

CUT SCHEDULING FOR OPTIMUM FABRIC UTILIZATION
IN APPAREL PRODUCTION

A THESIS

Presented to
The Faculty of the Division
of Graduate Studies

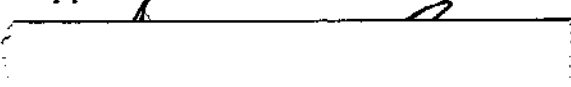
By
Howard S. Coff

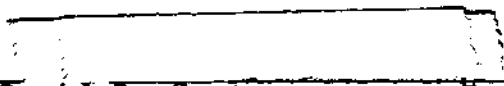
In Partial Fulfillment
of the Requirements for the Degree
Master of Science in Textiles

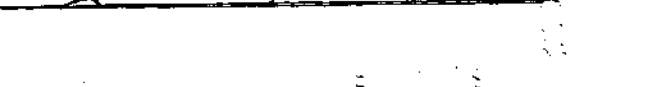
Georgia Institute of Technology
November, 1976

CUT SCHEDULING FOR OPTIMUM FABRIC UTILIZATION
IN APPAREL PRODUCTION

Approved:


Milos Konopasek, Chairman


David R. Gentry


John J. Jarvis

Date approved by Chairman: 11/15/76

DEDICATION

I gratefully dedicate this thesis to my parents,
Mr. and Mrs. David H. Coff, for their love and encouragement.

ACKNOWLEDGMENTS

I would like to express my deepest appreciation to my thesis advisor, and friend, Dr. Milos Konopasek, whose guidance, counsel and encouragement made this thesis possible.

I am grateful to Dr. David R. Gentry and Dr. John J. Jarvis for serving on my reading committee.

Special thanks should go to Chris Papaconstadopoulos for his help and guidance in the experimental work on the HP minicomputer.

I would like to express my gratitude to Camsco, Inc. for sponsoring the research project which introduced me to the exciting field of the use of computers for the improvement of fabric utilization in the apparel industry.

I also want to express my thanks to the management of Oxford Industries, Inc. for providing the technical information used in my experiments. Part-time employment with the company during the last stage of my thesis work gave me invaluable insight into the intricacies of practical applications of the results of my research.

TABLE OF CONTENTS

	Page
ACKNOWLEDGMENTS.	iii
LIST OF ILLUSTRATIONS.	vi
SUMMARY.	vii
Chapter	
I. INTRODUCTION.	1
1.1. Statement of the Problem	
1.2. Review of Literature	
II. LINEAR PROGRAMMING AND ITS IMPLEMENTATION IN CUT ORDER SCHEDULING	9
2.1. Formulation of Linear Programming Problem	
2.2. Integer Programming	
III. INTERACTIVE LP PROGRAMS	21
IV. SPECIAL PROBLEMS IN CUT ORDER PLANNING.	25
4.1. Dummy Markers	
4.2. Cut-Downs	
4.3. Cut-Down Assignment	
V. FINAL SET-UP OF THE PROGRAM PACKAGE	38
5.1. Cyber 74 Version	
5.2. HP Version and Industrial Implementation of the Package	
VI. CONCLUSIONS	51
VII. RECOMMENDATIONS	55
Appendix	
A. SIMPLE EXAMPLE SOLVED BY XLINEAR.	60
B. ENTERING, DISPLAYING, SOLVING, STORING, AND TERMINATING THE SIMPLE EXAMPLE PERFORMED BY LP4.	61

Appendix	Page
C. SOLUTION PRINTED BY UTILIZING "ALL" OPTION OF PROGRAM LP6.	63
D. LP6 FLOWCHART.	67
E. INPUT FILES.	75
F. TEMPORARY FILES.	78
G. PRODUCTION REPORTS	79
BIBLIOGRAPHY.	81

LIST OF ILLUSTRATIONS

Figure	Page
1. Example of Branch and Bound Method.	14
2. Solution of Integer Programming Problem by Branch and Bound Algorithm as Implemented in LP6.	15
3. Procedure for Obtaining an Integer Solution Generating of Additional Constraints in Integer Programming	17
4. Block Diagram of Cut Order Scheduling Program Package	39
5. Relationship of the Five Modules of the HP Version of the System	44
6. Program Relationship and File Organization of the System	45

SUMMARY

The broad objective of this study is to survey the existing and possible computer assisted techniques to optimize fabric utilization in the apparel plant. Special attention is paid to optimum cut scheduling for given quantities of garments produced from a stock of fabrics using a given variety of markers (pattern layouts). This supplements the existing computerized methods of marker making (layout design) and automatic digitally controlled cutting.

The fabric consumption to be minimized is a linear function of the required quantities of style/sizes and the waste percentages of the applied markers. The constraints concerning the quantities of the garments required and the fabrics available are also linear functions of the marker parameters. Consequently, the optimum cut schedule can be obtained by using linear programming.

An appropriate method of linear programming was selected and a conversational computer program was developed. This method will provide an easy-to-use tool for solving the scheduling problem in every day activities at an apparel manufacturing plant.

CHAPTER I

INTRODUCTION

1.1. Statement of the Problem

Only in the past ten years have there been any new and significant developments in the cutting department. Considering raw material costs typically run 40 percent to 50 percent of the wholesale price of an apparel product, it is difficult to explain this neglect. The level of fabric utilization in a cutting room is determined by three main factors: pattern engineering, marker making, and the selection of markers to cover a particular production plan.*

A basic requirement to achieve high material utilization is designing the patterns in a logical way that brings about an appropriate garment construction consistent with fashion and comfort requirements and also with fabric utilization and garment assembling aspects [1].

The second element which determines fabric utilization is the way in which patterns are arranged on the marker. This is a decisive stage where fabric waste can be minimized; the most progress has been made here in recent years. The most

* Making markers is the grouping and interlocking of various sized patterns of a given style and size range into the tightest formations possible within a given width [10].

important advance in this area was an implementation of computer graphics as an aid to a marker designer.

