p_n\alpha_{in} & \forall\ n\in\mathcal{T} \\ & & & \pi_{in}\in\mathbb{R}_+ & \forall\ n\in\mathcal{T}, \end{array}$$

in which the second equality comes from Theorem 1 and the third equality comes from linear program duality. We can further define:

$$\begin{array}{rcll} v_i^M & = & \min & \sum_{n\in\mathcal{T}} p_n\alpha_{in}x_{in} & \\ & & \text{s.t.} & \sum_{m\in\mathcal{P}(n)} x_{im} \geq d_{in} & \forall\ n\in\mathcal{T} \\ & & & x_{in}\in\mathbb{R}_+ & \forall\ n\in\mathcal{T} \\ & = & \max & \sum_{n\in\mathcal{T}} d_{in}\pi_{in} & \\ & & \text{s.t.} & \sum_{m\in\mathcal{T}(n)} \pi_{im} \leq p_n\alpha_{in} & \forall\ n\in\mathcal{T} \\ & & & \pi_{in}\in\mathbb{R}_+ & \forall\ n\in\mathcal{T}, \end{array}$$

Therefore,

$$\begin{array}{rccclll} o_i^M & \leq & v_i^M & + & \max & \sum_{n\in\mathcal{T}}(\lceil d_{in}\rceil-d_{in})\pi_{in} & \\ & & & & \text{s.t.} & \sum_{m\in\mathcal{T}(n)} \pi_{im} \leq p_n\alpha_{in} & \forall\ n\in\mathcal{T} \\ & & & & & \pi_{in}\in\mathbb{R}_+ & \forall\ n\in\mathcal{T} \\ & \leq & v_i^M & + & \max & \sum_{n\in\mathcal{T}} \pi_{in} & \\ & & & & \text{s.t.} & \sum_{m\in\mathcal{T}(n)} \pi_{im} \leq p_n\alpha_{in} & \forall\ n\in\mathcal{T} \\ & & & & & \pi_{in}\in\mathbb{R}_+ & \forall\ n\in\mathcal{T} \\ & = & v_i^M & + & \min & \sum_{n\in\mathcal{T}} p_n\alpha_{in}x_{in} & \\ & & & & \text{s.t.} & \sum_{m\in\mathcal{P}(n)} x_{im} \geq 1 & \forall\ n\in\mathcal{T} \\ & & & & & x_{in}\in\mathbb{R}_+ & \forall\ n\in\mathcal{T} \\ & = & v_i^M & + & \alpha_{i1}, & & \end{array}$$

where the second inequality comes from $\lceil d_{in} \rceil - d_{in} \leq 1$, the third equality comes from duality, the fourth equality comes from the fact that $p_1 = 1$ and an optimal solution to a stochastic lot-sizing problem with a cumulative demand of 1 unit in *every* node is to produce 1 unit in the root node. Therefore, we have:

$$\text{VMS} \geq \sum_{i=1}^{I} \Big(v_i^T - v_i^M - \alpha_{i1} \Big),$$

and the result follows the bounds (23) and (24) derived in the proof of Theorem 5. □

3.4 An approximation algorithm

In this section we develop an approximation algorithm for the multi-stage capacity planning problem (16). To appreciate the computational complexity of this problem, it can be shown that any instance of the NP-hard integer knapsack problem (cf. [25]) with I items can be polynomially transformed to a single period instance of the deterministic capacity planning problem (14). Since (14) is just a single scenario instance of the stochastic models (15) and (16), we have the following result.

THEOREM 7 *The deterministic capacity planning problem (*14*) and its stochastic counterparts (15) and (16) are NP-hard.*

PROOF. Consider (14). Let $T = 1$ and drop the subscript t. Let $J = I > 1$, δ be a real number, $\beta = 0$, A be identity matrix, and the ith entry of B be γ_i in (14), we get the following problem:

$$\begin{array}{ll} \min & \alpha x \\ s.t. & \sum_{i=1}^{I} \gamma_i x_i \geq \delta \\ & x \in \mathbb{Z}_+^I \end{array}$$

which is the NP-hard integer knapsack problem. □

Motivated by this intractability, we propose the approximation scheme outlined in Algorithm 3 for problem (16). The algorithm exploits the decomposable structure revealed by the reformulation (17)-(18) of the problem.

Algorithm 3 An approximation algorithm for (16)

```
1: Solve the LP relaxation of (16). Let $\{(x_n^{LP}, y_n^{LP})\}_{n\in\mathcal{T}}$ be an optimal solution and $v_{MS}^{LP}$
   be the optimal value. If $x_n^{LP}$ is integral for all $n$, stop and return $\{(x_n^{LP}, y_n^{LP})\}_{n\in\mathcal{T}}$.
2: For each resource $i = 1, 2, ..., I$, solve independent capacity acquisition (or stochastic
   lot-sizing) problems:
      $\min \; \sum_{n\in\mathcal{T}} p_n \alpha_{in} x_{in}$
      $\text{s.t.} \; \sum_{m\in\mathcal{P}(n)} x_{im} \geq \lceil (A_n y_n^{LP})_i \rceil \quad \forall\, n \in \mathcal{T}$
      $x_{in} \in \mathbb{R}_+ \quad \forall\, n \in \mathcal{T},$
   and let $x_{in}^H$ denote the corresponding solutions. Note that the integrality of the decision
   variables allows for the rounding up of $(A_n y_n^{LP})_i$.
3: For each $n \in \mathcal{T}$, solve independent capacity allocation problems:
      $\min \; \beta_n y_n$
      $\text{s.t.} \; A_n y_n \leq \sum_{m\in\mathcal{P}(n)} x_m^H$
      $B_n y_n \geq \delta_n, y_n \in \mathbb{R}_+^J,$
   and let $y_n^H$ denote the corresponding optimal solution.
4: Return $\{(x_n^H, y_n^H)\}_{n\in\mathcal{T}}$.
```

Step 1 of Algorithm 3 requires the solution of the LP relaxation of (16). This problem is a multi-stage stochastic linear program which can, in general, be solved by the Nested L-Shaped Decomposition algorithm (cf. [14, 49]). Step 2 requires the solution of I stochastic lot-sizing problems (19) which are multi-stage stochastic linear programs. In Chapter II, we have developed very efficient algorithms for this type of problems. In the following part, we present an efficient algorithm which is a refinement of the dual algorithm proposed in Chapter II. Finally, Step 3 requires the solution of independent simple linear capacity allocation problems for each node in the tree.

3.4.1 An efficient algorithm for the stochastic lot-sizing problem (19)

By virtue of Theorem 1 and the fact that the right-hand-sides of the stochastic lot-sizing problems solved in Step 2 of Algorithm 3 are integral, we only need to find an efficient scheme for the linear program

$$\begin{aligned} \min \quad & \textstyle\sum_{n\in\mathcal{T}} c_n x_n \\ \text{s.t.} \quad & \textstyle\sum_{m\in\mathcal{P}(n)} x_m \geq d_n \quad \forall\, n \in \mathcal{T} \\ & x_n \in \mathbb{R}_+ \quad \forall\, n \in \mathcal{T}, \end{aligned} \tag{25}$$

where we have used c_n to succinctly denote $p_n\alpha_n$. The dual of (25) is

$$\begin{array}{lll} \max & \sum_{n\in\mathcal{T}} d_n\pi_n & \\ \text{s.t.} & \sum_{m\in\mathcal{T}(n)} \pi_m \le c_n & \forall\ n \in \mathcal{T} \\ & \pi_n \in \mathbb{R}_+ & \forall\ n \in \mathcal{T}, \end{array} \tag{26}$$

Recall that we have developed very efficient primal and dual algorithms in Chapter II for this type of stochastic lot-sizing problems. Our proposed algorithm in this chapter is based on a refinement of the greedy dual algorithm for the dual problem (26) presented in Chapter II. The scheme takes advantage of complementary slackness conditions to recover primal optimal solutions. Algorithm 4 summarizes the proposed strategy. Here, we assume that the parameters c_n and d_n are strictly positive for all n. As in Chapter II, we use two different indexing schemes for the nodes in the tree $\mathcal{T}$:

Indexing scheme 1. The nodes in $\mathcal{T}$ are indexed $1, 2, \ldots, N_T$ in increasing order of their time stage, i.e., $t_1 \le t_2 \le \cdots \le t_{N_T}$. No particular ordering is imposed on the indices of the nodes in the same time stage. Thus the root node has an index of 1.

Indexing scheme 2. The nodes in $\mathcal{T}$ are indexed $1, 2, \ldots, N_T$ in decreasing order of the corresponding cumulative demand, i.e., $d_1 \ge d_2 \ge \cdots \ge d_{N_T}$. If $d_m = d_n$, then $m < n$ if $t_m < t_n$.

The illustration of the two indexing schemes are shown in Figure 2 (Chapter II, page 10).

The greedy dual step first assigns the largest dual value (as permitted by the constraints) to the node with the largest demand, and then considers the node with the next largest demand, and so on. The marker m_k is assigned the index of the node (closest on the path $\mathcal{P}(k)$) to k whose corresponding dual constraint becomes tight when the dual value for node k is set. Note that once k has a positive dual value, no other nodes in $\mathcal{T}(m_k)$ will be further considered (any other node in $\mathcal{T}(m_k)$ will satisfy the condition in line 4 of the dual step). Thus all nodes in $l \in \mathcal{T}(m_k)$ except for the ones with $m_l > 0$, will have $\pi_l^* = 0$. The marker

Algorithm 4 An algorithm for stochastic lot-sizing primal-dual pair (25)-(26).

The dual step:

```
1: label the nodes in $\mathcal{T}$ according to indexing scheme 2
2: initialize $\pi_n^* = 0$, $c_n^0 = c_n$ and $m_n = 0$ for all $n \in \mathcal{T}$
3: for $k = 1, \ldots, N_T$ do
4:    if there exists $n \in \mathcal{P}(k)$ such that $n = m_l$ for some $l \in \mathcal{T}(n)$ then
5:       break
6:    else
7:       set $\pi_k^* = \min_{n \in \mathcal{P}(k)} \{c_n^{k-1}\}$
8:       set $m_k = \text{argmin}_{n \in \mathcal{P}(k)} \{c_n^{k-1}\}$ such that $c_l^{k-1} > c_{m_k}^{k-1}$ for all $l \in \mathcal{P}(k) \setminus \mathcal{P}(m_k)$
9:       set $c_n^k = c_n^{k-1} - \pi_k^*$ if $n \in \mathcal{P}(k)$ and $c_n^k = c_n^{k-1}$ otherwise
10:   end if
11: end for
```

The primal step:

```
1: transform the node indices as well as $m_n$ to indexing scheme 1
2: initialize $x_n^* = 0$ for all $n \in \mathcal{T}$
3: for $n = 1, \ldots, N_T$ do
4:    if there exists $l \in \mathcal{T}(n)$ such that $n = m_l$ then
5:       set $x_n^* = d_l - \sum_{k \in \mathcal{P}(n) \setminus \{n\}} x_k^*$
6:    end if
7: end for
```

m_k is used in the primal step to set the primal variables such that complementary slackness conditions are satisfied. Note that according to the algorithm, only a node k with $m_k > 0$ could have a positive dual value, and a node n could have a positive primal value only if $n = m_l$ for some node $l \in \mathcal{T}(n)$.

The following results establish the validity of Algorithm 4. For the remainder of this section we use indexing scheme 2 for the node labels.

LEMMA 1 *In each iteration $k \in \{1, \ldots, N_T\}$ of the dual step of Algorithm 4, the dual solution π^* satisfies*

$$\sum_{m \in \mathcal{T}^k(n)} \pi_m^* \leq c_n \quad \text{for all } n \in \mathcal{T}, \tag{27}$$

where $\mathcal{T}^k(n) = \mathcal{T}(n) \cap \{1, 2, \ldots, k\}$.

PROOF. By induction on k, it can be seen that $c_n^k = c_n - \sum_{m \in \mathcal{T}^k(n)} \pi_m^*$. Also, $n \in \mathcal{P}(k)$ if and only if $k \in \mathcal{T}(n)$ and $\pi_k^* \leq c_n^{k-1}$ for all $n \in \mathcal{P}(k)$. Therefore, we always have $c_n^k \geq 0$

for any $n \in \mathcal{T}$. $\square$

Lemma 1 guarantees the feasibility of the dual solution π^* (let $k = N_T$ in (27)). Furthermore, we also have $\sum_{l \in \mathcal{T}^k(n)} \pi_l^* = c_n$ for all $n \in \text{argmin}_{m \in \mathcal{P}(k)}\{c_m^{k-1}\}$. Since dual feasibility of π^* implies $\sum_{l \in \mathcal{T}(n)} \pi_l^* \leq c_n$ and $\pi_n^* \geq 0$, we also have:

$$\sum_{l \in \mathcal{T}^k(n)} \pi_l^* = \sum_{l \in \mathcal{T}(n)} \pi_l^* = c_n \quad \forall\, n \in \text{argmin}_{m \in \mathcal{P}(k)}\{c_m^{k-1}\}. \tag{28}$$

That is, for any $l \in \mathcal{T}(n)$ such that $l \notin \{1, \ldots, k\}$, π_l^*=0.

LEMMA 2 *The primal solution $x^* = (x_1^*, x_2^*, \ldots, x_{N_T}^*)$ produced by the primal step of Algorithm 4 is feasible.*

PROOF. By construction, if a node n is such that $m_n > 0$, then the following equalities hold:

$$\sum_{m \in \mathcal{P}(n)} x_m^* = \sum_{m \in \mathcal{P}(m_n)} x_m^* = d_n. \tag{29}$$

Now consider a node n such that $m_n = 0$. Let $l \in \mathcal{P}(n)$ be such that $l = m_k$ for some node $k \in \mathcal{T}(m_k)$. Note that such a node l must always exists, since the root node is one such node. Suppose l be the closest (on the path $\mathcal{P}(n)$) such node to n. Note that $n, k \in \mathcal{T}(l)$ while $m_n = 0$ and $m_k > 0$, thus in the dual step node k must have been considered before node n, i.e., $d_k \geq d_n$. According to (29), $\sum_{m \in \mathcal{P}(n)} x_m^* = \sum_{m \in \mathcal{P}(m_k)} x_m^* = \sum_{m \in \mathcal{P}(k)} x_m^* = d_k \geq d_n$ (the first equality holds since $x_m^* = 0$ for $m \in \mathcal{P}(n) \backslash \mathcal{P}(m_k)$). $\square$

THEOREM 8 *The solutions x^* and π^* returned by Algorithm 4 are optimal solutions of (25) and (26), respectively.*

PROOF. Lemmas 1 and 2 have proven the feasibility of π^* and x^*. Here we show that π^* and x^* satisfies the complementary slackness conditions:

$$\pi_n^* > 0 \implies \sum_{m \in \mathcal{P}(n)} x_m^* = d_n \tag{30}$$

$$\sum_{m \in \mathcal{T}(n)} \pi_m^* < c_n \implies x_n^* = 0. \tag{31}$$

We prove by induction on the nodes indexed according to scheme 2.

The base case: Consider node 1. Note that $\pi_1^* = c_{m_1} > 0$, then (30) follows from (29). For all other nodes $n \in \mathcal{T}(m_1)$, $\pi_n^* = 0$. On the other hand, $\sum_{m \in \mathcal{T}(m_1)} \pi_m^* = c_{m_1}$, thus $\{n \in \mathcal{T}(m_1) : \sum_{m \in \mathcal{T}(n)} \pi_m^* < c_n\} \subseteq \mathcal{T}(m_1) \backslash \{m_1\}$. Then (31) holds, since $x_n^* = 0$ for all $n \in \mathcal{T}(m_1) \backslash \{m_1\}$. Note that we have verified the complementary slackness conditions for all nodes in $\mathcal{T}(m_1)$, and not just node 1.