In a typical computer-assisted marker making system, the information on pattern shapes, taken from a digitizer (converts pattern geometry into computer-acceptable numerical form), is stored in the computer memory. Sets of patterns are then displayed on a cathode ray tube (CRT). A marker designer controls the movement of individual panels by a stylus and/or control box to achieve the most efficient layout. The computer prevents overlapping of panels, secures predetermined match of stripes, plaid, etc., stores the information about the markers and relays this information to a computer driven plotter, or, via magnetic tape or disk to a digitally controlled cutting device [5].

Fabric utilization also depends to a large extent on the way the stock of fabrics (possibly in a few different widths) is scheduled for the products of different styles and sizes. In contrast to the computer-assisted marker design, there has been little development in this area. This investigation deals mainly with the third area, optimal marker selection to cover a particular production plan.

The individual markers are made for a specific combination of styles and sizes on a material of a given width. The bank of markers provides a unique basis for selection of relevant markers and optimal scheduling. Quite often existing markers are neglected. This is a waste of previous marker

making efforts; it is done because manual retrieval of markers is a cumbersome process.

The selection of markers is now frequently accomplished by either first match or enumeration method. In the first match method, the first combination of markers observed that contain all required style/sizes in the cut order is the one used. This method is obviously inefficient.

The enumeration method is carried out by manually listing several marker combinations in order of efficiency. This is time consuming and the solution is most likely not optimal.

The intuitive way of assigning the planned quantities of styles/sizes to available fabrics, starting with combinations with the highest efficiency percentage, usually will not lead to optimum. The unfavorable assignments left to be allocated at the end cannot compensate for the favorable ones made at the beginning.

The purpose of this study is to survey the existing and possible computerized ways to optimize fabric utilization by selecting the most favorable combination of markers which satisfy the cut order requested. Hopefully, this method will provide an easy-to-use tool for solving the cut scheduling problem in everyday activities at an apparel manufacturing plant.

1.2. Review of Literature

As noted in the previous section, very little has been done or is known about using mathematical methods and computing techniques for optimum scheduling of available fabrics and markers in the cutting department. Therefore, in the survey of literature the emphasis will be on the progress in computer-assisted marker making, because this made efficient cut-scheduling possible and desirable.

The references to the sources of mathematical techniques used during the investigation will be made in Sections 2.1 and 2.3.

Some of the new developments in the cutting department in the last ten years include individual incentive systems, computerized pattern grading,* miniature markers, etc. [5].

Computerized marker making is the bridge between computerized pattern grading and computer controlled cutting. It passes the information from the grading operation to cutting through a computer controlled plotter. The plotter draws full scale master markers from the arrangement of panels created on the CRT and stored in memory. A plotter can draw marks for pockets, darts, and notches. Various statistics, usually the pattern piece area, can be calculated and printed with every graded pattern.

*Grading is the process by which a garment pattern--cut to fit a group of persons of a standard size--is made larger and smaller to fit groups of larger and smaller people [1].

Grade rules are how each point (in the pattern) of significance moves inwards and outwards to the corresponding point of other sizes [5]. Grade rules along with digitized patterns are used in both pattern grading and marker making. Therefore, the integrated marker making systems will mean reduced costs in the pattern making area in reference to labor, supplies, and occupancy [3].

The computer-assisted marker making systems primarily reduce the marking labor costs. Productivity increases that of conventional marker making mainly due to the elimination of manually drawing around the patterns which is substituted for by the use of a plotter.

Quality of the marker making operation can also be improved through use of the computerized systems. Pattern pieces can be drawn with improved accuracy because marking restrictions are more easily controlled by the computer software.

The greater productivity provided by the computer-assisted marking systems can be used to meet peak workloads or permit additional effort to improve marker efficiency. If increased capacity is available, then throughput time in marker making can be maintained in periods of peak workloads. The value of any throughput improvement varies depending on what the limiting factors (or binding constraints on the total operation) are for a particular company [4].

One of the most advertised features of the computer

assisted marking system is the impact on material utilization. Marking on a reduced scale provides pattern area measurements which when divided by total marker area, gives the marker efficiency. Utilization is improved compared to the conventional method because less time is needed to make the original marker and therefore more time can be used to improve efficiency. Also the fact that "more markers made by your best marker maker using this equipment can reduce fabric waste" is a valid consideration [4].

If every advancement in manual marker making were implemented, the improvement in layout efficiency with computer-assisted systems is probably negligible. But, not every apparel manufacturer has the opportunity to install every innovation in manual marker making. Therefore computer assisted marker making has the advantages in that it can solve the need for rapid response and at a higher efficiency [4].

Computerized pattern grading is the source data of manufacturing control. The data are digitized and controlled by the grading rules to be used. This additional output can be used for further stages in production. Because computerized pattern grading systems perform the translation from a sample size pattern into a full range of sizes, it primarily affects the pattern making and grading labor costs.

Quality is improved by the fact that possible human errors are avoided. Without the accuracies attainable in

computerized grading, such things as plaid and stripe cutting and sewing, reduction in seam allowances, and pattern modifications become very tedious.

Computerized grading systems can improve the capacity to process many pattern sets in a given period, thereby reducing cycle time. This is sometimes necessary due to seasonal or style changes. The computerized grading system, like computerized marker making, is most useful to those firms where frequent style changes and short lead times are required. Commercial computer grading services are available for a producer of relatively few styles, who may find it difficult to justify the expense for hardware.

The current state of the art in computerized marker making still requires a man at some point to direct the computer to perform certain functions, such as selection and placement of pattern pieces which are necessary to build the marker [9]. In addition to existing software, some developments have been added to some systems which assist the marker maker. These aids can pack or squeeze the panels in a tighter formation in an attempt to reduce the length once the marker is created. This can improve the marker efficiency achieved by the operator. Additional software has been developed to further reduce waste in creating markers.

There is research currently being done in this area on the methods of arranging the patterns in an optimal formation without human intervention. This usually involves

an iterative procedure and/or uses previous experience and predefined marker making rules. An efficient fully automatic marker making system has not yet become a reality.

CHAPTER II

LINEAR PROGRAMMING AND ITS IMPLEMENTATION IN CUT ORDER SCHEDULING

Although some aspects of marker making have been vastly improved, the decision concerning "what to cut from what" is still performed by individuals based on general guidelines. When a cut order (required quantities of garments of given sizes and styles) is received, the first stage of the marking procedure is to decide what combination of sizes/panels should be arranged together to form a marker. This process is critical because the efficiency is determined here. Many factors such as the number of markers which will have to be created (to satisfy the cut order), the mechanics of laying out the cloth, and the pile height of each marker, must be considered. Because this process is time consuming, whether it is done by retrieving old markers or creating new ones, and is rarely optimal, a different approach was investigated.

2.1. Formulation of Linear Programming Problem

Linear programming deals with the problem of allocating limited resources among competing activities in the best possible way. Linear programming uses a mathematical model to describe the problem. As the name implies, the mathematical

functions in this model are required to be linear. Linear programming involves the planning of activities to obtain an optimal result among the feasible alternatives. This can be illustrated by the following simple example. Assume that a company manufactures two products, each with the same raw material costs and each has unlimited demand. Let x_1 and x_2 represent the number of products of product 1 and product 2 produced per hour. The selling price is one dollar for product 1 and two dollars for product 2. X_1 and X_2 are the decision variables for the model and the objective is to choose the combination so as to maximize function Z where

$$Z = X_1 + 2X_2$$

subject to the following restrictions

$$X_1 + X_2 \leq 6$$

$$2X_1 + X_2 \leq 8$$

This problem is of the classic "product mix" type (how much to produce of what). The solution to this problem, solved by program XLINEAR (Appendix A), is to produce 6 units of product two per hour and none of product one.

Both the exhaustion of the stock and the saturation of the planned quantities of the style/sizes are linear

functions of the numbers of layers to be cut using different markers. Therefore, the optimum fabric utilization given a certain production plan, fabric stock, and marker bank may be solved as a linear programming problem.

The total area of the fabric

$$A = \sum_{j=1}^m \sum_{i=1}^{\ell} z_j w_i h_{ij}$$

is to be minimized, subject to the constraints concerning the production plan

$$\sum_{j=1}^m z_j p_{j,k} = S_k \quad \text{for } k = 1, 2, \dots, n$$

and the fabric stock available

$$\sum_{j=1}^m z_j h_{ij} \leq t_i \quad \text{for } i = 1, 2, \dots, \ell$$

where

- w_i is width of the fabric
- t_i is total length of the fabric w_i wide on stock
- z_j is number of layers cut using j -th marker
- $h_{i,j}$ is element of an auxiliary marker length matrix denoting length of j -th marker w_i wide,
 $h_{\alpha,j} = 0$ for $\alpha = 1, 2, \dots, i-1, i+1, \dots, \ell$
- $p_{j,k}$ is number of sets of patterns of k -th style/size in j -th marker

S_k is required quantity of k-th style/size,
 all for $i = 1, 2, \dots, l$; $j = 1, 2, \dots, m$; $k = 1, 2, \dots, n$
 where

l is the number of fabric widths
 m is number of markers
 n is number of style/sizes

Using linear programming, the best possible combination of markers may be selected to satisfy the required sizes as well as several other constraints.

2.2. Integer Linear Programming

The solution of the linear programming problem consists of the combination of markers which give the best overall utilization. The solution will usually include fractional numbers. (For example $x_1 = 2.67$ --part of the original solution in Figure 2 (Appendix C). In practice it is meaningless to have 2.67 layers of fabric. Therefore, the solution makes sense only if the decision variables have integer values.

The branch and bound algorithm of pure integer programming is used to restrict the solution to integer values. This method leads to optimum integer solutions through solving a sequence of related non-integer problems.

A typical linear programming problem given by Wagner [12] can be used to illustrate how the integer solution is obtained. The original problem appears as follows:

$$Z = 3x_1 + 3x_2 + 13x_3$$

$$-3x_1 + 6x_2 + 7x_3 \leq 8$$

$$6x_1 - 3x_2 + 7x_3 \leq 8$$

where function z is to be maximized by employing the branch and bound algorithm. The solution to the original problem is $x_1 = x_2 = 2.67$. It must be altered to comply with integer constraints. The tree diagram shows how Wagner obtained the sequence of solutions leading to the integer optimal solution (Figure 1).

The term simplex iteration will refer to a step in the linear programming algorithm which uses the simplex method to reach the optimum. The term branch and bound iteration will be used to indicate an addition of a constraint or "branching" of the problem in seeking to find the optimal integer solution.

The latest branch rule is implemented in order to obtain an integer solution. One advantage of this method is that it decreases the amount of bookkeeping necessary. This rule selects the most recently created subset of branches that has not been fathomed, breaking a tie between subsets created at the same time by taking the one with the most favorable bound (Figure 2).