The induction step: Assume that we have checked nodes $1, ..., k$, and now consider node $k+1$. If $m_{k+1} = 0$, then this node has already been checked since then $k+1 \in \mathcal{T}(m_j)$ for some $j < k+1$. So we assume that $m_{k+1} > 0$. Denote $\{1, 2, ..., k\}$ by $\mathcal{H}(k)$ and $\{m_1, m_2, ..., m_k\}$ by $\mathcal{R}(k)$. Also define $\mathcal{F}(k) = \cup\{\mathcal{T}(m_n) : n \in \mathcal{H}(k), m_n > 0\}$. We now examine the nodes in $\mathcal{T}(m_{k+1}) \backslash \mathcal{F}(k)$. Notice for all nodes n in $\mathcal{T}(m_{k+1}) \backslash \mathcal{F}(k) \backslash \{k+1\}$, $\pi_n^* = 0$ since $m_n = 0$. Amongst the nodes in $\mathcal{T}(m_{k+1}) \backslash \mathcal{F}(k)$, only node $k+1$ could have a positive dual value. For node $k+1$, (30) then holds from (29). On the other hand, $\sum_{m \in \mathcal{T}(m_{k+1})} \pi_m^* = c_{m_{k+1}}$ from (28), so $\{l \in \mathcal{T}(m_{k+1}) \backslash \mathcal{F}(k) : \sum_{m \in \mathcal{T}(l)} \pi_m^* < c_l\} \subseteq \mathcal{T}(m_{k+1}) \backslash \mathcal{F}(k) \backslash \{m_{k+1}\}$. The conclusion then holds since $x_m^* = 0$ for all $m \in \mathcal{T}(m_{k+1}) \backslash \mathcal{F}(k) \backslash \{m_{k+1}\}$. $\square$

It is easy to see that Algorithm 4 is just a refined procedure of Algorithm 2. Therefore, Algorithm 4 has the same complexity $O(N_T \log N_T \log\log N_T)$ as Algorithm 2.

3.5 Analysis of the approximation algorithm

This section analyzes the optimality gap of the approximate solution produced by Algorithm 3. Given capacity acquisition-allocation solutions (x, y), let us denote the corresponding objective function value as

$$f(x, y) = \sum_{n \in \mathcal{T}} p_n \Big(\alpha_n x_n + \beta_n y_n \Big).$$

Recall that (x^{LP}, y^{LP}) denotes the capacity acquisition-allocation solutions corresponding to the LP relaxation of (16), and (x^H, y^H) denotes the capacity acquisition-allocation solutions returned by Algorithm 3. Let (x^*, y^*) denote an optimal solution to (16). Then the optimality gap of (x^H, y^H) is

$$\text{GAP} = f(x^H, y^H) - f(x^*, y^*).$$

THEOREM 9 GAP $\leq \sum_{i=1}^{I} \alpha_{i1}$, *where* 1 *is the root node of the scenario tree.*

Theorem 9 shows the surprising result that the optimality gap of Algorithm 3 is bounded above by a factor that is independent of the number of time stages, number of branches in the tree, number of tasks, or any problem data except for the sum of the capacity acquisition costs of the resources in the *first stage*. At the first sight, this is against our intuition. However, through the following proof of Theorem 9, we can see that this is just a natural result of the approximation algorithm design and the property of simple stochastic lot-sizing problem. The basic observation is that we can produce a feasible solution of the original multi-stage integer program from the optimal solution of its linear relaxation. The feasible solution is just "slightly different" from the linear relaxation solution, which is determined by the approximation scheme. Note that in step 2 of Algorithm 3, we obtain x^H by solving simple stochastic lot-sizing problems with rounded up right hand sides compared to the linear relaxation problem. We can show that, comparing a simple stochastic lot-sizing problem with fractional right hand side and its variant with rounded up right hand side, the difference in the optimal objective values can be upper bounded by the cost of adding just one more unit in the root node.

PROOF. Note that

$$\begin{aligned}
\text{GAP} &\leq f(x^H, y^H) - f(x^{LP}, y^{LP}) \\
&= f(x^H, y^H) - f(x^H, y^{LP}) + f(x^H, y^{LP}) - f(x^{LP}, y^{LP}) \\
&\leq f(x^H, y^{LP}) - f(x^{LP}, y^{LP}),
\end{aligned}$$

where last inequality follows from the fact that $f(x^H, y^H) \leq f(x^H, y^{LP})$ (recall that y^H is an optimal capacity allocation corresponding to x^H, i.e., an optimal solution to capacity allocation problem solved in step 3 of Algorithm 3, whereas y^{LP} is just a feasible capacity allocation solution). Next

$$f(x^H, y^{LP}) - f(x^{LP}, y^{LP}) = \sum_{i=1}^{I} \sum_{n \in \mathcal{T}} p_n \alpha_{in} \Big(x^H_{in} - x^{LP}_{in} \Big) \tag{32}$$

corresponding to step 2 of Algorithm 3. Note that

$$\begin{aligned}
\textstyle\sum_{n\in\mathcal{T}} p_n \alpha_{in} x^H_{in} \quad &= \quad \min \quad \textstyle\sum_{n\in\mathcal{T}} p_n \alpha_{in} x_{in} \\
&\qquad\quad \text{s.t.} \quad \textstyle\sum_{m\in\mathcal{P}(n)} x_{im} \geq \lceil (A_n y^{LP}_n)_i \rceil \quad \forall\, n \in \mathcal{T} \\
&\qquad\qquad\quad\; x_{in} \in \mathbb{R}_+ \quad \forall\, n \in \mathcal{T} \\
&= \quad \max \quad \textstyle\sum_{n\in\mathcal{T}} \lceil (A_n y^{LP}_n)_i \rceil \pi_{in} \\
&\qquad\quad \text{s.t.} \quad \textstyle\sum_{m\in\mathcal{T}(n)} \pi_{im} \leq p_n \alpha_{in} \quad \forall\, n \in \mathcal{T} \\
&\qquad\qquad\quad\; \pi_{in} \in \mathbb{R}_+ \quad \forall\, n \in \mathcal{T},
\end{aligned} \tag{33}$$

and

$$\begin{aligned}
\textstyle\sum_{n\in\mathcal{T}} p_n \alpha_{in} x^{LP}_{in} \quad &= \quad \min \quad \textstyle\sum_{n\in\mathcal{T}} p_n \alpha_{in} x_{in} \\
&\qquad\quad \text{s.t.} \quad \textstyle\sum_{m\in\mathcal{P}(n)} x_{im} \geq (A_n y^{LP}_n)_i \quad \forall\, n \in \mathcal{T} \\
&\qquad\qquad\quad\; x_{in} \in \mathbb{R}_+ \quad \forall\, n \in \mathcal{T} \\
&= \quad \max \quad \textstyle\sum_{n\in\mathcal{T}} (A_n y^{LP}_n)_i \pi_{in} \\
&\qquad\quad \text{s.t.} \quad \textstyle\sum_{m\in\mathcal{T}(n)} \pi_{im} \leq p_n \alpha_{in} \quad \forall\, n \in \mathcal{T} \\
&\qquad\qquad\quad\; \pi_{in} \in \mathbb{R}_+ \quad \forall\, n \in \mathcal{T}.
\end{aligned} \tag{34}$$

Thus

$$\begin{aligned}
\textstyle\sum_{n\in\mathcal{T}} p_n \alpha_{in} \Big(x^H_{in} - x^{LP}_{in} \Big) \quad &= \quad \max \quad \textstyle\sum_{n\in\mathcal{T}} \Big(\lceil (A_n y^{LP}_n)_i \rceil - (A_n y^{LP}_n)_i \Big) \pi_{in} \\
&\qquad\quad \text{s.t.} \quad \textstyle\sum_{m\in\mathcal{T}(n)} \pi_{im} \leq p_n \alpha_{in} \quad \forall\, n \in \mathcal{T} \\
&\qquad\qquad\quad\; \pi_{in} \in \mathbb{R}_+ \quad \forall\, n \in \mathcal{T} \\
&\leq \quad \max \quad \textstyle\sum_{n\in\mathcal{T}} \pi_{in} \\
&\qquad\quad \text{s.t.} \quad \textstyle\sum_{m\in\mathcal{T}(n)} \pi_{im} \leq p_n \alpha_{in} \quad \forall\, n \in \mathcal{T} \\
&\qquad\qquad\quad\; \pi_{in} \in \mathbb{R}_+ \quad \forall\, n \in \mathcal{T} \\
&= \quad \min \quad \textstyle\sum_{n\in\mathcal{T}} p_n \alpha_{in} x_{in} \\
&\qquad\quad \text{s.t.} \quad \textstyle\sum_{m\in\mathcal{P}(n)} x_{im} \geq 1 \quad \forall\, n \in \mathcal{T} \\
&\qquad\qquad\quad\; x_{in} \in \mathbb{R}_+ \quad \forall\, n \in \mathcal{T} \\
&= \quad \alpha_{i1},
\end{aligned} \tag{35}$$

where the first equality follows from (33) and (34), the next inequality follows from the fact that $\lceil (A_n y_n^{LP})_i \rceil - (A_n y_n^{LP})_i \leq 1$, the next equality follows from duality, and the last equality follows from the fact that an optimal solution to a stochastic lot-sizing problem with a cumulative demand of 1 unit in *every* node is to produce 1 unit in the root node. The result then follows from incorporating (35) in (32). □

If we consider instances of (16) that have the same first-stage acquisition costs, but different topology of the scenario tree, then we have the following asymptotic quality guarantees for Algorithm 3. The following corollary is immediate.

COROLLARY 1 $\lim_{T\to\infty} \frac{f(x^H,y^H)-f(x^*,y^*)}{T} = 0.$

Under some very general positivity assumption, the relative gap between objective values of heuristic solution and the optimal solution will also vanish as the planning horizon goes to infinity.

COROLLARY 2 *In (16), let δ_{kn} be an item in $K \times 1$ vector δ_n; b_{kjn} be an item in $K \times J$ matrix B_n. Assume there exists $\epsilon_1 > 0$, $\epsilon_2 > 0$ such that for any node n, $\delta_n \neq 0$; and for any positive item $\delta_{kn} > 0$, we have $\delta_{kn} \geq \epsilon_1$; and for any nonzero item $b_{kjn} > 0$, we have $\frac{\beta_{kn}}{b_{kjn}} \geq \epsilon_2$. Then the following limit holds:*

$$\lim_{T\to\infty} \frac{f(x^H,y^H)-f(x^*,y^*)}{f(x^*,y^*)} = 0.$$

PROOF. We only need to show that $f(x^*,y^*) \to \infty$ as $n \to \infty$. Since $f(x^*,y^*) \geq f(x^{LP},y^{LP})$, we only need to show $f(x^{LP},y^{LP}) \to \infty$ as $n \to \infty$. For this purpose, we rewrite the linear relaxation of (16) as follows:

$$\begin{array}{lll}
\min & \sum_{n\in\mathcal{T}} p_n \Big[\alpha_n x_n + \beta_n y_n\Big] & \\
\text{s.t.} & \sum_{m\in\mathcal{P}(n)} x_m - A_n y_n \geq 0 & \forall\, n \in \mathcal{T} \\
 & B_n y_n \geq \delta_n & \forall\, n \in \mathcal{T} \\
 & y_n \in \mathbb{R}_+^J & \forall\, n \in \mathcal{T} \\
 & x_n \in \mathbb{R}_+^I & \forall\, n \in \mathcal{T},
\end{array}$$

whose dual program is:

$$\begin{array}{lll} \max & \sum_{n\in\mathcal{T}} \eta_n\delta_n & \\ \text{s.t.} & \sum_{m\in\mathcal{T}(n)} \gamma_n \le p_n\alpha_n & \forall\ n\in\mathcal{T} \\ & -\gamma_n A_n + \eta_n B_n \le p_n\beta_n & \forall\ n\in\mathcal{T} \\ & \gamma_n \in \mathbb{R}^I_+ & \forall\ n\in\mathcal{T} \\ & \eta_n \in \mathbb{R}^K_+ & \forall\ n\in\mathcal{T}, \end{array}$$

where $\alpha_n, \beta_n, \delta_n, \gamma_n, \eta_n$ are $1\times I, 1\times J, K\times 1, 1\times I, 1\times K$ vectors, respectively; and A_n, B_n are $I\times J$ and $K\times J$ matrices respectively. Let the objective function of the dual program be $g(\gamma_n,\eta_n)$. We will try to find a feasible solution $(\tilde{\gamma}_n,\tilde{\eta}_n)$ such that $g(\tilde{\gamma}_n,\tilde{\eta}_n) \ge T\epsilon$ for some $\epsilon > 0$. First, we assign $\tilde{\gamma}_n = 0$. Second, note that when $\delta_{k_1n} > 0$ for some k_1, δ_{k_1n} is the demand for some product or task. Then in (16), the corresponding row for this product or task in constraint $B_ny_n \ge \delta_n$ will be $\sum_{j=1}^J b_{k_1jn}y_{jn} \ge \delta_{k_1n} \ge \epsilon_1$. So we can find at least one j_1 such that $b_{k_1j_1n} > 0$, otherwise (16) is not feasible (we can always make the problem feasible by adding a very expensive artificial resource that can satisfy all the demands). Now we assign $\tilde{\eta}_{k_1n} = \frac{p_n\beta_{k_1n}}{b_{k_1j_1n}} \ge p_n\epsilon_2$ and $\tilde{\eta}_{kn} = 0$ for all $k \ne k_1$. For this solution $(\tilde{\gamma}_n,\tilde{\eta}_n)$, it is easy to see that $\sum_{n\in\mathcal{T}} \eta_n\delta_n \ge \sum_{n\in\mathcal{T}} p_n\epsilon_1\epsilon_2 = T\epsilon_1\epsilon_2$, where the last equality comes from the fact that $\sum_{n\in\mathcal{S}_t} p_n = 1$ for all $1 \le t \le T$. $\square$

Note the assumptions in Corollary 2 are very general. The condition $\delta_n \ne 0$ is to guarantee that there is always some positive demand (maybe for just one product); the condition $\delta_{kn} \ge \epsilon_1 > 0$ is to guarantee that the positive demand of a product or task is always no less than a small positive number; the condition $\frac{\beta_{kn}}{b_{kjn}} \ge \epsilon_2 > 0$ is to guarantee that no matter how to produce the product or how to realize the task by utilizing the resources, the unit production cost of one product or task will always be no less than a small positive number. All these conditions are clearly satisfied in any realistic setting.

3.6 Summary

In this chapter, we propose a generic multi-period capacity planning problem under uncertainty involving multiple resources, tasks and products.

First, we compare two-stage and multi-stage stochastic integer programming approaches for this problem. The concept of value of multi-stage stochastic programming (VMS) is discussed and informative analytical bounds are developed.

Second, by identifying and exploiting a key lot-sizing substructure in the problem, we propose an efficient approximation scheme for the difficult multi-stage model. We show that the absolute optimality gap of the approximation scheme is bounded above by a factor that is independent of the number of time stages, number of branches in the scenario tree, number of tasks, or any problem data except for the sum of the capacity acquisition costs of the resources in the first stage. This leads to an asymptotic optimality guarantee of the approximation scheme with respect to the number of planning stages.