After some experiments, it was decided to design the

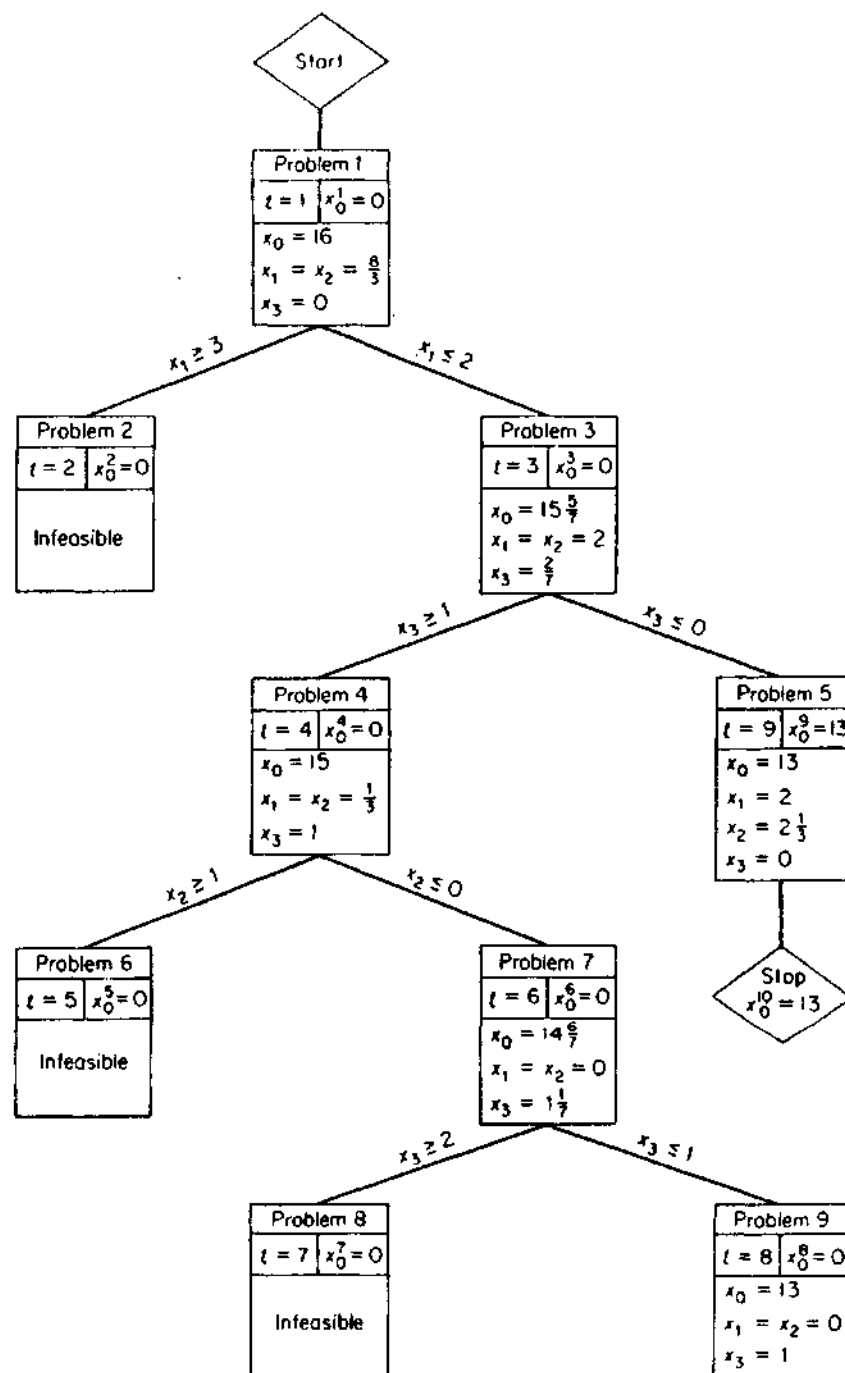


Figure 1. Example of Branch and Bound Method [12]

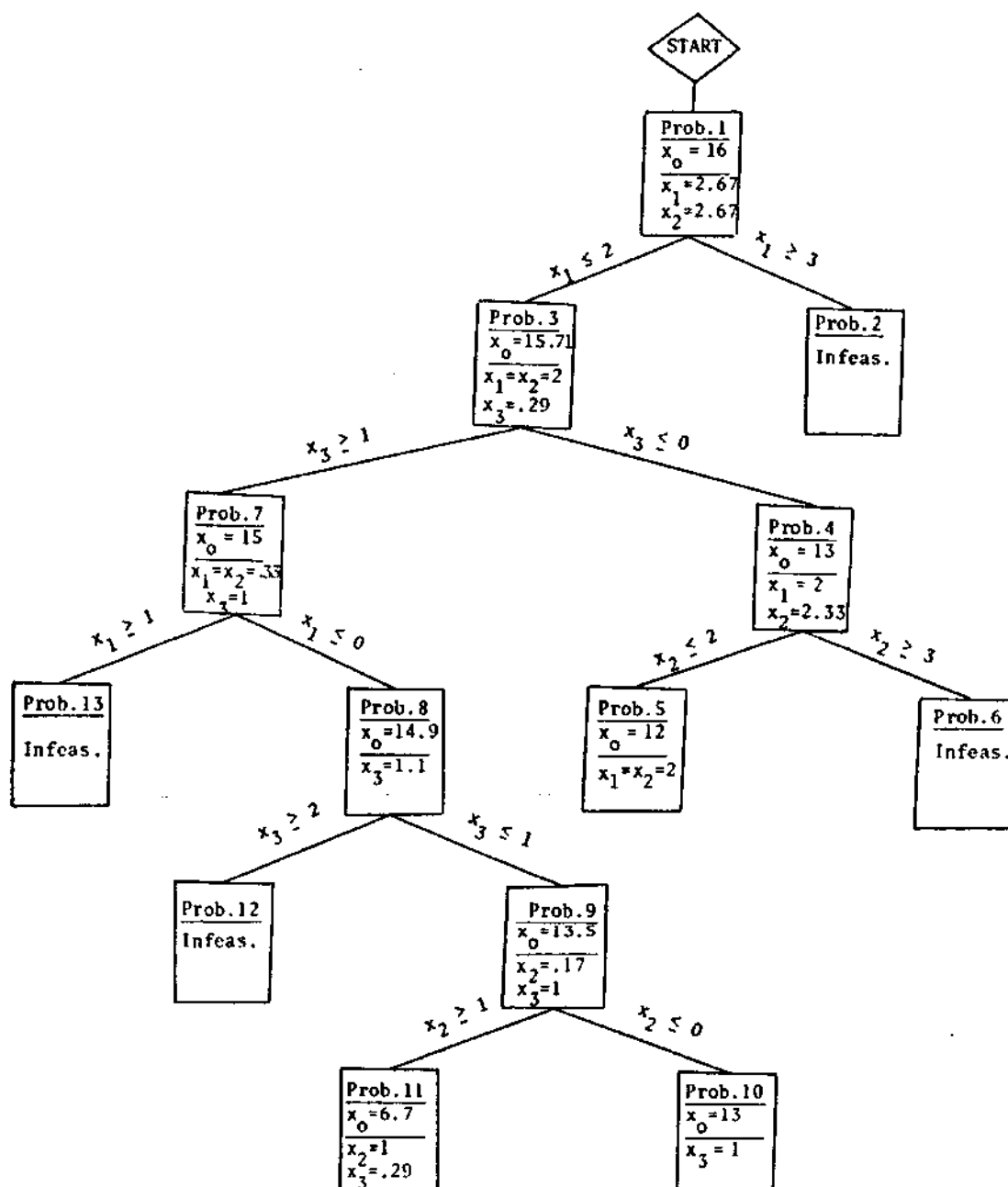


Figure 2. Solution of Integer Programming Problem by Branch and Bound Algorithm as Implemented in LP6