CHAPTER IV

APPLICATION: SEMICONDUCTOR TOOL PLANNING

4.1 Introduction

Most existing computational studies of stochastic programming models in semiconductor tool planning are based on the two-stage approach (cf. [1, 5, 28, 54, 55, 56]), while few are based on the multi-stage approach. In [1], a multi-stage stochastic programming model for a very small exemplary problem is tested. This chapter describes a study of the multi-stage stochastic programming model proposed in Chapter III, with industrial-size data, for a semiconductor tool planning problem. The objective is two-fold. First, the performances of the two-stage and multi-stage approaches are compared; i.e., the *value of multi-stage stochastic programming*, with respect to different scenario trees and random demand patterns, is tested. Second, the solution quality of the approximation scheme proposed in Chapter III with respect to different scenario trees and random demand patterns is tested.

4.2 Semiconductor tool planning

Modern semiconductor manufacturing is a complex and capital-intensive procedure [26, 41] in which highly customized machine tools are needed for processing different products. Machine tools are very expensive; in a typical wafer fab, the cost of machine tools every year can range from several million dollars to tens of millions. Therefore, machine tool purchase planning is of great interest to operations research and has been studied intensively. However, this is a difficult problem due to the characteristics of the semiconductor industry.

The first difficulty derives from the fact that the capacity of semiconductor production lines highly depends on the product mix. To see this, one needs to understand the composition of production lines. Semiconductor manufacturing comprises a large number of processes, of which the major ones are: oxidation, where an insert layer of silicon dioxide is formed on the surface of a silicon wafer; photo-lithography, where the circuit image on a mask is transferred to the surface of the silicon wafer; etching, where the unwanted silicon dioxide layer is removed from the wafer surface; resist stripping, where the unwanted photoresist layer is removed from the wafer surface; ion implant, where the electrical conductivity of the silicon is selectively modified; and layering, where another layer of silicon is put on the wafer. Each major process may include a number of sub-processes. Furthermore, there may be multiple layers of circuits on a semiconductor chip, so a component will undergo these processes one or more times before it is finished. Combining these facts, one can see that the number of processes is very large. For each process, a set of machine tools is used. Each type of tool may be used for multiple processes. Therefore, the relationship between products, processes, and machine tools is very complicated; a detailed description can be found in [5]. Moreover, the capacity requirements of machine tools for different products can be very different. So it is not realistic to use an aggregate product to make machine tool purchase plans. One must consider the product mix and incorporate the complex relationship among products, processes and machine tools in the tool planning model. In other words, one cannot extract the strategic/tactical level decisions (machine tool purchase) from the operational level decisions (production arrangement).

The second difficulty derives from the lead time of machine tool manufacturing. Machine tools are highly customized; the lead time for a tool can be as long as one and half year. Therefore, to receive the appropriate machines at the time of manufacturing, one must make the order long before that. As mentioned above, the capacity requirement for machine tools highly depends on the product mix, which requires a good forecast of the product demand long before the exact manufacturing time. However, it is well known that semiconductor product demand is volatile and very hard to forecast. Therefore, a traditional deterministic planning model does not perform well in this case.

An optimization model is needed which incorporates uncertainty as well as a complex decision structure to model the relationship among products, processes and machine tools. For this, stochastic programming is an appropriate methodology. It has been shown that stochastic programming models can provide robust solutions to hedge against the uncertainty in semiconductor tool planning [1, 5, 28, 54, 55, 56]. In the following, we use the model in [5, 28] to test the development proposed in Chapter III.

4.3 Model and data

The original model of our test problem is a multi-period semiconductor tool planning problem from [5, 28]. The deterministic tool planning model we use is as follows:

$$
\begin{array}{lll}
\min & \sum_{p,t} U^+_{p,t} + q_1 \sum_{i\in PT} \sum_t N_{i,t} + q_2 \sum_{i\in NP} \sum_t N_{i,t} & \\
s.t. & \gamma_{p,t} W_{p,t} + U^+_{p,t} - U^-_{p,t} = d_{p,t} & \forall p,t \\
& \sum_p b_{j,p,t} W_{p,t} = \sum_{i\in I(j)} O_{j,i,t} & \forall j,t \\
& \mu_{i,0} + c_i \sum_{\tau=1}^{t} N_{i,\tau} - \sum_{j\in J(i)} h_{i,j,t} O_{i,j,t} \geq 0 & \forall i,t \\
& \sum_i m_{i,t} N_{i,t} \leq \beta_t & \forall t \\
& U^+_{p,t} \leq \alpha_p & \forall p,t \\
& W_{p,t}, U^+_{p,t}, U^-_{p,t}, O_{i,j,t} \in \mathbb{R}^+ & \forall i,j,p,t \\
& N_{i,t} \in \mathbb{Z}^+ & \forall i,t
\end{array}
\tag{36}
$$

The following notation is used to describe the model:

Indices:

i	:	index for semiconductor machine tools
j	:	index for manufacturing operations
p	:	index for products
t	:	index for planning periods

Variables:

$U^+_{p,t}$	:	unmet demand for product p in period t
$U^-_{p,t}$	:	excess production for product p in period t
$W_{p,t}$	:	quantity of wafers for product p in period t
$O_{j,i,t}$	:	quantity of wafers that pass through operation j, on tool type i in period t
$N_{i,t}$	:	number of machine tool type i in period t

Data:

$\gamma_{p,t}$	:	expected number of products completed per wafer for product p in period t
$d_{p,t}$	:	demand for product p in period t
$b_{j,p,t}$	:	number of passes of operation j on product p in period t
$\mu_{i,0}$	:	initial capacity for tool type i (measured in time)
c_i	:	additional unit capacity for tool type i (measured in time)
$h_{i,j,t}$	:	time to process one wafter through operation j on tool type i in period t
$m_{i,t}$	:	unit cost of tool type i in period t
α_p	:	upper bound of unmet demand for product p
β_t	:	maximum budget in period t
$I(j)$	:	collection of tool groups that can perform operation j
$J(i)$	:	collection of operations that can be performed on tool type i
PT	:	collection of primary tool types
NT	:	collection of non-primary tool types
q_1	:	cost related coefficient of primary tool types
q_2	:	cost related coefficient of non-primary tool types

In this formulation, the tools are divided into primal and non-primal tools, i.e., $I = PT \cup NP$, where PT is the set of primal tools and NP is the set of non-primal tools. Primal and non-primal tools have different weights in the objective function. The objective function is a sum of unmet customer demand $U^+_{p,t}$ and weighted number of tools. The first constraint relates the demand $d_{p,t}$ to production variable $W_{p,t}$, unmet demand $U^+_{p,t}$ and excess demand $U^-_{p,t}$. The second constraint relates the production of wafers and the manufacturing processes. The third constraint requires that the accumulated machine tool capacity can satisfy the production requirements. The fourth constraint is a budget constraint. A more detailed explanation of the objective function and constraints can be found in [5]. In this chapter, we extend this deterministic multi-period capacity planning model to two-stage and multi-stage stochastic programming models by introducing stochastic data description. The original data from [5, 28] is for a two stage, two period stochastic programming model. The data includes four scenarios. The authors in [5, 28] create a base scenario consisting of two periods. The base scenario is considered as having the highest probability to happen in the future. Based on this scenario, three other scenarios are developed corresponding to other possible outcomes of future demand evolvement with smaller probabilities. In the original data, there are 306 machine tools, 40 wafer types (products) and 2575 operations. To formulate one scenario, above 10000 variables and 6000 constraints are needed. For one period, there are above 5000 variables and 3000 constraints.

4.4 Experimental environment

In our numerical experiments, we made two modifications to the original model (36). First, we drop the budget constraint from the original model since this constraint will destroy the decomposable structure that is required by our approximation algorithm. Moreover, in the base scenario, this constraint is not tight. In other scenarios of original data, this constraint is violated only when the demands are significantly higher than the base case. Second, we consider discounted cost data, whereas the original data has the same cost for each period.

To generate the scenario tree for our stochastic program, we use the data of the first period of the base scenario from [5, 28] as the root node. Based on this node, all the nodes in future stages are generated by introducing random demand and discounted cost to the root node data. In each node, we have about 5000 variables and 3000 constraints.

For discounted cost data, according to [5, 28], we assume the length of one planning period is six months; and set the cost discounting factor for one planning period to 5%. For random demand data, we use lognormal distribution $\lambda(\mu, \sigma)$, where μ is the expectation and σ is the standard deviation. To describe the demand evolution along the planning horizon, we consider four demand distributions of combinations of different expectation and standard deviation (S.D.) trends as follows:

In Table 2, z_n is the demand of one single product in node n, and z_1 is the demand of the product in the root node. Note s_n is the stage number of node n (if $n \in \mathcal{S}_t$, then $s_n = t$). So for all the nodes in the same stage, we will have the same demand distribution.

Table 2: Demand distributions

Demand pattern	Characteristic	Distribution
1	constant mean, constant S.D.	$z_n \sim z_1\lambda(1, 0.5)$
2	constant mean, increasing S.D.	$z_n \sim z_1\lambda(1, 0.5 + 0.1s_n)$
3	increasing mean, constant S.D.	$z_n \sim z_1\lambda(1 + 0.5s_n, 0.5)$
4	increasing mean, increasing S.D.	$z_n \sim z_1\lambda(1 + 0.5s_n, 0.5 + 0.1s_n)$

In the numerical experiments, we consider scenario trees with the number of stages (T) varying from 2 to 5, and the number of branches (B) for each non-leaf node varying from 2 to 5. We get a total number of 7 scenario trees. The nodes in these trees vary from 3 to 31. The stochastic data is generated by Monte-Carlo sampling. For each data point, we conduct 5 independent experiments and use the average of 5 runs. Considering the different demand patterns, scenario trees, and repetitive numerical experiments, we generate a total number of 140 problems and solve them. All numerical experiments are conducted on an IBM PC with 1024 MB RAM and a PENTIUM4 1.6GHz processor.

4.5 Computational results

4.5.1 Comparison of two-stage and multi-stage models

To compare two-stage and multi-stage models, we read the original data and generate the scenario tree and stochastic programming formulations. In the following table, we compare the number of all variables(No. var.), integer variables (No. int. var.) and constraints (No. con.) for two-stage and multi-stage formulations. In the table, T is the number of stages in the scenario tree and N is the total number of nodes. Note the number of branches of non-leaf nodes is fixed at 2 for all the scenario trees generated in this section. From Table 3, we can see the number of integer variables of multi-stage problem grows much faster than the two-stage problem as the number of stages increases:

To compare two-stage and multi-stage model, we define the Relative Value of Multi-stage Stochastic Programming as:

$$RVMS = \frac{v^{TS} - v^{MS}}{v^{TS}},$$

Table 3: Comparison of two-stage and multi-stage formulations

T	N	Two-stage			Multi-stage		
		No. var.	No. int. var.	No. con.	No. var.	No. int. var.	No. con.
2	3	15315	612	8763	15621	918	8763
3	7	35225	918	20447	36449	2142	20447
4	15	74739	1224	43815	78105	4590	43815
5	31	153461	1530	90551	161417	9486	90551

where v^{TS}, v^{MS} are the optimal values of two-stage model and multi-stage model, respectively. However, since it is hard to solve the two-stage and multi-stage models to optimality, we only use a lower bound in our numerical experiments:

$$RVMS \geq \frac{v_{LP}^{TS} - v_{HR}^{MS}}{v_{LP}^{TS}},$$

where v_{LP}^{TS} is the optimal value of the linear relaxation of two-stage model; v_{HR}^{MS} is the heuristic solution objective value of the multi-stage model.

In the experiments, we first solve the linear relaxation of the two-stage model and get its objective value v_{LP}^{TS}. For the multi-stage formulation, we also solve the linear relaxation and get the objective value v_{LP}^{MS}. Based on the multi-stage LP relaxation solution, we apply the approximation scheme and obtain the heuristic solution and its objective value v_{HR}^{MS}. Note all the linear relaxation problems are solved using CPLEX9.0 primal simplex solver except the simple stochastic lot-sizing subproblem. We test all the four demand patterns. For each pattern, we draw the curve of $RVMS$ lower bound with respect to the total number of stages. In these experiments, we fix the number of branches for each non-leaf node to 2. The results are shown in Figure 5. Notice that each data point is obtained by getting the average value of 5 independent experiments.

Our first observation of Figure 5 is that for all the four demand patterns, the $RVMS$ lower bound increases as the number of stages increases. This implies that the value of multi-stage stochastic programming becomes higher when the planning horizon becomes longer, which agrees with our theoretical analysis. Recall the results for stochastic lot-sizing problem. We show that the value of multi-stage stochastic programming increases as the demand data variability increases under almost constant cost. When the number

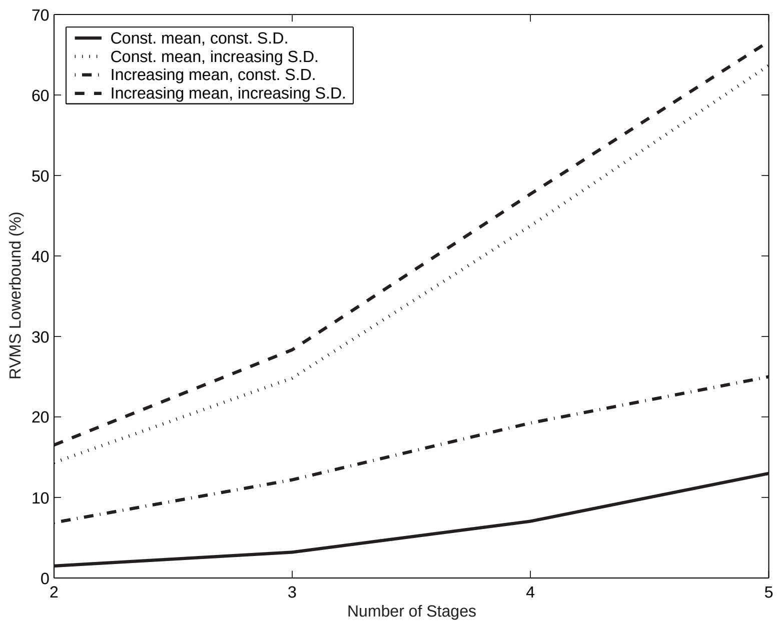

Figure 5: The value of multi-stage stochastic programming with increasing planning horizon

of stages is increasing, there is an increasing number of nodes in the tree. So it is more likely for a very high demand to appear when we generate random demand data. This very high demand will cause a large data variability, which leads to higher value of multi-stage stochastic programming.

The second observation is that the pattern of demand distribution has an influence on the magnitude of the $RVMS$ lower bounds. For demand pattern 1, the $RVMS$ lower bound is less than 13% for all the four scenario trees. For demand pattern 2, all the $RVMS$ lower bounds are larger than 14%; and the $RVSM$ becomes larger than 40% when the number of stages is larger than 3. For demand pattern 3, the largest $RVMS$ lower bound is less than 25%. But for demand pattern 4, the $RVMS$ lower bounds are above 28% when there are larger than 2 stages; and the lower bounds are above 45% when there are 4 and 5 stages. Also, the growth of $RVMS$ lower bounds with respect to the number of stages for demand pattern 2 and 4 is faster than that for demand pattern 1 and 3. These differences are caused by the setting of standard deviation. In demand pattern 2 and 4, the standard deviation is increasing, while the standard deviation is constant in demand pattern 1 and 3. Since the increasing standard deviation will bring larger variability in the demand data,

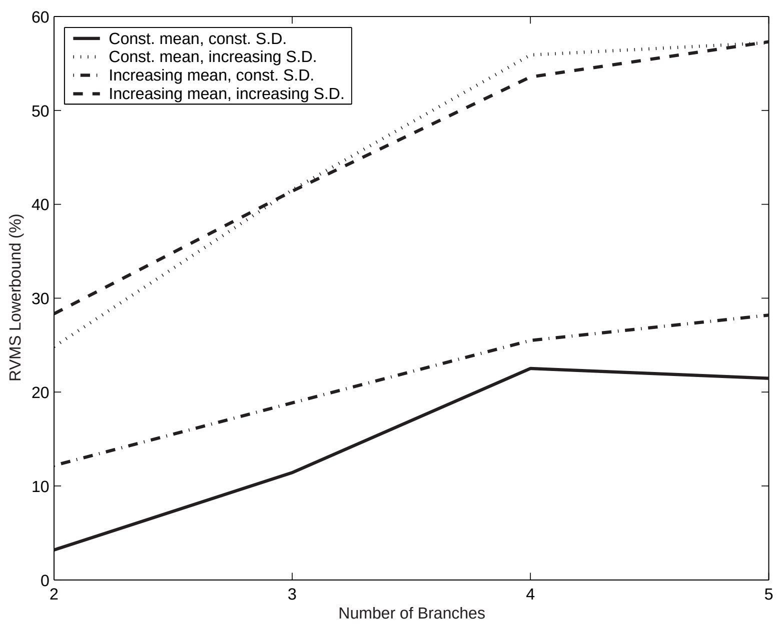

Figure 6: The value of multi-stage stochastic programming with increasing number of branches

so the value of multi-stage stochastic programming will increase as the standard deviation increases, which again conforms to our theoretical results.

In all the previous experiments, we fix the number of branches for non-leaf nodes to 2, and increase the number of stages. In fact, we can also increase the scale of the scenario tree by increasing the number of branches of non-leaf nodes. We would like to test the performance of the stochastic programming models with respect to an increasing number of branches. For this purpose, we fix the number of stages of scenario tree to 3, and increase the number of branches for all the non-leaf nodes from 2 to 5. For a total number of 4 scenario trees, we run the numerical experiments and draw the curves of $RVMS$ lower bound against the number of branches. The results are shown in Figure 6.

In Figure 6, we can see that the $RVMS$ lower bound is always increasing as the number of branches increases, except that it dropped slightly under demand pattern 1 when the number of branches increases from 4 to 5. This observation can be explained by our analysis on VMS. When there are more branches, the scale of the scenario tree increases, i.e., there are more nodes in the tree. This implies there is a higher probability for a very high demand to appear. Therefore, on average, the VMS will increase as the number of nodes increases. Especially, when the standard deviation is larger, the VMS tends to be larger. This is exactly what we find in Figure 6. We note that the VMS lower bounds for demand pattern 2 and 4 are much higher than that for demand pattern 1 and 3. This is caused by using increasing standard deviation in demand pattern 2 and 4, compared with the constant standard deviation in demand pattern 1 and 3.

4.5.2 The quality of heuristic solutions

For all the test problems in our experiment, the heuristic solution can be obtained within 90 seconds. So the efficiency of the approximation scheme is satisfactory. To study the solution quality, we define the relative gap between optimal objectives of heuristic solution and optimal solution of multi-stage model as:

$$RGAP = \frac{v_{HR}^{MS} - v^{MS}}{v^{MS}}.$$

Since it is difficult to solve the multi-stage model, we use an upper bound for $RGAP$:

$$RGAP \leq \frac{v_{HR}^{MS} - v_{LP}^{MS}}{v_{LP}^{MS}},$$

where v_{LP}^{MS} is the optimal value of the linear relaxation of multi-stage model. We draw the curves of $RGAP$ upper bounds against the number of stages in the scenario tree. Note in all these curves we fix the number of branches of non-leaf nodes to 2. The results are shown in Figure 7.

In Figure 7, we first notice that all the curves are deceasing as the number of stages increases, except that the $RGAP$ increases slightly in demand patten 1 when the number of stages increases from 4 to 5. This observation is compatible with our theoretical analysis. Recall that the gap between the heuristic solution objective value and the optimal objective value of the multi-stage model is bounded by the summation of all the resource costs in the first stage (for our numerical experiment, it is the summation of all the tool costs in the first stage), which is a constant number with respect to the increasing number of stages. When the number of stages increase 1, we need to satisfy the added demands from the new stage. So the optimal value of multi-stage model will increase on average. In other words,

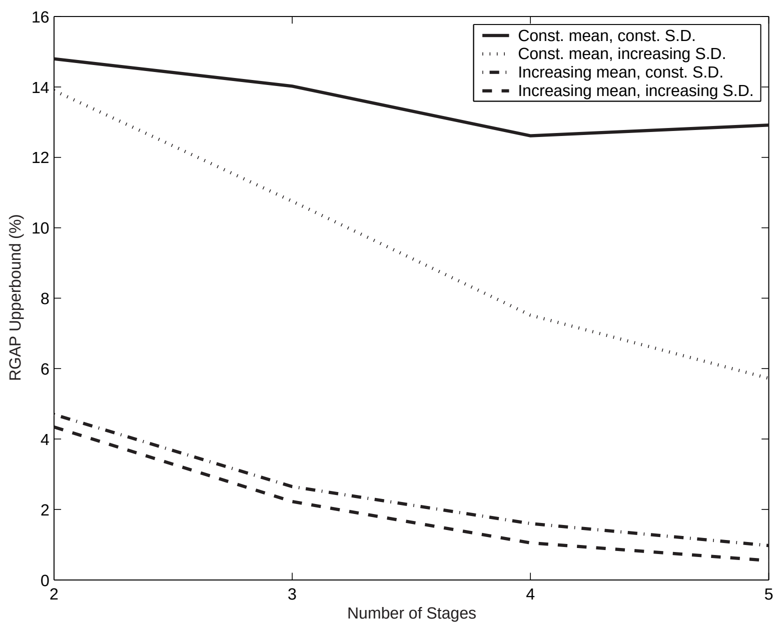

Figure 7: The heuristic solution quality of multi-stage stochastic programming with increasing planning horizon

on an average basis the optimal objective value of the multi-stage model will increase as the number of stages increases. Therefore, on an average basis the *RGAP* value will decrease as the number of stages increases.

The other observation of the curves is that when the mean demand is increasing, the quality of heuristic solution is better. For demand pattern 1, the heuristic solution quality is not satisfactory. The lowest *RGAP* upper bound for all four scenarios trees is larger than 14%. For demand pattern 2, *RGAP* lower bound is smaller compared to demand pattern 1. However, the lowest *RGAP* value is still above 5%. Note demand pattern 1 and demand pattern 2 both have distributions with constant mean. For the demand patterns with increasing mean demand, i.e., demand pattern 3 and 4, the *RGAP* upper bound is obviously smaller than that of demand pattern 1 and 2 for all the scenario trees. This phenomenon is due to the increasing demand expectation, which will lead to faster increase of the optimal objective value of the multi-stage model. With the same reasoning as in the first observation, the *RGAP* will then tend to be smaller.

As in the last subsection, we are also interested in the heuristic solution quality when

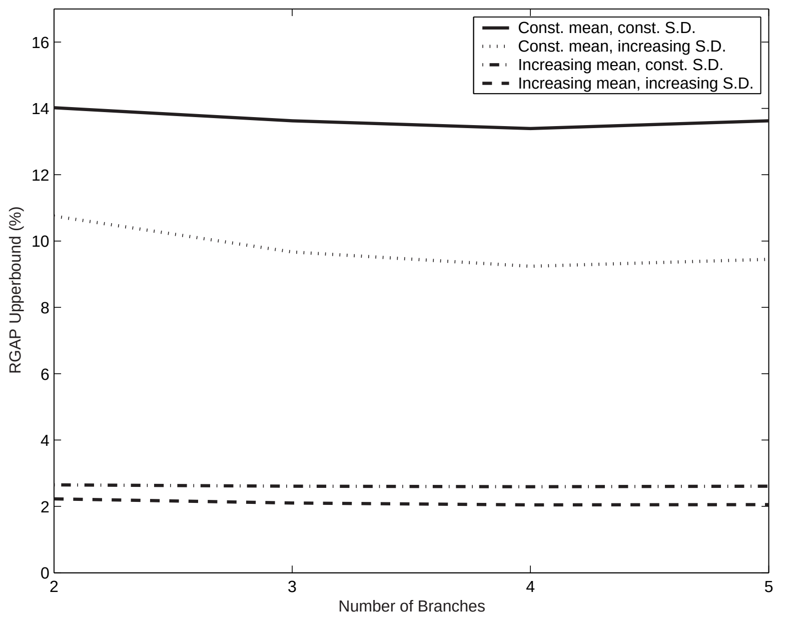

Figure 8: The heuristic solution quality of multi-stage stochastic programming with increasing number of branches

we fix the number of stages and increase the number of branches of non-leaf nodes. For this purpose, we test 4 scenario trees with the number of stages fixed at 3, and the number of branches of non-leaf nodes increasing from 2 to 5. We draw the curves of $RGAP$ upper bound against the number of branches. The results are shown in Figure 8.

In Figure 8, we notice that all the curves are almost level. To explain this, we can use a similar reasoning as before. Since the number of stages is fixed for all the scenario trees, on average the optimal value of multi-stage model is almost constant. Therefore, increasing the number of branches will not affect $RGAP$ very much. However, the influence of different distributions is still noticeable. The $RGAP$ upper bound for demand pattern 3 and 4 is much smaller than that of demand pattern 1 and 4. The reason is that demand pattern 3 and 4 use increasing mean while demand pattern 1 and 2 use constant mean.

Finally, compared with the running times of two-stage LP and multi-stage LP, the heuristic for multi-stage model always consumes the least amount of time. Among all the instances, the heuristic running time is always less than 2 minutes. As a comparison, we use the CPLEX9.0 MIP solver to solve an instance with 3 nodes($T = 2, B = 2$) in the scenario tree, both the two-stage model and multi-stage model take more than 1 hour to solve. In another instance with 7 nodes($T = 3, B = 2$) in the scenario tree, both the two-stage model and multi-stage model cannot be solved after 2 hours. These results show that a good heuristic is very necessary for this category of stochastic capacity planning problems; and the heuristic proposed in this paper can provide satisfactory solutions in reasonable amount of time.

4.6 Summary

In this chapter, we conduct a large-scale computational study, using realistic-scale problem instances corresponding to semiconductor tool planning. Numerical results indicate that a lower bound on the relative VMS can be as high as 70%. Recall that this lower bound is obtained by comparing the cost of an approximate solution to the multi-stage model to that of a lower bound on the cost of an optimal solution of the two-stage model. Therefore, this suggests that even an approximate solution to the multi-stage model may be far superior to any optimal solution to the two-stage model. These results confirm that the VMS for these problems is quite high. Moreover, the quality and performance of the approximation scheme is very satisfactory, more so for cases where the VMS is high.

CHAPTER V

THE PLANNING HORIZON OF AN INFINITE HORIZON STOCHASTIC LOT-SIZING PROBLEM

5.1 Introduction

A critical consideration in developing a multi-period production planning model is the length of the planning horizon [20, 43, 60]. On one hand, the planning horizon should be long enough to avoid myopic solutions; on the other hand, an excessively long horizon requires an excessively large input data set which may render the problem intractable. In the case where the planning horizon is very long or indefinite, a practical approach is to use a finite horizon approximation, also called the rolling horizon approach. In this approach, a planner solves the finite approximation problem and implements the solution for only the first few periods, solving a new approximation problem with the same length of horizon when in the next period. Theoretically, it is important to know the conditions under which the finite horizon approximation problem generates a good approximate solution with respect to the optimal solution of the original infinite horizon problem (cf. [11]). Under certain conditions, one may be able to find a length of the planning horizon such that the finite horizon approximation problem will have an optimal first-period solution the same as that of the original infinite horizon problem. This length is called the planning horizon. In the field of stochastic dynamic programming, there have been extensive studies for the planning horizon problem. In [50, 51], a class of general infinite horizon discounted-cost optimization problems is studied. Some sufficient conditions and necessary conditions are given for the convergence of optimal solutions of finite horizon approximation problems to the optimal solution of the original infinite horizon problem. In [6, 7, 8, 12], some general infinite horizon optimization problems are studied and conditions for the existence of a planning horizon are given. In [53], the authors consider a type of deterministic lot-sizing problem with convex production and inventory costs, by using the monotonicity of the finite horizon approximation problem. It is shown that under mild conditions there exists a planning horizon. In an extension of this work, the authors in [17] study the stochastic case and show that the planning horizon still exists under a linear cost structure and other conditions.

However, all past studies of the planning horizon problem remain within a Markovian decision analysis context. In other words, only feasible solutions or policies with the Markovian property are considered. There exists very few studies of the planning horizon problem within the stochastic programming context. In [34], a general approximation framework of infinite horizon optimization problems based on stochastic programming is developed. The research discusses approximation techniques for a general class of stationary infinite horizon problems with discounted costs. In this chapter, a specific infinite horizon stochastic lot-sizing model under the framework of stochastic programming is established. It is assumed that the stochastic processes are discrete and can be represented by an infinite scenario tree. Both cost and demand are random parameters and the underlying stochastic processes are general. By studying the structure of optimal solutions of the finite horizon approximation problem, it is shown that the optimal solution of the finite horizon approximation problem has a non-decreasing property as the planning horizon increases. Moreover, it is shown that when the demand and cost data are properly bounded, the finite horizon approximation problem converges to the original infinite horizon problem, both in optimal objective value and in solution. Therefore, one can show the existence of a planning horizon under the framework of stochastic programming. This result applies to non-stationary, arbitrary random demand and cost parameters. One can also develop a useful formula to determine a lower bound of the planning horizon.

5.2 The stochastic programming formulation

In this study, we consider a single item, uncapacitated, infinite horizon stochastic lot-sizing problem in which the stochastic data is described by a scenario tree. The notations are as follows:

Scenario tree:

$\mathcal{T}^\infty$	:	the set of all nodes in the infinite horizon scenario tree
$\mathcal{P}(n)$	:	the set of nodes on the path from node n to the root node in $\mathcal{T}^\infty$
$a(n)$	:	the direct ancestor node of node n
$t(n)$	:	the stage of node n, i.e., the cardinality of $\mathcal{P}(n)$
$\mathcal{T}^T$	:	the subtree composed of all nodes n with $t(n)$ no greater than T
$\mathcal{S}_t$	:	the set of all nodes n with $t(n) = t$, which is called the tth stage
B	:	the largest number of children of a node in $\mathcal{T}^\infty$ (note $B \geq 2$)

Variables and input:

δ_n	=	the demand at node n of the scenario tree ($\delta_n \in \mathbb{Z}_+$)
q_n	=	the discounting factor at node $n (0 < \mu < 1)$
I_0	=	the inventory at the present time (we usually assume $I_0 = 0$)
α_n	=	the unit production cost at node n
β_n	=	the unit inventory holding cost at node n
x_n	=	production level at node n
I_n	=	inventory level at node n

With these notations, the infinite horizon stochastic lot-sizing problem and its finite approximation, denoted by $(\mathcal{Q}^\infty)$ and $(\mathcal{Q}^T)$, respectively, are as follows:

$$
\begin{aligned}
(\mathcal{Q}^\infty): \quad \min \quad & \textstyle\sum_{n \in \mathcal{T}^\infty} q_n[\alpha_n x_n + \beta_n I_n] & \\
\text{s.t.} \quad & I_{a(n)} + x_n = I_n + \delta_n \qquad & n \in \mathcal{T}^\infty \\
& x_n, I_n \in \mathbb{Z}_+ & n \in \mathcal{T}^\infty
\end{aligned}
\tag{37}
$$

$$
\begin{array}{llll}
(\mathcal{Q}^T): & \min & \sum_{n\in\mathcal{T}^T} q_n[\alpha_n x_n + \beta_n I_n] & \\
& \text{s.t.} & I_{a(n)} + x_n = I_n + \delta_n & n \in \mathcal{T}^T \\
& & x_n, I_n \in \mathbb{Z}_+ & n \in \mathcal{T}^T
\end{array} \tag{38}
$$