program so that the variable with the smallest positive fractional value is selected to be "split on." This procedure usually allows the program to reach the solution faster. It was also found that if the variable selected is always rounded down first, this will further accelerate convergence to the first integer solution of the cut planning problem.

The original solution found by linear programming is converted to integer by the procedure in Figure 3. Basically the problem begins with the entire set of solutions under consideration. Once it is determined that the solution is feasible, it is further considered. If the solution is integer and it is better (smaller for minimization) than the best integer solution (BIS) up to that point, then this is the new BIS and is stored as the new incumbent solution. If the trial solution (whether it is integer or real) is not better than the present incumbent, the trial solution and potential branches from it is no longer considered. The reason the non-integer solution can be eliminated is because branching further (adding more constraints) can only make the solution value worse. Therefore, at this point a branch of the tree may be fathomed without enumerating all the solutions.

If the solution is not integer, but its value is better than the incumbent, then one of the fractional variables (using the newest bound rule) is "split on."

The following variables and their meanings correspond

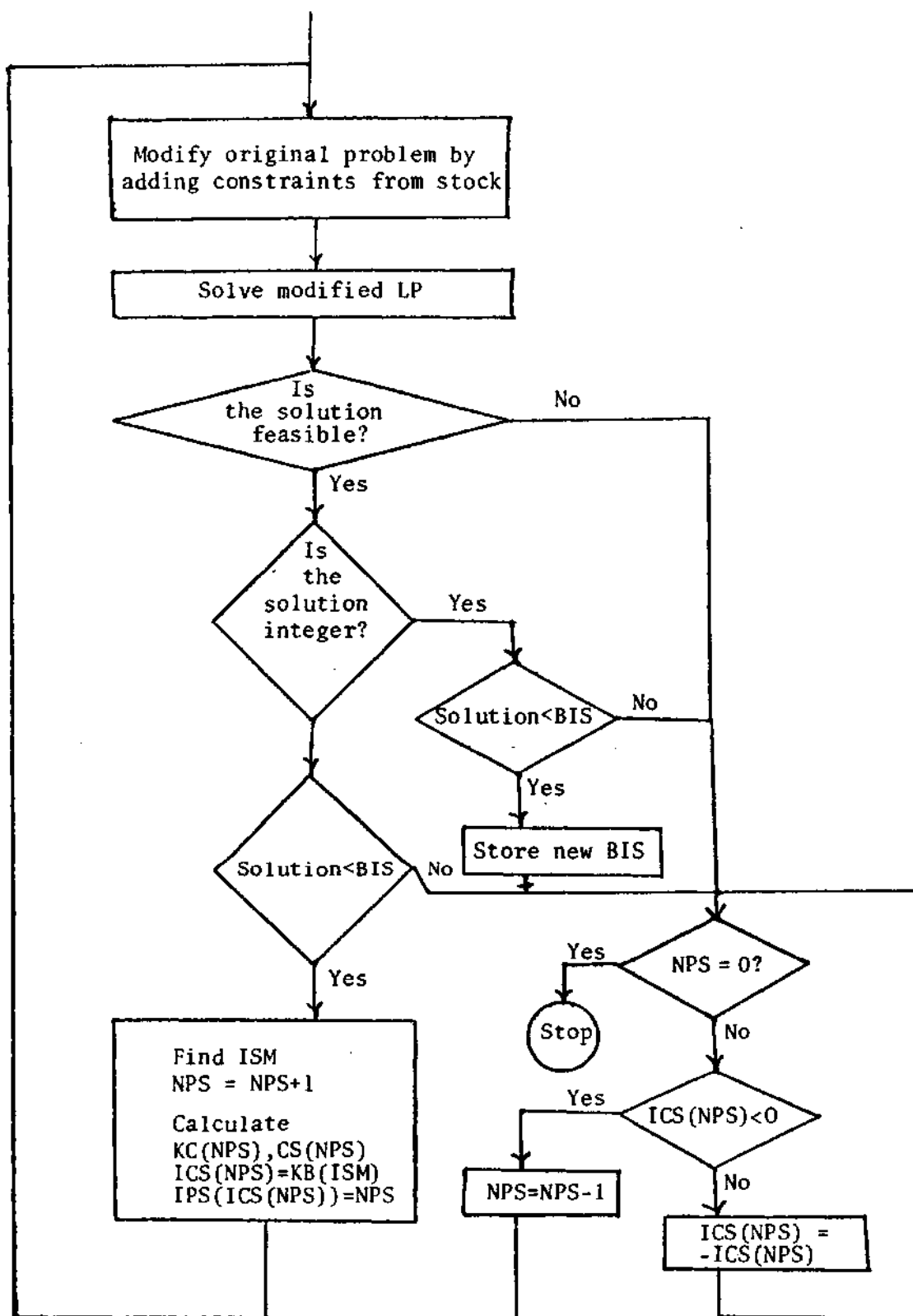


Figure 3. Generating Additional Constraints in Integer Programming

to Figure 3. Note that new variables are chosen in order of increasing positive fractional values as discussed earlier.

BIS	Basic Integer Solution
NPS	Number of additional constraints in stack
ISM	Order index of variable with smallest remainder
KCS(i)	Kind of constraint
CS(i)	Value of constraint
ICS(i)	Index of constrained variable--carrier of branching information
IPS(i)	Pointer to location in constraint stack where constraint of j-th variable is stored
KB	array of pointers linking solution columns with variable indices

The solution of Wagner's example obtained by using this algorithm is shown in Figure 2 and its printout is given in Appendix C. There are some differences between Wagner's tree diagram in Figure 1 and Figure 2; Wagner's solution appears to need fewer branch and bound iteration steps. However, closer examination shows some unexplained short-cuts in Figure 1. For instance, it is difficult to see why after problem 3 the inequality $X_3 \geq 1$ should be preferred to $X_3 \leq 0$ when X_3 is equal to $2/7$, i.e., is closer to zero than one. Also, there is no obvious reason why X_2 is constrained in preference to X_1 after problem 4.

Any bounded integer programming problem has a finite number of feasible solutions. However, this finite number

usually is very large so that complete enumeration is not practical. Therefore, it is important that only a small fraction of the feasible solutions be examined.

Despite these techniques to reduce the time to implicitly enumerate the solutions, several problems proved to take too long to use this method. This is due to either too many variables and/or a very heavily constrained problem. When dealing with a large problem, requesting only the first integer solution is sometimes necessary due to time limitations. In a large problem the difference between the first integer solution and the optimal solution will usually be marginal. However, in a relatively small problem, the difference may be significant. Because of this situation, the program which prints the solution without the technical data offers the options "first" and "best". When "first" is chosen, the computation is terminated after obtaining and printing the first integer solution. It has been found that this solution is usually optimal or near optimal. In this particular case (Figure 2), the first integer solution value is twelve, whereas the optimal solution value is thirteen (problem 10).

Conditions will determine whether "first" or "best" which gives optimal solution, is appropriate. For instance, if computer time is relatively inexpensive and the cut order is relatively large, it would be worth the extra computation time to request "best" solution. There is also

an option ALL in the program LP6 which triggers printing the solution of all non-integer and integer solutions during the Branch and Bound iteration. This option is used in Appendix C. In this case the additional constraints currently in force are printed at the beginning of every branch and bound iteration. Constraints are coded in the following way: the first number is the constraint number; the second number is the subscript of the variable which is constrained. The third number is the kind of constraint (-1 for greater than or equal to and 1 for less than or equal to). The fourth number is the value of a given constraint. The second number is preceded by a negative sign when both possibilities (rounding down and rounding up) for a given variable have been checked out.

For example, problem 2 is constrained by $x_1 \geq 3$ and problem 3 by $x_1 \leq 2$. Both alternatives, by the time problem number three is considered, have been chosen. Therefore the second number is preceded by a negative sign at problem number three.

CHAPTER III

INTERACTIVE LP PROGRAMS

A simple interactive LP (simplex algorithm) program XLINEAR used at Georgia Tech for teaching purposes was selected as a vehicle for preliminary experiments and further development. The problem solving part of the program is satisfactory. However the conversation part is rather poor: the user is expected to type in a complete problem specification (coefficients of object function, constraints, all the matrix coefficients); there is no provision for storing and editing the problem specification and for re-entering the solving part.

A series of modifications have been made in order to improve the conversation part for more efficient experimentation on various aspects of cut scheduling. The modification went through the following stages:

- LP0: changing specified objective function coefficients, constraint values, matrix elements and also repetition of the solution is made possible.
- LP1: The code of the solving part was simplified. Interactive alteration of the number of variables and constraints is added. There is a provision made for writing the formulated

problem into a data file and accessing stored problems as well as displaying the problem during the conversation.

LP2: Minor changes and improvement of efficiency.

LP3: Finalizing the design of conversation. The following responses to computer's request for the next step are available:

PROBLEM...initiate entering a problem, either typing in from terminal or reading from data file; DISPLAY...prints objective function coefficients, constraints, and matrix of the current problem.

ALT COF	}	alteration of specified objective function coefficient, constraint, or matrix element.
ALT RS		
ALT MC		
ADD VAR	}	adding or removing any variable or constraint to and from the problem.
REM VAR		
ADD CNSTR		
REM CNSTR		
STORE		Writes the whole problem into specified data file.
SOLVE		Solves LP problem using simplex method, prints out the trace of iteration (column replacement and current value of object function) and optimal solution.
END		Terminates the conversation.

- LP4: Differs from LP3 in the following respects:
- (a) Does not require inputting the basic part of the matrix and objective function coefficient. User indicates the type of constraint (+1, 0, -1 for \leq , $=$, \geq respectively) and program creates the basis using the subroutine SUSLAR;
 - (b) Does not display the base part of the problem;
 - (c) Prints out the trace of the iteration process only when asked;
 - (d) Prints the solution in natural order of variables including coefficients of object function for easier interpretation of results (Appendix B).

The flow chart for LP4 is the same as the flow-chart for LP6 (see Appendix D) without the "SOLVE INT" option.

- LP5: Differs from LP4 in that the internal matrix is represented in a "string" containing only the nonzero coefficients so that less storage is necessary. The number of non zero matrix coefficients increases due to the pivoting procedure in the simplex algorithm so that more numbers must be stored at later iterations. But, even this greater number is considerably less than storing a full matrix. The matrix is stored in three one-dimensional arrays; one

storing the actual coefficients; another identifies the row each coefficient is located in; the third adds cumulatively the numbers of nonzero coefficients in each column.

LP6: This is the first version of the program that is capable of reaching an integer solution. A flow chart showing the dialogue in program LP6 is in Appendix D. All the user commands have previously been explained except for "SOLVE INT". This simply tells the computer that an integer solution is desired. Then the computer will ask whether all, only integer, or only best integer solution is to be printed.

LP7: Is identical with LP6 with the only difference in format of the representation of the problem matrix in written and read-in problem data files; whereas the LP6 uses full matrix representation, the LP7 uses a string representation which makes it compatible with files TEMP2 created by the production version of the cut order scheduling program package described in Chapter 4.4.