To denote the objective values and solutions of $(\mathcal{Q}^\infty)$, we have the following notations:

X^∞	:	a feasible solution of $(\mathcal{Q}^\infty)$
Π^∞	:	the set of all feasible solutions of $(\mathcal{Q}^\infty)$
$F(X^\infty)$	:	the objective value of solution X^∞
$X^{\infty*}$	:	the optimal solution of $(\mathcal{Q}^\infty)$
$F^{\infty*}$	:	the optimal objective value of $(\mathcal{Q}^\infty)$

Note X^∞, $X^{\infty*}$ are vectors in $(\mathbb{Z}_+^2)^\infty$. For example, $X^\infty = \{X_n^\infty\}_{n\in\mathcal{T}^\infty}$ where $X_n^\infty = (x_n^\infty, I_n^\infty)$ for all n in $\mathcal{T}^\infty$. The corresponding notations for finite horizon approximation problem are X^T, Π^T, $F(X^T)$, X^{T*} and $F(X^{T*})$, respectively. We treat X^T and X^{T*} as vectors in $(\mathbb{Z}_+^2)^\infty$ such that all the entries corresponding to nodes after the Tth stage are zeroes, i.e., $X_n^T = (0,0)$ for all $n \in \mathcal{T}^\infty \backslash \mathcal{T}^T$. We also define $\Pi = \Pi^\infty \cup (\cup_{t=1}^\infty \Pi^t)$, which is called the solution space. An arbitrary element in Π is denoted as $X = \{X_n\}_{n\in\mathcal{T}^\infty}$ and $X_n = (x_n, I_n)$ for all $n \in \mathcal{T}^\infty$. The corresponding objective value is:

$$
F(X) = \sum_{n\in\mathcal{T}^\infty} q_n[\alpha_n x_n + \beta_n I_n] \tag{39}
$$

5.3 Value and optimal solution convergence

5.3.1 Assumptions

Before we discuss the convergence properties of the finite horizon approximation problem, we adopt the following assumptions:

$(A1)$ The discounting factors of nodes at each stage is uniformly bounded by an exponential function of the stage number: There exists $\epsilon_2 > 0$ and $0 < \mu < 1$ such that for all $n \in \mathcal{T}^\infty$,

$$
0 < q_n \le \epsilon_2 \mu^{t(n)}. \tag{40}
$$

$(A2)$ The demands of nodes in each stage is uniformly bounded by an exponential function of the stage number: There exists $\epsilon_3 > 0$ and $\nu > 1$ such that for all $n \in \mathcal{T}^\infty$,

$$0 < \delta_n \leq \epsilon_3 \nu^{t(n)}. \tag{41}$$

$(A3)$ The unit production and inventory costs are strictly positive and uniformly bounded. There exists $M_2 \geq M_1 > 0$ such that for all $n \in \mathcal{T}^\infty$:

$$M_1 \leq \alpha_n, \beta_n \leq M_2. \tag{42}$$

$(A4)$ We assume

$$\mu\nu B < 1. \tag{43}$$

Notice that assumption $(A1)$ is easy to satisfy. One sufficient condition is as follows. Let $q_n = p_n r^{t(n)}$, where p_n is the probability of node n and r is a uniform discounting factor $(0 < r < 1)$. If we assume that each node has exactly B children and all the nodes in the same stage have the same probabilities (in other words, the probability distribution in each stage is uniform). Then we have $p_n = \frac{B}{B^{t(n)}}$. In this case, it is easy to see that $\mu = \frac{r}{B}$ and $\epsilon_2 = B$. Furthermore, we can also assume that the probability distribution satisfies that $p_n \leq \frac{\epsilon_2}{B^{t(n)}}$. Assumption $(A2)$ is also not a very strong assumption. It only requires the increasing demand be bounded by an exponential distribution. Assumption $(A3)$ is easy to satisfy for costs in real life. The last assumption $(A4)$ is very important for the existence of optimal solution, which can be seen in the following developments. The intuition behind assumption $(A4)$ is that the decreasing rate μ of discounting factors has to be small enough, so that it can have a larger influence on the objective value than that of the increasing rate ν of demand and increasing number of nodes (whose rate is controlled by the largest number of branches B).

From the above assumptions, we can get some very useful properties of the problem under study. First, note that for any sequence of demands $\{\delta_n\}_{n \in \mathcal{T}^\infty}$, $X_n^T = (\delta_n, 0)$ for all $n \in \mathcal{T}^T$ constitutes a feasible solution X^T for $(\mathcal{Q}^T)$, and $X_n^\infty = (\delta_n, 0)$ for all $n \in \mathcal{T}^\infty$

constitutes a feasible solution X^∞ for $(\mathcal{Q}^\infty)$. For this solution, the objective value:

$$\begin{aligned} F_1 = \sum_{n\in\mathcal{T}^\infty} q_n[\alpha_n\delta_n + \beta_n 0] &\leq \sum_{n\in\mathcal{T}^\infty} \epsilon_2\mu^{t(n)} M_2\epsilon_3\nu^{t(n)} \\ &\leq \sum_{t=1}^{\infty} \epsilon_2\epsilon_3 M_2 B^{-1}(\mu\nu B)^t \\ &< \infty \end{aligned}$$

Note the second inequality comes from the fact that $\mathcal{T}^\infty = \cup_{t=1}^\infty \mathcal{S}_t$, $|\mathcal{S}_t| \leq B^{t-1}$ and $t(n) = t$ for all $n \in S_t$. The last inequality is guaranteed by assumption $(A4)$. Therefore, without loss of generality, we can define the space Π^∞ as the set of all the feasible solutions of $(\mathcal{Q}^\infty)$ whose total cost is no greater than F_1. Therefore, the objective value is finite for each solution in Π.

Second, it is easy to see that for all the feasible solutions $X = \{(x_n, I_n)\}_{n\in\mathcal{T}^\infty}$ in Π and any fixed node $n \in \mathcal{T}^\infty$, we can find uniform finite upper bounds for x_n and I_n. Otherwise the maximum objective value of feasible solutions will go to infinity by assumption $(A3)$. Combining this fact with the integrality of x_n and I_n, we conclude that for each fixed n, there are only *finitely many* possible values of x_n and I_n in any feasible solution. In fact, x_n and I_n can be uniformly bounded by an exponential function of $t(n)$. Indeed, assume that there is an unit produced in node n that will satisfy the demand at stage $t(n) + K$. Then fix n, according to assumption $(A1)$, we can further assume that $\epsilon_1\mu^{t(n)} < q_n$ for some $0 < \epsilon_1 < \epsilon_2$. Combining this assumption and $(A3)$, the inventory cost is at least $\sum_{i=0}^{K-1} M_1\epsilon_1\mu^{t(n)+i}$. Now consider the alternative to produce this unit at stage $t(n) + K$ directly so there is no inventory cost. Then the total cost is at most $M_2\epsilon_2\mu^{t(n)+K}$. Clearly, $\sum_{i=0}^{K-1} M_1\epsilon_1\mu^i$ converges to a positive number as K goes to infinity, while $M_2\epsilon_2\mu^K$ goes to zero. Therefore, we can find a positive integer K_1 such that $\sum_{i=0}^{K-1} M_1\epsilon_1\mu^i > M\epsilon_2\mu^K$ for all $K > K_1$, i.e., we would never produce an unit to satisfy the demand of a node more than K_1 stages from the current stage. Therefore, at one node n, the maximum number of units we would produce is:

$$\begin{aligned} & \sum_{i=0}^{K_1} B^i \epsilon_3 \nu^{t(n)+i} \\ = \;& \epsilon_3 \nu^{t(n)} \frac{1-(B\nu)^{K_1+1}}{1-B\nu} \\ = \;& \epsilon_4 \nu^{t(n)}, \end{aligned}$$

where $\epsilon_4 = \epsilon_3 \frac{1-(B\nu)^{K_1+1}}{1-B\nu}$ is a positive number. We also notice that the maximum number of units that can be carried as inventory at node n is also bounded by $\epsilon_4 \nu^{t(n)}$. Therefore, we conclude that for any optimal solution we have $x_n, I_n \leq \epsilon_4 \nu^{t(n)}$. We can further define the feasible solutions as $X = \{(x_n, I_n)\}_{n \in \mathcal{T}^\infty}$ such that $F(X) \leq F_1$ and $x_n, I_n \leq \epsilon_4 \nu^{t(n)}$.

5.3.2 Existence of optimal solution

To study the existence of optimal solution of the infinite horizon problem, we need to discuss the topology of the solution space first. By comments of the last section, there are only finite possible choices of values for any entry in a feasible solution. So we can define a metric $\rho(\cdot,\cdot)$ on the solution space such that $\rho(X, X') = \sum_{n \in \mathcal{T}^\infty} 2^{-n} \phi(X_n, X'_n)$ where:

$$\phi(X_n, X'_n) = \begin{cases} 0: & X_n = X'_n \\ 1: & o.w. \end{cases}$$

Note $\phi(\cdot,\cdot)$ itself corresponds to a metric defined on $\mathbb{R}^2$.

THEOREM 10 *ρ is a metric and Π is compact with respect to this metric.*

PROOF. The first half of the statement is clear from the definition of metric [15]. Clearly, $\rho(X, X) = 0$ for all $X \in \Pi$; $\rho(X, X') = \rho(X', X) \geq 0$ for all $(X, X') \in \Pi \times \Pi$; and if $\rho(X, X') = 0$ then $X = X'$. For all $X, Y, Z \in \Pi$, we have $\rho(X, Z) = \sum_{n \in \mathcal{T}^\infty} 2^{-n} \phi(X_n, Z_n) \leq \sum_{n \in \mathcal{T}^\infty} 2^{-n} [\phi(X_n, Y_n) + \phi(Y_n, Z_n)] = \rho(X, Y) + \rho(Y, Z)$. So the triangle inequality also holds. To prove the compactness of Π, we only need to show Π is totally bounded and complete. We first show that Π is totally bounded. Note that for any $\epsilon > 0$, we can find n_1 such that $2^{-n_1} < \epsilon$. Consider two solutions X and X'. If $X_n = X'_n$ for all $n \leq n_1$, then we have $\rho(X, X') \leq \sum_{n=n_1+1}^{\infty} 2^{-n} = 2^{-n_1} < \epsilon$. On the other hand, recall that for a fixed

n, there are only *finitely many* possible values of x_n and I_n in any feasible solution. So we can find a finite set of feasible solutions such that any solution $X \in \Pi$ will have the same values in the first n_1 entries as one of the solutions in this set. Combining these results, it is not hard to see that we can construct a finite cover of Π consisting of cells of radius ϵ. Therefore, Π is totally bounded. Now let $\{X^i\}_{i=1}^{\infty}$ be a Cauchy sequence in Π. Clearly, for any fixed n, $\{X_n^i\}_{i=1}^{\infty}$ is also a Cauchy sequence with respect to the metric implied by $\phi(\cdot,\cdot)$. Therefore, we can find $\widehat{X}_n$ such that $X_n^i = \widehat{X}_n$ when i is large enough by the definition of ϕ. Define $\widehat{X} = (\widehat{X}_1, \widehat{X}_2, \ldots)$. It is easy to see that $\lim_{i\to\infty} X^i = \widehat{X}$. We only need to show that $\widehat{X}$ is feasible, which is guaranteed by the feasibility of $\{X^i\}_{i=1}^{\infty}$ and the fact that $X_n^i = \widehat{X}_n$ when i is large enough. This proves the completeness of Π. $\square$

Next, we show the continuity of $F(X)$ with respect to X.

THEOREM 11 *Let $\{X^i\}_{i=1}^{\infty}$ be a sequence in Π and $\lim_{i\to\infty} X^i = X$ where $X \in \Pi$. Then $\lim_{i\to\infty} F(X^i) = F(X)$.*

PROOF. From the proof of last theorem, we know that for any fixed T, we can find $K(T)$ such that for all $i > K(T)$ and $1 \le n \le |\mathcal{T}^T|$, $X_n^i = X_n$. Therefore, for $i > K(T)$:

$$\begin{aligned} & |F(X) - F(X^i)| \\ = & \left| \sum_{n\in\mathcal{T}^\infty\setminus\mathcal{T}^T} q_n[\alpha_n(x_n - x_n^i) + \beta_n(I_n - I_n^i)]\right| \\ \le & \sum_{n\in\mathcal{T}^\infty\setminus\mathcal{T}^T} \epsilon_2 \mu^{t(n)}[\alpha_n|x_n - x_n^i| + \beta_n|I_n - I_n^i|] \\ \le & \sum_{n\in\mathcal{T}^\infty\setminus\mathcal{T}^T} 4\epsilon_2\mu^{t(n)} M_2 \epsilon_4 \nu^{t(n)} \end{aligned}$$

Note $\sum_{n\in\mathcal{T}^\infty\setminus\mathcal{T}^T} 4\epsilon_2\epsilon_4 M_2 \mu^{t(n)}\nu^{t(n)} \to 0$ when $T \to \infty$ by assumption $(A4)$. $\square$

Finally, we can show the existence of optimal solution of the infinite horizon problem:

COROLLARY 1 *An optimal solution of the infinite horizon problem $(\mathcal{Q}^\infty)$ exists, i.e., $\exists X^{\infty *}$ such that $F^{\infty *} = F(X^{\infty *}) < \infty$.*

PROOF. According to Theorem 10 and Theorem 11, Π is compact and $F(X)$ is continuous in X. Therefore, the image of Π under F is also compact. $\square$

5.3.3 Convergence results

First, we show a certain type of monotonicity of an optimal solution of the finite horizon problem.

LEMMA 3 *Let $X^{T*} = \{(x_n^{T*}, I_n^{T*})\}_{n\in\mathcal{T}^\infty}$ (note $x_n^{T*} = I_n^{T*} = 0$ for all $n \in \mathcal{T}^\infty \backslash \mathcal{T}^T$) be any optimal solution of $(\mathcal{Q}^T)$ with respect to a vector of demands $D^T = (\delta_n)_{n\in\mathcal{T}^T}$. If the demand at one leaf node of $\mathcal{T}^T$ increases one unit, then there is a new optimal solution such that the production in the root node of scenario tree is nondecreasing.*

PROOF. According to Chapter II, we can reformulate $(\mathcal{Q}^T)$ as follows:

$$\begin{array}{llll} (\mathcal{Q}^T): & \min & \sum_{n\in\mathcal{T}^T} c_n x_n & \\ & s.t. & \sum_{m\in\mathcal{P}(n)} x_m \geq d_n & n\in\mathcal{T}^T \\ & & x_n \in \mathbb{R}_+ & n\in\mathcal{T}^T, \end{array} \tag{44}$$

where $d_n = \sum_{m\in\mathcal{P}(n)} \delta_m$ and $c_n = q_n(\alpha_n + \frac{\sum_{m\in\mathcal{T}^T(n)} q_m\beta_m}{q_n})$. Let $(x_n^*)_{n\in\mathcal{T}^T}$, $(\pi_n^*)_{n\in\mathcal{T}^T}$ be the optimal primal and dual solutions to (44), respectively. Clearly, $(x_n^*)_{n\in\mathcal{T}^T}$ is primal feasible, while $(\pi_n^*)_{n\in\mathcal{T}^T}$ is dual feasible. The complementary slackness is satisfied:

$$x_n^* > 0 \implies \sum_{m\in\mathcal{T}^T(n)} \pi_m^* = c_n, \tag{45}$$

$$\sum_{m\in\mathcal{P}(n)} x_m^* > d_n \implies \pi_n^* = 0. \tag{46}$$

Assume $(d'_n)_{n\in\mathcal{T}^T}$ is a new vector of demands such that $d'_n = d_n$ for all $n \in \mathcal{T}^T \backslash \{n_1\}$ and $d'_{n_1} = d_{n_1} + 1$, where n_1 is a leaf node of $\mathcal{T}^T$ ($n_1 \in \mathcal{S}_T$). Note node 1 is the root node and $T^T(n)$ denotes the subtree in $\mathcal{T}^T$ with root $n \in \mathcal{T}^T$. To prove the result, we need to show there exists an optimal solution $(x'_n)_{n\in\mathcal{T}^T}$ for $(d'_n)_{n\in\mathcal{T}^T}$ such that $x'_1 \geq x_1^*$.

First, if $\sum_{m\in\mathcal{P}(n_1)} x_m^* \geq d_{n_1} + 1$, then $(x_n^*)_{n\in\mathcal{T}^T}$ and $(\pi_n^*)_{n\in\mathcal{T}^T}$ are still feasible and conform to complementary slackness. So they are still optimal for the demand sequence

$(d'_n)_{n\in\mathcal{T}^T}$. Second, if $\sum_{m\in\mathcal{P}(n_1)} x^*_m = d_{n_1}$ and $x^*_{n_1} > 0$, then we let $x'_n = x^*_n$ for all $n \in \mathcal{T}^T\backslash\{n_1\}$ and $x'_{n_1} = x^*_{n_1} + 1$. Clearly, $(x'_n)_{n\in\mathcal{T}^T}$ is primal feasible. It is also easy to check $(x'_n)_{n\in\mathcal{T}^T}$, $(\pi^*_n)_{n\in\mathcal{T}^T}$ still satisfy complementary slackness. Therefore, they are still optimal for the demands $(d'_n)_{n\in\mathcal{T}^T}$. Third, if $\sum_{m\in\mathcal{P}(n_1)} x^*_m = d_{n_1}$ and $x^*_{n_1} = 0$, we consider the nearest ancestor node n_2 of n_1 which has positive solution in $(x^*_n)_{n\in\mathcal{T}^T}$, i.e., $x^*_{n_2} > 0$ and $x^*_n = 0$ for all $n \in \mathcal{P}(n_1)\backslash\mathcal{P}(n_2)$. Then we let $x'_n = x^*_n$ and $\pi'_n = \pi^*_n$ for all $n \in \mathcal{T}^T\backslash\mathcal{T}^T(n_2)$. Now consider the problem with respect to scenario tree $\mathcal{T}^T(n_2)$ and $((d_n - \sum_{m\in\mathcal{P}(n_2)\backslash\{n_2\}} x^*_m)^+)_{n\in\mathcal{T}^T(n_2)}$ (where $(\cdot)^+ = \max\{\cdot, 0\}$). Assume we can find a pair of primal feasible and dual feasible solutions $(x'_n)_{n\in\mathcal{T}^T(n_2)}$, $(\pi'_n)_{n\in\mathcal{T}^T(n_2)}$ in $T^T(n_2)$ such that $x^{'}_{n_2} \geq x^*_{n_2} > 0$ and the complementary slackness is satisfied, then it is easy to check that $(x'_n)_{n\in\mathcal{T}^T}$ and $(\pi'_n)_{n\in\mathcal{T}^T}$ are optimal. Therefore, without loss of generality, we can assume that $n_2 = 1$, that is, we assume that $x^*_1 > 0$, $x^*_n = 0$ for all $n \in \mathcal{P}(n_1)\backslash\{1\}$ and $x^*_1 = d_{n_1}$. We only need to show that we could find a pair of optimal solutions $(x'_n)_{n\in\mathcal{T}^T}$, $(\pi'_n)_{n\in\mathcal{T}^T}$ for $(d'_n)_{n\in\mathcal{T}^T}$ such that $x'_1 \geq x^*_1 > 0$.

To facilitate our proof, we define the following notations:

$$\begin{aligned}
\mathcal{A}^*_+(n) &= \{m \in \mathcal{T}^T(n)\backslash\{n\} : x^*_m > 0, \quad x^*_i = 0 \quad \forall i \in \mathcal{P}(m)\backslash\{m\}\backslash\mathcal{P}(n)\} \\
\mathcal{A}^*_{>1}(n) &= \{m \in \mathcal{T}^T(n)\backslash\{n\} : x^*_m > 1, \quad x^*_i = 0 \quad \forall i \in \mathcal{P}(m)\backslash\{m\}\backslash\mathcal{P}(n)\} \\
\mathcal{A}^*_{=1}(n) &= \{m \in \mathcal{T}^T(n)\backslash\{n\} : x^*_m = 1, \quad x^*_i = 0 \quad \forall i \in \mathcal{P}(m)\backslash\{m\}\backslash\mathcal{P}(n)\} \\
\mathcal{A}^*_=(n) &= \{m \in \mathcal{T}^T(n) : \textstyle\sum_{i\in\mathcal{P}(n)} x^*_i = d_m, \quad x^*_i = 0 \quad \forall i \in \mathcal{P}(m)\backslash\mathcal{P}(n)\} \\
\mathcal{A}^*_>(n) &= \{m \in \mathcal{T}^T(n) : \textstyle\sum_{i\in\mathcal{P}(n)} x^*_i > d_m, \quad x^*_i = 0 \quad \forall i \in \mathcal{P}(m)\backslash\mathcal{P}(n)\}
\end{aligned}$$

Now let $x_n = x^*_n$ for all $n \in \mathcal{T}^T\backslash\{n_1\}$ and $x_{n_1} = 1$, which constitutes a primal feasible solution for $(d'_n)_{n\in\mathcal{T}^T}$. Then starting from this solution, we use the shifting up primal algorithm proposed in Chapter II (page 11). We claim that when the algorithm ends, we get a solution $(x'_n)_{n\in\mathcal{T}^T}$ which is optimal. The basic idea is to construct a dual feasible solution $(\pi'_n)_{n\in\mathcal{T}^T}$ from $(\pi^*_n)_{n\in\mathcal{T}^T}$ that satisfies the complementary slackness with respect to $(x'_n)_{n\in\mathcal{T}^T}$.

Case one: The shifting up algorithm ends at node n_1, i.e., we have $x'_n = x^*_n$ for all $n \in \mathcal{T}^T\backslash\{n_1\}$, $x'_{n_1} = 1$ and for all $n \in \mathcal{P}(n_1)\backslash\{n_1\}$:

$$c_n \geq c_{n_1} + \sum_{m\in\mathcal{A}^*_+(n)} c_m \tag{47}$$

In the original dual solution, we have $\sum_{m\in\mathcal{T}^T(n)} \pi^*_m = c_n$ for all $n \in \mathcal{A}^*_+(1)$ and $\pi^*_n = 0$ for all $n \in \mathcal{A}^*_>(1)$ by complementary slackness. Notice $\mathcal{T}^T = (\cup_{m\in\mathcal{A}^*_+(1)}\mathcal{T}^T(m))\cup\mathcal{A}^*_>(1)\cup\mathcal{A}^*_=(1)$. We let $\pi'_n = \pi^*_n$ for all $n \in (\cup_{m\in\mathcal{A}^*_+(1)}\mathcal{T}^T(m)) \cup \mathcal{A}^*_>(1)$; $\pi'_n = \pi^*_n - \theta_n$ (where $0 \leq \theta_n \leq \pi^*_n$) for all $n \in \mathcal{A}^*_=(n)\backslash\{n_1\}$; and $\pi'_{n_1} = c_{n_1}$. We also define $\theta_{n_1} = c_{n_1} - \pi^*_{n_1}$. Note θ_n is unknown right now for all $n \in \mathcal{A}^*_=(n)\backslash\{n_1\}$, and θ_{n_1} is known. We need to show this set of dual solution $(\pi'_n)_{n\in\mathcal{T}^T}$ is dual feasible and satisfies complementary slackness with respect to $(x'_n)_{n\in\mathcal{T}^T}$.

To check dual feasibility, note that for all $n \neq n_1$, we have $\pi'_n \leq \pi^*_n$. Only $\pi'_{n_1} > \pi^*_{n_1}$. Therefore, $\sum_{m\in\mathcal{T}^T(n)} \pi'_m \leq \sum_{m\in\mathcal{T}^T(n)} \pi^*_m \leq c_n$ for all $n \in \mathcal{T}^T$ except perhaps for $n \in \mathcal{P}(n_1)$. For complementary slackness (45) and (46), it is also easy to see that we only need to check $n \in \mathcal{P}(n_1)$. Note for n_1, we have $\pi'_{n_1} = c_{n_1}$ (so the complementary slackness and feasibility constraint for node n_1 are satisfied). Combining these observations, we need to check for $n \in \mathcal{P}(n_1)\backslash\{n_1\}$:

$$\begin{aligned} c_n &\geq c_{n_1} + \textstyle\sum_{m\in\mathcal{A}^*_+(n)} c_m + \sum_{m\in\mathcal{A}^*_=(n)\backslash\{n_1\}}(\pi^*_m - \theta_m) \\ &= \theta_{n_1} + \pi^*_{n_1} + \textstyle\sum_{m\in\mathcal{A}^*_+(n)} c_m + \sum_{m\in\mathcal{A}^*_=(n)\backslash\{n_1\}}(\pi^*_m - \theta_m) \quad \forall n \in \mathcal{P}(n_1)\backslash\{n_1\}\backslash\{1\}, \\ c_1 &= c_{n_1} + \textstyle\sum_{m\in\mathcal{A}^*_+(1)} c_m + \sum_{m\in\mathcal{A}^*_=(1)\backslash\{n_1\}}(\pi^*_m - \theta_m) \\ &= \theta_{n_1} + \pi^*_{n_1} + \textstyle\sum_{m\in\mathcal{A}^*_+(1)} c_m + \sum_{m\in\mathcal{A}^*_=(1)\backslash\{n_1\}}(\pi^*_m - \theta_m) \quad n = 1. \end{aligned}$$

On the other hand, also by feasibility and complimentary slackness, in the original dual solution $(\pi^*_n)_{n\in\mathcal{T}^T}$:

$$\begin{aligned} \triangle_n &= c_n - \pi^*_{n_1} - \textstyle\sum_{m\in\mathcal{A}^*_+(n)} c_m - \sum_{m\in\mathcal{A}^*_=(n)\backslash\{n_1\}} \pi^*_m \quad \forall n \in \mathcal{P}(n_1)\backslash\{n_1\}\backslash\{1\}, \\ c_1 &= \pi^*_{n_1} + \textstyle\sum_{m\in\mathcal{A}^*_+(1)} c_m + \sum_{m\in\mathcal{A}^*_=(1)\backslash\{n_1\}} \pi^*_m \qquad n = 1, \end{aligned}$$

where $\theta_{n_1} = c_n - \pi^*_{n_1} \geq \triangle_n \geq 0$ by dual feasibility. So we require the existence of a feasible solution to the following linear system:

$$\begin{aligned} \triangle_n &\geq \theta_{n_1} - \textstyle\sum_{m\in\mathcal{A}^*_=(n)\backslash\{n_1\}} \theta_m \quad \forall n \in \mathcal{P}(n_1)\backslash\{n_1\}\backslash\{1\} \\ 0 &= \theta_{n_1} - \textstyle\sum_{m\in\mathcal{A}^*_=(1)\backslash\{n_1\}} \theta_m \end{aligned} \tag{48}$$

where the unknowns are θ_m for all $m \in \mathcal{A}^*_=(1)\backslash\{n_1\}$ (θ_{n_1} is known). Note now (47) becomes:

$$\begin{array}{lcll} \triangle_n + \sum_{m\in\mathcal{A}^*_=(n)\backslash\{n_1\}} \pi^*_m & \geq & \theta_{n_1} & \forall n \in \mathcal{P}(n_1)\backslash\{n_1\}\backslash\{1\} \\ \sum_{m\in\mathcal{A}^*_=(1)\backslash\{n_1\}} \pi^*_m & \geq & \theta_{n_1} & n = 1 \end{array} \tag{49}$$

Note $\mathcal{A}^*_=(n) \subseteq \mathcal{A}^*_=(a(n))$ for all $n \in \mathcal{P}(n_1)\backslash\{n_1\}\backslash\{1\}$. Therefore, according to (49), (48) is obviously solvable, i.e., there exists a set of θ_m for all $m \in \mathcal{A}^*_=(1)\backslash\{n_1\}$ such that $0 \leq \theta_m \leq \pi^*_m$ and (48) is satisfied.

Case two: The shifting up algorithm ends at node $n_3 \in \mathcal{P}(n_1)\backslash\{n_1\}$. In this case, $x'_n = x^*_n$ for all $n \in \mathcal{T}^T\backslash\mathcal{T}^T(n_3)$; $x'_{n_3} = x^*_{n_3} + 1$; $x'_n = x^*_n - 1$ for all $n \in \mathcal{A}^*_+(n_3)$; $x'_n = x^*_n$ for all $n \in \cup_{m\in\mathcal{A}^*_+(n_3)}(\mathcal{T}^T(m)\backslash\{m\})$; and $x'_n = x^*_n = 0$ for all $n \in \{m \in \mathcal{T}^T(n_3)\backslash\{n_3\} :$ $x^*_i = 0 \quad \forall i \in \mathcal{P}(m)\backslash\mathcal{P}(n_3)\}$. Notice if $n_3 = 1$, then $x^*_{n_3} > 0$; otherwise $x^*_{n_3} = 0$. Also note that $\mathcal{A}^*_+(n_3) = \mathcal{A}^*_{>1}(n_3) \cup \mathcal{A}^*_{=1}(n_3)$. So $x'_n > 0$ for all $n \in \mathcal{A}^*_{>1}(n_3)$ and $x'_n = 0$ for all $n \in \mathcal{A}^*_{=1}(n_3)$.

According to our shifting up algorithm, we will have the following two types of inequalities. One type of inequalities comes from the fact that the one unit production is shifted up from node n_1 to n_3:

$$c_{a(n)} < c_n + \sum_{m\in\mathcal{A}^*_+(a(n))\backslash\mathcal{T}^T(n)} c_m \quad \forall n \in \mathcal{P}(n_1)\backslash\mathcal{P}(n_3)$$

which implies

$$c_{n_3} < c_n + \sum_{m\in\mathcal{A}^*_+(n_3)\backslash\mathcal{T}^T(n)} c_m \quad \forall n \in \mathcal{P}(n_1)\backslash\mathcal{P}(n_3) \tag{50}$$

To see this, just add the inequalities along the path from n to n_3. The other type of inequalities comes from the fact that this one unit can not be shifted up from n_3 to one of its ancestors any more:

$$c_n \geq c_{n_3} + \sum_{m\in\mathcal{A}^*_+(n)\backslash\mathcal{T}^T(n_3)} c_m \quad \forall n \in \mathcal{P}(n_3)\backslash\{n_3\} \tag{51}$$

Note if $n_3 = 1$, then (51) does not exist. In the original dual solution, we have $\sum_{m\in\mathcal{T}^T(n)} \pi^*_m = c_n$ for all $n \in \mathcal{A}^*_+(1)$ and $\pi^*_n = 0$ for all $n \in \mathcal{A}^*_>(1)$ by complementary slackness. To construct a new dual solution, first we let $\pi'_n = \pi^*_n$ for all $n \in \cup_{m\in\mathcal{A}^*_+(1)}\mathcal{T}^T(m)$ and $n \in \mathcal{A}^*_>(1)$. Second, let $\pi'_n = 0$ for all $n \in \mathcal{A}^*_=(1) \cap \mathcal{T}^T(n_3)\backslash\{n_1\}$ and $\pi'_n = \pi^*_n - \theta_n$

(where $0 \leq \theta_n \leq \pi_n^*$) for all $n \in \mathcal{A}_=^*(1) \backslash \mathcal{T}^T(n_3)$. Third, let $\pi'_{n_1} = c_{n_3} - \sum_{m \in \mathcal{A}_+^*(n_3)} c_m$. We also define $\theta_{n_1} = c_{n_3} - \sum_{m \in \mathcal{A}_+^*(n_3)} c_m - \pi_{n_1}^* - \sum_{m \in \mathcal{A}_=^*(n_3) \backslash \{n_1\}} \pi_m^* = c_{n_3} - \sum_{m \in \mathcal{T}^T(n_3)} \pi_m^*$. By dual feasibility, we have $\sum_{m \in \mathcal{T}^T(n_3)} \pi_m^* = \sum_{m \in \mathcal{A}_+^*(n_3)} c_m + \pi_{n_1}^* + \sum_{m \in \mathcal{A}_=^*(n_3) \backslash \{n_1\}} \pi_m^* \leq c_{n_3}$. So $\theta_{n_1} \geq 0$. Note θ_{n_1} is known and θ_n is unknown for all $n \in \mathcal{A}_=^*(1) \backslash \mathcal{T}^T(n_3)$.