CHAPTER IV

SPECIAL PROBLEMS IN CUT ORDER PLANNING

The formulation of real linear programming problems in cut order planning involves special considerations which are not covered by the introduction to the problem in Chapter II. This applies in particular to the situations when a given cut order cannot be satisfied by the exclusive use of existing markers, and to so called "cut-downs". It would be possible to account for these and other peculiarities by modifying the linear program manually (using for example interactive program LP6, see Chapter III). However, it would be rather inefficient and time consuming. Therefore in this chapter we concentrate on defining the rules for modifying the linear programming problem so that the necessary manipulation may be programmed and performed automatically by the computer.

4.1. Dummy Markers

Markers available in the marker bank or marker library sometimes cannot completely satisfy the cut order requested. In this case additional markers will have to be made of the remaining sizes to satisfy the cut order.

Utilizing the formulation of the linear programming problem, introduced in Chapter II, the unsatisfied part of

the cut order is completed by artificial variables which can come into the solution. These variables are assigned large coefficient values (for a minimization problem) in the objective function and therefore come into the final solution only when the cut order cannot possibly be fulfilled otherwise.

If artificial variables are used there is no way the overall efficiency of the solution can be assessed because the objective function coefficients of artificial variables are worse (higher) than any real marker. All markers regardless of their efficiency, will come into the solution before any artificial variables. The artificial variables will then fill the unsatisfied part of the order; because the objective function coefficients of all the artificial variables are equal and the solution would be biased towards better coverage of smaller sizes by regular markers. This is the reason why the concept of a "dummy marker", that of an imaginary one-size-one garment marker with a given efficiency, is introduced. If the cut order cannot be satisfied at all, the program will automatically bring "dummy markers" into the solution. These dummy markers, one for each style/size, are considered in the problem along with the real markers.

There is one other purpose that these dummy markers serve besides coming into solution when it is impossible for real markers to cover cut order. Dummy markers can be used to allow only the best real markers to come into the

solution. For example, the user enters in a desired efficiency, 82 percent. Only markers with 82 percent efficiency or greater will enter into the final solution (with dummy markers present where sizes could not be covered by regular markers). The final output will, therefore, consist of a list of the markers to be used from the library plus a list of sizes that must be marked to satisfy the cut order.

By increasing the desired efficiency fewer regular markers will be able to satisfy the requirement and the number of dummy markers in the solution will increase. On the other hand, if the efficiency asked for is lowered, more markers will be able to comply with this lower efficiency and fewer dummy markers will appear in the solution. It is necessary to make dummy markers competitive with real ones so that they could be added as regular variables in the linear programming problem. The objective function coefficient of each marker is the area of fabric the particular marker requires. Therefore, the area each size/style uses has to be calculated in similar fashion. Oxford Industries, Inc., Monroe, supplied real data for complete style named DUA. Utilizing the markers which had only one style/size, 34 sizes could be formulated by

$$A_p = W \cdot L \cdot E$$

where

A_p is area in square yards utilized

W is width of marker

L is length of marker

E is efficiency percentage for a given marker.

There are, however, a total of 158 different sizes in marker bank DUA. Utilizing the 34 data points already obtained, linear multiple regression was used to obtain the remaining 124 sizes. The two independent variables, length and waist and the corresponding areas were used to formulate the three coefficients a , b , c .

$$A = aW + bL + c$$

Once the three coefficients were found, the areas of every particular style/size could then be calculated. These areas are correct assuming 100 percent efficiency so that by dividing each area by the efficiency required, the objective function coefficient is evaluated.

For instance, if a larger efficiency percentage is wanted, this efficiency will result in smaller values of dummy marker objective function coefficients. Therefore, in a minimization problem the dummy markers have greater probability of appearing in the final solution.

4.2. Cut-Downs

The freedom of selecting the cutting schedule with

optimum fabric utilization is usually severely restricted by

- (a) fragmentation of orders;
- (b) differences between size distribution in the cut order and in the available markers;
- (c) difficulties or impossibility to create new markers following exactly the size distribution in the individual orders;
- (d) required level of efficiency of running the cutting operations from the viewpoint of labor costs, equipment utilization, etc.

Very often the only way of satisfying (d) under the conditions (a), (b), (c) is to cut the full height of the spread fabric into a number of panels exceeding the order requirements in certain sizes, and consequently to cut down part of the larger panels into smaller ones as required.

This method inevitably leads to a slight increase in waste due to additional reduction of the useful area of some of the panels. However, this may be compensated (even from the viewpoint of material utilization only) by avoiding possible inefficient distribution of sizes in a marker and by considering all the markers available. After all, minimization of fabric consumption takes into account the adverse effect of any additional waste accompanying the cut-downs.

The constraints concerning required number of garments of different sizes, consequently have to be modified in order to reflect the possibility of reducing some of the larger

size panels into panels of smaller sizes. No similar problem was found in the literature on linear programming.

The problem may be formulated as follows: The plan requirements for cutting schedule without cut-downs is given by a series of equality constraints (see Chapter II):

$$\sum_{j=1}^m z_j P_{j,k} = S_k \quad \text{for } k = 1, 2, \dots, n$$

where

z_j is the j -th component of the solution vector of the number of layers to be cut using j -th marker;

$P_{j,k}$ is number of sets of panels of k -th style/size in j -th marker;

m is number of markers under consideration;

n is number of style/sizes;

S_k is required quantity of k -th style/size

Supposing there is such a group of consecutive style/size items $k = \alpha, \alpha+1, \dots, \ell, \dots, \beta-1, \beta$, that panels for every item ℓ may be cut from panels for item $\ell+1$ (but not vice versa). In this case the equality constraints have to be replaced by the following ones:

$$\sum_{k=\alpha}^{\alpha+i} \sum_{j=1}^m z_j P_{j,k} = \sum_{k=\alpha}^{\alpha+i} S_k \quad \text{for } i = 0, 1, \dots, \beta-\alpha-1$$

$$\sum_{k=\alpha}^{\beta} \sum_{j=1}^m z_j P_{j,k} = \sum_{k=\alpha}^{\beta} S_k$$