We need to show this set of dual solution $(\pi'_n)_{n \in \mathcal{T}^T}$ is feasible and satisfies complementary slackness with respect to $(x'_n)_{n \in \mathcal{T}^T}$. It is easy to see that for all $n \notin \mathcal{P}(n_1)$, we have $\sum_{m \in \mathcal{T}^T(n)} \pi'_m \leq \sum_{m \in \mathcal{T}^T(n)} \pi_m^* \leq c_n$; and $x'_n > 0$ implies $\sum_{m \in \mathcal{T}^T(n)} \pi'_m = c_n$; and $\sum_{m \in \mathcal{P}(n)} x'_m > d_n$ implies $\pi'_n = 0$. Therefore, we only need to check nodes $n \in \mathcal{P}(n_1)$. For all nodes in $n \in \mathcal{P}(n_1) \backslash \mathcal{P}(n_3)$, for dual feasibility we need:

$$\begin{aligned} c_n &\geq \pi'_{n_1} + \textstyle\sum_{m \in \mathcal{A}_+^*(n)} c_m \\ &= c_{n_3} - \textstyle\sum_{m \in \mathcal{A}_+^*(n_3)} c_m + \textstyle\sum_{m \in \mathcal{A}_+^*(n)} c_m \\ &= c_{n_3} - \textstyle\sum_{m \in \mathcal{A}_+^*(n_3) \backslash \mathcal{T}^T(n)} c_m, \end{aligned}$$

which is guaranteed by (50) already. For complimentary slackness at node n_3 ($x'_{n_3} > 0$), we require that

$$c_{n_3} = \pi'_{n_1} + \textstyle\sum_{m \in \mathcal{A}_+^*(n_3)} c_m,$$

which is guaranteed by the definition of π'_{n_1}. For dual feasibility at node $n \in \mathcal{P}(n_3) \backslash \{n_3\} \backslash \{1\}$, we require:

$$c_n \geq c_{n_3} + \textstyle\sum_{m \in \mathcal{A}_+^*(n) \backslash \mathcal{T}^T(n_3)} c_m + \textstyle\sum_{m \in \mathcal{A}_=^*(n) \backslash \mathcal{T}^T(n_3)} (\pi_m^* - \theta_m),$$

and for feasibility and complimentary slackness at node 1:

$$c_1 = c_{n_3} + \textstyle\sum_{m \in \mathcal{A}_+^*(1) \backslash \mathcal{T}^T(n_3)} c_m + \textstyle\sum_{m \in \mathcal{A}_=^*(1) \backslash \mathcal{T}^T(n_3)} (\pi_m^* - \theta_m).$$

Using the notation θ_{n_1}, the above two conditions become:

$$\begin{aligned} c_n &\geq \theta_{n_1} + \textstyle\sum_{m \in \mathcal{T}^T(n_3)} \pi_m^* + \textstyle\sum_{m \in \mathcal{A}_+^*(n) \backslash \mathcal{T}^T(n_3)} c_m \\ &\quad + \textstyle\sum_{m \in \mathcal{A}_=^*(n) \backslash \mathcal{T}^T(n_3)} (\pi_m^* - \theta_m) \qquad \forall n \in \mathcal{P}(n_3) \backslash \{n_3\} \backslash \{1\}, \\ c_1 &= \theta_{n_1} + \textstyle\sum_{m \in \mathcal{T}^T(n_3)} \pi_m^* + \textstyle\sum_{m \in \mathcal{A}_+^*(1) \backslash \mathcal{T}^T(n_3)} c_m \\ &\quad + \textstyle\sum_{m \in \mathcal{A}_=^*(1) \backslash \mathcal{T}^T(n_3)} (\pi_m^* - \theta_m) \qquad n = 1. \end{aligned}$$

On the other hand, in the original dual solution:

$$\begin{aligned} c_n &= \triangle_n + \textstyle\sum_{m \in \mathcal{T}^T(n_3)} \pi_m^* + \sum_{m \in \mathcal{A}_+^*(n) \backslash \mathcal{T}^T(n_3)} c_m \\ &\quad + \textstyle\sum_{m \in \mathcal{A}_=^*(n) \backslash \mathcal{T}^T(n_3)} \pi_m^* \qquad \forall n \in \mathcal{P}(n_3) \backslash \{n_3\} \backslash \{1\}, \\ c_1 &= \textstyle\sum_{m \in \mathcal{T}^T(n_3)} \pi_m^* + \sum_{m \in \mathcal{A}_+^*(1) \backslash \mathcal{T}^T(n_3)} c_m \\ &\quad + \textstyle\sum_{m \in \mathcal{A}_=^*(1) \backslash \mathcal{T}^T(n_3)} \pi_m^* \qquad n = 1, \end{aligned}$$

where $\triangle_n \geq 0$ by dual feasibility. So we require a solution of the following linear system where the unknowns are θ_m for all $m \in \mathcal{A}_=^*(1) \backslash \mathcal{T}^T(n_3)$ (θ_{n_1} is known):

$$\begin{aligned} \triangle_n &\geq \theta_{n_1} - \textstyle\sum_{m \in \mathcal{A}_=^*(n) \backslash \mathcal{T}^T(n_3)} \theta_m \quad \forall n \in \mathcal{P}(n_3) \backslash \{n_3\} \backslash \{1\} \\ 0 &= \theta_{n_1} - \textstyle\sum_{m \in \mathcal{A}_=^*(1) \backslash \mathcal{T}^T(n_3)} \theta_m \quad n = 1. \end{aligned} \tag{52}$$

Notice now (51) becomes:

$$\begin{aligned} \triangle_n + \textstyle\sum_{m \in \mathcal{A}_=^*(n) \backslash \mathcal{T}^T(n_3)} \pi_m^* &\geq \theta_{n_1} \quad \forall n \in \mathcal{P}(n_3) \backslash \{n_3\} \backslash \{1\} \\ \textstyle\sum_{m \in \mathcal{A}_=^*(1) \backslash \mathcal{T}^T(n_3)} \pi_m^* &\geq \theta_{n_1} \quad n = 1. \end{aligned} \tag{53}$$

Therefore, (52) are obviously solvable, i.e., there exists a set of θ_m for all $m \in \mathcal{A}_=^*(1) \backslash \mathcal{T}^T(n_3)$ such that $0 \leq \delta_m \leq \pi_m^*$ and linear system (52) is satisfied. $\square$

According to this lemma, we have the following corollary:

COROLLARY 2 *x_1^{T*} is monotonically increasing in T.*

PROOF. Consider the $T+1$ horizon problem with demand vector $D^{T+1} = (\delta_n)_{n \in \mathcal{T}^{T+1}}$. We can obtain the optimal solution of this $T+1$ horizon problem by the following way: Solve the T horizon problem first and set $x_n^{T+1} = x_n^{T*}$ for all $n \in \mathcal{T}^T$ and $x_n^{T+1} = 0$ for all $n \in \mathcal{T}^{T+1} \backslash \mathcal{T}^T$. Notice that this solution $(x_n^{T+1})_{n \in \mathcal{T}^{T+1}}$ is an optimal solution for the $T+1$ horizon problem with demand vector $(\tilde{\delta}_n)_{n \in \mathcal{T}^{T+1}}$ such that $\tilde{\delta}_n = \delta_n$ for all $n \in \mathcal{T}^T$ and $\tilde{\delta}_n = 0$ for all $n \in \mathcal{T}^{T+1} \backslash \mathcal{T}^T$. According to Lemma 3, if we increase the demand $\tilde{\delta}_n$ by one unit for some $n \in \mathcal{T}^{T+1} \backslash \mathcal{T}^T$, i.e., $\tilde{\delta}_n \leftarrow \tilde{\delta}_n + 1$, then we can find a new optimal solution in which the first stage production x_1^{T+1} will be nondecreasing. Repeating this procedure until $\tilde{\delta}_n = \delta_n$ for all $n \in \mathcal{T}^{T+1} \backslash \mathcal{T}^T$, then $(x_n^{T+1})_{n \in \mathcal{T}^{T+1}}$ will be exactly an optimal solution of the $T+1$ horizon problem; and we have $x_1^{T+1} \geq x_1^{T*}$. $\square$

Next, we will show that for any fixed $n \in \mathcal{T}^\infty$, the solution x_n^{T*} is bounded for all T.

LEMMA 4 *For any fixed $n \in \mathcal{T}^\infty$, there exists $\tilde{x}_n^* \in \mathbb{Z}_+$ such that $x_n^{T*} \le \tilde{x}_n^*$ for any T.*

PROOF. Assume this conclusion is not true for node n, then there exists sequence $\{T_k^n\}_k$ such that $x_n^{(T_k^n)*} \to \infty$ as $k \to \infty$. By our assumptions, we have $F^{(T_k^n)*} \to \infty$, which is a contradiction. □

According to Corollary 2 and Lemma 4, it can be shown that x_n^{T*} converges as T goes to infinity by Monotone Convergence Theorem.

COROLLARY 3 *For any fixed n, there exists $\bar{x}_n$ such that $x_n^{T*} \to \bar{x}_n$ as $T \to \infty$.*

PROOF. First, consider set $\mathcal{S}_1$. There is only one node 1 in this set and it is the root of the whole scenario tree. By Corollary 2 and Lemma 4 and monotone convergence theorem, there exists $\bar{x}_1$ such that $\lim_{T\to\infty} x_1^{T*} = \bar{x}_1$. Also by integrality of x_1, $x_1^{T*} = \bar{x}_1$ when T is large enough. Under such circumstance, we can consider the nodes in $\mathcal{S}_2$, i.e., the children of node 1. Now since the values of decision variables of node 1 are already fixed, the original problem becomes $|\mathcal{S}_2|$ independent problems whose roots are exactly the nodes in $\mathcal{S}_2$. We can use the former arguments again to show that there exists $\bar{x}_n$ such that $\lim_{T\to\infty} x_n^{T*} = \bar{x}_n$ for all $n \in \mathcal{S}_2$. Therefore, by induction, our conclusion holds for nodes in $\mathcal{S}_T N$ for any $T \ge 1$. □

It is not hard to show the vector $(\bar{x}_n)_{n\in\mathcal{T}^\infty}$ is a feasible solution for the infinite horizon problem (note that when $(x_n)_{n\in\mathcal{T}^\infty}$ is determined, $(I_n)_{n\in\mathcal{T}^\infty}$ is also determined by the constraints of $\mathcal{Q}^\infty$).

LEMMA 5 *$(\bar{x}_n)_{n\in\mathcal{T}^\infty}$ is a feasible solution for $(\mathcal{Q}^\infty)$.*

PROOF. For any fixed $n \in \mathcal{T}^\infty$, we have:

$$\begin{aligned}\textstyle\sum_{m\in\mathcal{P}(n)} \bar{x}_m = \sum_{m\in\mathcal{P}(n)} \lim_{T\to\infty} x_n^{T*} &= \textstyle\lim_{T\to\infty} \sum_{m\in\mathcal{P}(n)} x_n^{T*} \\ &\ge \textstyle\lim_{T\to\infty}(\sum_{m\in\mathcal{P}(n)} \delta_m - I_0) = \sum_{m\in\mathcal{P}(n)} \delta_m - I_0,\end{aligned}$$

where the first inequality comes from the feasibility of $(x_n^{T*})_{n\in\mathcal{T}^\infty}$. □

Finally, we obtain the following convergence result.

THEOREM 12 *(Solution and value convergence property)*

$$\lim_{T\to\infty} F^{T*} = \bar{F} = F^{\infty*}, \tag{54}$$

where $\bar{F}$ is the objective value corresponding to the solution $(\bar{x}_n)_{n\in\mathcal{T}^\infty}$.

PROOF. First, let $\bar{X}^\infty$ be the feasible solution corresponding to $(\bar{x}_n)_{n\in\mathcal{T}^\infty}$. It is easy to see that $X^{T*} \to \bar{X}^\infty$ as $T \to \infty$ by Corollary 3 and the integrality of decision variables. Therefore, according to Theorem 11, we have:

$$\lim_{T\to\infty} F^{T*} = \bar{F}.$$

On the other hand, we have the following inequalities:

$$F^{T*} \le F^{\infty*} \le \bar{F}.$$

The first inequality comes from the fact that we can obtain a feasible solution X^T of $(\mathcal{Q}^T)$ from any optimal solution $X^{\infty*}$ of $(\mathcal{Q}^\infty)$ by setting $X_n^T = X_n^{\infty*}$ for all $n \in \mathcal{T}^T$, and the objective value of this solution is an upper bound of F^{T*}. The second inequality comes from Lemma 5. Therefore, we have $\lim_{T\to\infty} F^{T*} = \bar{F} = F^{\infty*}$. $\square$

5.4 Determination of the planning horizon

Theorem 12 implies the existence of a planning horizon of (Q^∞), i.e., we can find a length T^1 of planning horizon such that for all $T \ge T^1$, the optimal solution x_1^{T*} at the root node of the finite approximation problem (Q^T) will be equal to $x_1^{\infty*}$ of the infinite horizon problem (Q^∞). In fact, we can develop some useful upper bound for this length of planning horizon.

By Lemma 3, $x_1^{(T+1)*} \ge x_1^{T*}$. Consider the case $x_1^{(T+1)*} > x_1^{T*}$, it implies that it will cost less to increase one more unit production at the root node 1 than to produce this unit at the $T+1$ stage. Since the production cost and inventory cost are linear, we only need to consider the following $T+1$ stage problem. Let $\delta'_n = 0$ for all $n \in \mathcal{T}^T$; and let $\delta'_n = 1$ for all $n \in \mathcal{S}_{T+1}$. Compare two solutions for this set of data. In the first solution, $x_1 = 1$

and $x_n = 0$ for all other n. In the second solution, $x'_n = 0$ for all $n \in \mathcal{T}^T$ and $x'_n = 1$ for all $n \in \mathcal{S}_{T+1}$. The objective values of these two solutions are:

$$\begin{aligned} F &= q_1\alpha_1 + \textstyle\sum_{n\in\mathcal{T}^T} q_n\beta_n \\ F' &= \textstyle\sum_{n\in\mathcal{S}_{T+1}} q_n\alpha_n \end{aligned}$$

If $F \geq F'$, then it means we could not decrease the objective value by produce one more unit at the root node to satisfy the demands T stages away from the first stage. In this case we can conclude that $T^1 \leq T$. To find a good upper bound for T^1, we would like to find the smallest T such that

$$q_1\alpha_1 + \textstyle\sum_{n\in\mathcal{T}^T} q_n\beta_n \geq \sum_{n\in\mathcal{S}_{T+1}} q_n\alpha_n \tag{55}$$

holds. For this purpose, let $q^{l^t}\beta^{l^t} = \min_{n\in\mathcal{S}_t} q_n\beta_n$ for $t = 1,\cdots,T$ and $q^{u^{T+1}}\alpha^{u^{T+1}} = \max_{n\in\mathcal{S}_{T+1}} q_n\alpha_n$. We would require:

$$q_1\alpha_1 + \textstyle\sum_{t=1}^T q^{l^t}\beta^{l^t}|\mathcal{S}_t| \geq q^{u^{T+1}}\alpha^{u^{T+1}}|\mathcal{S}_{T+1}|.$$

If we assume each node in $\mathcal{T}^\infty$ has exactly B children, then $|\mathcal{S}_t| = B^{t-1}$ for $t = 1,\cdots,T+1$. We have:

$$T_1 \leq \min\{T : q_1\alpha_1 + \textstyle\sum_{t=1}^T q^{l^t}\beta^{l^t}B^{t-1} \geq q^{u^{T+1}}\alpha^{u^{T+1}}B^T\} \tag{56}$$

Under further assumptions, we can even get closed form representation of T^1 upper bound. For example, if we use assumptions $(A1)$ and $(A3)$ and assume that

$$\epsilon_1\mu^{t(n)} \leq q_n \tag{57}$$

for some $\epsilon_2 > \epsilon_1 > 0$ (note that ϵ_1 is uniform for all n) , then we have $T_1 \leq \min\{T : q_1\alpha_1 + \sum_{t=1}^T \epsilon_1\mu^t M_1 B^{t-1} \geq \epsilon_2\mu^{T+1}M_2B^T\}$, from which we get

$$T_1 \leq \lceil \exp\left(\frac{\log\frac{(\mu B-1)q_1\alpha_1 - \mu\epsilon_1 M_1}{(\mu B-1)\mu\epsilon_2 M_2 - \mu\epsilon_1 M_1}}{\log \mu B}\right)\rceil.$$

That is,

$$T_1 \leq \lceil \left(\frac{(\mu B-1)q_1\alpha_1 - \mu\epsilon_1 M_1}{(\mu B-1)\mu\epsilon_2 M_2 - \mu\epsilon_1 M_1}\right)^{\frac{1}{\log\mu B}}\rceil. \tag{58}$$

5.5 Summary

This chapter describes the study of an infinite horizon stochastic lot-sizing problem. First, we show that under mild conditions on the stochastic data there exists an optimal solution to the problem. By using a shifting-up procedure similar to the primal algorithm in Chapter II, we show a monotonicity property of the optimal solution of the finite approximation problem of the original infinite problem. Next, we prove the solution and objective value convergence of the finite horizon approximation problem to the infinite horizon problem, as the planning horizon goes to infinity. Combining these results with the integrality of decision variables, we show the existence of a planning horizon for the infinite horizon stochastic lot-sizing problem, in the context of stochastic programming. Finally, we develop a formula to calculate the planning horizon.

CHAPTER VI

CONCLUSIONS

In this thesis, we report on our study of multi-stage stochastic programming models for production planning problems. We make the following contributions.

We start from a very simple multi-stage stochastic lot-sizing problem with a single item, with no fixed charge and capacity constraint. We develop primal and dual algorithms by exploiting the problem structure. Both algorithms are strongly polynomial and much more efficient than the Simplex method.

We propose a multi-stage stochastic programming model for capacity planning under uncertainty. First, we compare the two-stage model and multi-stage model. We discuss the concept of *value of multi-stage stochastic programming* and develop informative theoretical bounds. The analysis shows that the value of the multi-stage stochastic programming model can be very high for certain classes of multi-period capacity planning problems. Second, by studying the problem structure, we identify the simple stochastic lot-sizing problem studied previously as a key sub-model that can be solved by an efficient algorithm. Based on the decomposition structure and the property of the simple stochastic lot-sizing problem, we design an efficient approximation scheme for the capacity planning model. We show that the relative gap between the objective values of the approximation solution and the optimal solution converges to zero asymptotically as the number of stages of the scenario tree approaches infinity. We conduct numerical experiments for a semiconductor planning problem using industrial-size data, the results of which conform to our theoretical analysis.

We establish an infinite horizon stochastic lot-sizing problem with linear cost and single item. Under mild assumptions, we show that both the optimal solution and objective value of the finite approximation problem converge to that of the infinite horizon problem. Therefore, there exists a planning horizon for this infinite horizon stochastic lot-sizing problem. We develop useful upper bounds for the planning horizon. This study is the first for a planning horizon in the context of infinite horizon stochastic programming models. The infinite horizon stochastic lot-sizing problem here is different from previous stochastic dynamic programming models in that it is assumed that both the cost and demand data are random; the stochastic processes are general and all feasible solutions are considered.

The study reported here is a first step leading to a series of open questions in multi-stage stochastic programming models for production planning. As in the deterministic case, the stochastic lot-sizing problem is prevalent in many production planning problems under uncertainty. The study of variants of the simple stochastic lot-sizing problem studied in Chapter II can be useful in understanding more general production planning problems under uncertainty. For example, one can study the capacitated stochastic lot-sizing problem with fixed charge and the multi-item stochastic lot-sizing problem with budget constraints. It would also be interesting to compare stochastic lot-sizing models from the viewpoints of stochastic dynamic programming and stochastic programming. Chapter III presents a comparison of two-stage and multi-stage models in a static fashion, i.e., with a fixed finite time window. However, in practice one always uses a rolling horizon method. Therefore, it is of interest to study the performance of stochastic programming models under a rolling horizon approach. It is also of interest and importance to study conditions for the existence of a planning horizon in the context of general classes of infinite horizon stochastic programming problems.

REFERENCES

[1] S. Ahmed. Semiconductor tool planning via multi-stage stochastic programming. In *Proceedings of the International Conference on Modeling and Analysis in Semiconductor Manufacturing*, pages 153–157, 2002.

[2] S. Ahmed, A.J. King, and G. Parija. A multi-stage stochastic integer programming approach for capacity expansion under uncertainty. *Journal of Global Optimization*, 26:3–24, 2003.

[3] S. Ahmed and N.V. Sahinidis. An approximation scheme for stochastic integer programs arising in capacity expansion. *Operations Research*, 51:461–471, 2003.

[4] K.J. Arrow, T.E. Harris, and J. Marschak. Optimal inventory policy. *Econometrica*, 19:250–272, 1951.

[5] F. Barahona, S. Bermon, O. Gunluk, and S. Hood. Robust capacity planning in semiconductor manufacturing. *Naval Research Logistics*, 52(5):459–468, 2005.

[6] J.C. Bean, J.L. Higle, and R. L. Smith. Capacity expansion under stochastic demands. *Operations Research*, 40:210–217, 1992.

[7] J.C. Bean and R.L. Smith. Conditions for the existence of planning horizons. *Mathematics of Operations Research*, 9(3):391–401, 1984.

[8] J.C. Bean and R.L. Smith. Conditions for the discovery of solution horizon. *Mathematical Programming*, 59:215–229, 1993.

[9] J.F. Benders. Partitioning procedures for solving mixed variables programming problems. *Numersiche Mathematik*, 4:238–252, 1962.

[10] O. Berman, Z. Ganz, and J. M. Wagner. A stochastic optimization model for planning capacity expansion in a service industry under uncertain demand. *Naval Research Logistics*, 41:545–564, 1994.

[11] D.P. Bertsekas. *Dynamic Programming and Optimal Control.* Athena Scientific, 2000.

[12] C. Bes and S. Sethi. Concepts of forecast and decision horizon: Applications to dynamic stochastic optimization problems. *Mathematics of Operations Research*, 13:295–310, 1988.

[13] J.R. Birge. Decomposition and partitioning methods for multistage stochastic linear programs. *Operations Research*, 33:989–1007, 1985.

[14] J.R. Birge and F. Louveaux. *Introduction to Stochastic Programming.* Springer, New York, 1997.

[15] G.L. Cain. *Introduction to general topology.* Addison-Wesley, 1994.

[16] L.M.A. Chan, D. Simchi-Levi, and J. Swann. Dynamic pricing strategies for manufacturing with stochastic demand and discretionary sales. 2002. Submitted for publication.

[17] T. Cheevaprawatdomrong and R. Smith. Infinite horizon production scheduling in time-varying systems under stochastic demand. *Operations Research*, 52(1), 2004.

[18] Z.L. Chen, S. Li, and D. Tirupati. A scenario based stochastic programming approach for technology and capacity planning. *Computers and Operations Research*, 29:781–806, 2002.

[19] M.H.A. David, A.H. Dempster, S.P. Sethi, and D. Vermes. Optimal capacity expansion under uncertainty. *Advanced Applied Probability*, 19:156–176, 1987.

[20] E. Denardo. *Dynamic programming: Models and applications.* Prentice-Hall, 1982.

[21] M.N. EL Agizy. Dynamic inventory models and stochastic programming. *IBM Journal of Research and Development*, pages 351–356, July 1969.

[22] G.D. Eppen, R.K. Martin, and L. Schrage. A scenario approach to capacity planning. *Operations Research*, 37:517–527, 1989.

[23] C.H. Fine and R.M. Freund. Optimal investment in product-flexible manufacturing capacity. *Management Science*, 36:449–466, 1990.

[24] J. Freidenfelds. Capacity expansion when demand is a birth-death random process. *Operations Research*, 28:712–721, 1980.

[25] M.R. Garey and D.S. Johnson. *Computers and intractability: A guide to the theory of NP-completeness.* W. H. Freeman and Co., 1979.

[26] H. Geng. *Semiconductor Manufacturing Handbook.* McGraw-Hill, 2005.

[27] K.K. Haugen, A. Lokketangen, and D.L. Woodruff. Progressive hedging as a meta-heuristic applied to stochastic lot-sizing. *European Journal of Operations Research*, 132:116–122, 2001.

[28] S.J. Hood, S. Bermon, and F. Barahona. Capacity planning under demand uncertainty for semiconductor manufacturing. *IEEE Transactions on Semiconductor Manufacturing*, 16(2), 2003.

[29] W.J. Hopp and M.L. Spearman. *Factory Physics.* McGraw-Hill, 2000.

[30] S.M. Johnson. Sequential production planning over time at minimum cost. *Management Science*, 3(4):435–437, 1957.

[31] P. Kall and S.W. Wallace. *Stochastic Programming.* John Wiley and Sons, 1994.

[32] A. Kanudia. Robust responses to climate change via stochastic MARKAL: The case of quebec. *European Journal of Operational Research*, 106(1):15–30, 1998.

[33] S. Karabuk and S.D. Wu. Coordinating strategic capacity planning in the semiconductor industry. *Operations Research*, 51(6):839–849, 2003.

[34] L.A. Korf. An approximation framework for infinite horizon optimization problems in a mathematical programming setting. In *Springer-Verlag Lecture Notes in Mathematics and Economics: Proceedings of the BFG Conference on Optimization, Namur, Belgium*, 2000.

[35] R. Kouwenberg. Scenario generation and stochastic programming models for asset liability management. *European Journal of Operational Research*, 134(2):279–292, 2001.

[36] H. Luss. Operations research and capacity expansion problems: A survey. *Operations Research*, 30:907–947, 1982.

[37] A.S. Manne. Capacity expansion and probabilistic growth. *Econometrica*, 29:632–649, 1961.

[38] A.S. Manne, editor. *Investments for Capacity Expansion.* The MIT press, 1967.

[39] M. Melo. *Stochastic lot-sizing in production planning: Strategies for make-to-order and make-to-stock.* PhD thesis, Erasmus University, 1996.

[40] G.L. Nemhauser and L.A. Wolsey. *Integer and Combinatorial Optimization.* John Wiley and Sons, 1988.

[41] Y. Nishi and R. Doering. *Handbook of Semiconductor Manufacturing Technology.* Marcel Dekker Inc, 2000.

[42] State of New York Press Office. Governor and IBM open big blue's $2.5 billion chip plant. *http:\\www.semi-ny.com\article.asp?ID=23*, 2005.

[43] Y. Pochet and L.A. Wolsey. Algorithms and reformulations for lot sizing problems. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 20:245–293, 1995.

[44] W.B. Powell. A stochastic model of the dynamic vehicle allocation problem. *Transportation Science*, 20:117–129, 1986.

[45] U.S. Rao, J.M. Swaminathan, and J. Zhang. Multi-product inventory planning with downward substitution, stochastic demand and setup costs. *IIE Transactions*, 36(1), 2004.

[46] R.T. Rockafellar and R.J. Wets. Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of Operations Research*, 16(1):119–147, 1991.

[47] W. Romisch and Schultz R. Multistage stochastic integer programming: An introduction. *The Stochastic Programming E-Print Series (SPEPS)*, 2001-03, 2001.

[48] T.J. Roy. Cross decomposition for mixed integer programming. *Mathematical Programming*, 25:46–63, 1983.

[49] A. Ruszczynski and A. Shapiro (eds.). *Stochastic Programming.* Elsevier, 2003.

[50] I.E. Schochetman and R.L. Smith. Infinite horizon optimization. *Mathematics of Operations Research*, 14:559–574, 1989.

[51] I.E. Schochetman and R.L. Smith. Finite dimensional approximation in infinite dimensional mathematical programming. *Mathematical Programming*, 54:307–333, 1992.

[52] A. Shapiro. Inference of statistical bounds for multistage stochastic programming problems. *Mathematical Methods of Operations Research*, 58:57–68, 2003.

[53] R.L. Smith and R.Q. Zhang. Infinite stage production planning in time varying systems with convex production and inventory costs. *Management Science*, 44(9), 1998.

[54] M. Stafford. A product-mix capacity planning model. Technical Report ORP97-03, University of Texas at Austin, http://www.me.utexas.edu/~orie/techrep.html, 1997.

[55] J. Swaminathan. Tool capacity planning for semiconductor fabrication facilities under demand uncertainty. *European Journal of Operational Research*, 120:545–558, 2000.

[56] J. Swaminathan. Tool capacity planning for semiconductor fabrication facilities under demand uncertainty. *IIE Transactions*, 34:145–155, 2002.

[57] S. Takriti, J.R. Birge, and E. Long. Stochastic model for the unit commitment problem. *IEEE Transactions on Power Systems*, 11(3):1497–1508, 1996.

[58] M.H. van der Vlerk and W. K. Haneveld. Stochastic integer programming: General models and algorithms. *Annals of Operations Research*, 85:39–57, 1999.

[59] R.J-B. Wets and W.T. Ziemba. *Stochastic programming: State of the art.* Baltzer Science, 1998.

[60] P.H. Zipkin. *Foundations of Inventory Management.* McGraw-Hill, 2000.

VITA

Kai Huang was born on February 21, 1974, in Wuhan, China. He is the second child of Naiyu Huang and Yan Li, both professors at the Huazhong University of Science and Technology (HUST) where Kai lived for 23 years. He received a Bachelors degree in 1997 in Automatic Control Engineering from HUST and entered the Automation Department of Tsinghua University to pursue a Masters degree, in which he investigated supply chain management system design. He completed his Masters thesis, Multi-agent Coordination and Negotiation in Supply Chain Management, and his degree, in 2000. In the same year, Kai married Ying Luo and was admitted to the Ph.D. program in the School of Industrial and Systems Engineering at the Georgia Institute of Technology. In the first two years, Kai worked in the Virtual Factory Laboratory with Dr. Marc Goetschalckx and Dr. Leon McGinnis on systematic warehousing design and other industrial projects. In 2003, Kai began work with Dr. Shabbir Ahmed on stochastic programming. The research, first motivated by a semiconductor tool planning problem, led to a series of more general and interesting models and results. This thesis reports on Kai's PhD research on multi-stage stochastic programming models in production planning . His general research interests are in operations research and supply chain management.