For instance, if we have three markers yielding the following quantities of each of four sizes

	M1	M2	M3
S1	2	--	4
S2	--	3	--
S3	2	--	4
S4	1	3	--

and if the required quantities in sizes are

S1	S2	S3	S4
160	200	240	180

and the unknown numbers of layers to be cut from each marker are denoted X_1 , X_2 , X_3 , the constraints allowing cut-downs of larger sizes into smaller are

$$2X_1 + 4X_3 \leq 160$$

$$2X_1 + 3X_2 + 4X_3 \leq 360$$

$$4X_1 + 3X_2 + 8X_3 \leq 600$$

$$5X_1 + 6X_2 + 8X_3 = 780$$

In some situations a two dimensional cut-down is used in order to secure greater flexibility in obtaining a solution. This is used for instance in the manufacturing of slacks where not only the inseam length but also the waist may be reduced. (In the former case the term cut-off is used.)

When allowing for two dimensional cut-downs one has to modify the matrix coefficients and equality constraints in a manner similar to that of one dimensional cut-downs. The values of the matrix coefficients and right hand side values are transformed by cumulation in two directions instead of one:

$$\tilde{X}_{i,j} = \sum_{\zeta=1}^i \sum_{\eta=1}^j X_{\zeta,\eta}$$

where

X = original value

\tilde{X} = transformed value

The transformation procedure may be illustrated by the following example. Let the original matrix and constraints be

i,j	Size		A	B	C	D	E	Constraint
1,1	30	27	2	0	0	0	0	= 2
1,2	30	28	0	1	1	0	0	= 4
1,3	30	29	0	0	0	1	0	= 0
1,4	30	30	0	0	0	0	2	= 0
2,1	31	27	0	0	0	1	0	= 0
2,2	31	28	0	0	0	0	5	= 6
2,3	31	29	1	0	0	0	0	= 0
2,4	31	30	0	3	0	0	0	= 1
3,1	32	27	0	0	0	2	0	= 1
3,2	32	28	3	0	0	0	1	= 0
3,3	32	29	0	0	4	0	0	= 3
3,4	32	30	0	0	3	0	0	= 2

The right hand side values are transformed according to the last equation as follows:

	27	28	29	30		27	28	29	30
30	2	4			30	2	6	6	6
31		6		1	31	2	12	12	13
32	1		3	2	32	3	13	16	19

Original
Transformed

The matrix columns corresponding to individual markers have to be transformed in a similar way, so that the formulation of the problem will change to the following:

i,j	Size		A	B	C	D	E	
1,1	30	27	2	0	0	0	0	≤ 2
1,2	30	28	2	1	1	0	0	≤ 6
1,3	30	29	2	1	1	1	0	≤ 6
1,4	30	30	2	1	1	1	2	≤ 6
2,1	31	27	2	0	0	1	0	≤ 2
2,2	31	28	2	1	1	1	5	≤ 12
2,3	31	29	3	1	1	2	5	≤ 12
2,4	31	30	3	4	1	2	7	≤ 13
3,1	32	27	2	0	0	3	0	≤ 3
3,2	32	28	5	1	1	3	6	≤ 13
3,3	32	29	6	1	5	4	6	≤ 16
3,4	32	30	6	4	8	4	8	$= 19$

Note: all the equality constraints but the last are now "less than or equal to."

Only one-dimensional cut-downs are considered in the final version of the program package described in Chapter V.

4.3. Cut-Down Assignment

The modification of the LP problem for cut-downs described in the previous section leads to a solution which

satisfies all the requirements and constraints and which may or may not imply cut-downs. The distribution of the cut-downs cannot be seen directly from the solution. The decision about an assignment of cut-downs is a routine and, sometimes, rather laborious task. Performing this task by computer is an obvious choice.

It has been realized that the assignment of cut-downs can be formulated as a separate LP problem. All the possible cut-downs would play the role of unknown variables. The discrepancies between the number of garments required and suggested by the original solution would become right hand side values of the equality constraints. All objective function coefficients would be equal to one and the objective function to be minimized would be the total number of plies to be cut down.

This idea was abandoned after a few experiments. The reason was that from the production viewpoint the number of cut-down cases (bundles) rather than cut-down plies should be minimized. It is not possible to formulate a relevant LP problem and it is not easy to solve this problem by another optimization method and therefore a simple allocation algorithm is used.

The algorithm takes one cut-down group after another and compares the number of garments of each size in the solution with those originally required (i.e. with original constraints before cut-down adjustment). It evaluates the

surpluses and deficiencies (if there are any) in each size within a cut-down group. It picks up the largest surplus and allocates it to relevant deficiencies, then the next largest surplus, and so on, until all the surpluses are exhausted and all the deficiencies are made up for (this will always be the case once the solution satisfies all the constraints).

On the following table the cut-down allocations are indicated by single circles (donors) and double circles (recipients). The data are taken from the example discussed in Chapter V and shown in Appendices E, F and G.

[illegible]

CHAPTER V

FINAL SET-UP OF THE PROGRAM PACKAGE

5.1. Cyber 74 Version

The final version of the program package for cut order planning was originally developed and implemented on the Cyber 74 computer. The block diagram of the components of the package and their interaction is given in Figure 4.

The functions of the components are as follows:

- LPM2: Master program which serves to interact selection of requirement file and marker bank. It calls the subroutines LPG4 and LP8. After the linear programming problem is generated and solved the LPM2 generates and prints a standard production report.
- LPG4: Linear programming problem generator. It reads the information on sizes/style requirements and fabric stock constraints from a selected requirement file, and the information on the characteristics of regular markers and pattern areas from a marker bank. It creates the nonbasic part of the L.P. matrix, objective function coefficients, and right hand side values (including dummy markers). It also adjusts the matrix and RHS values for cut-downs,

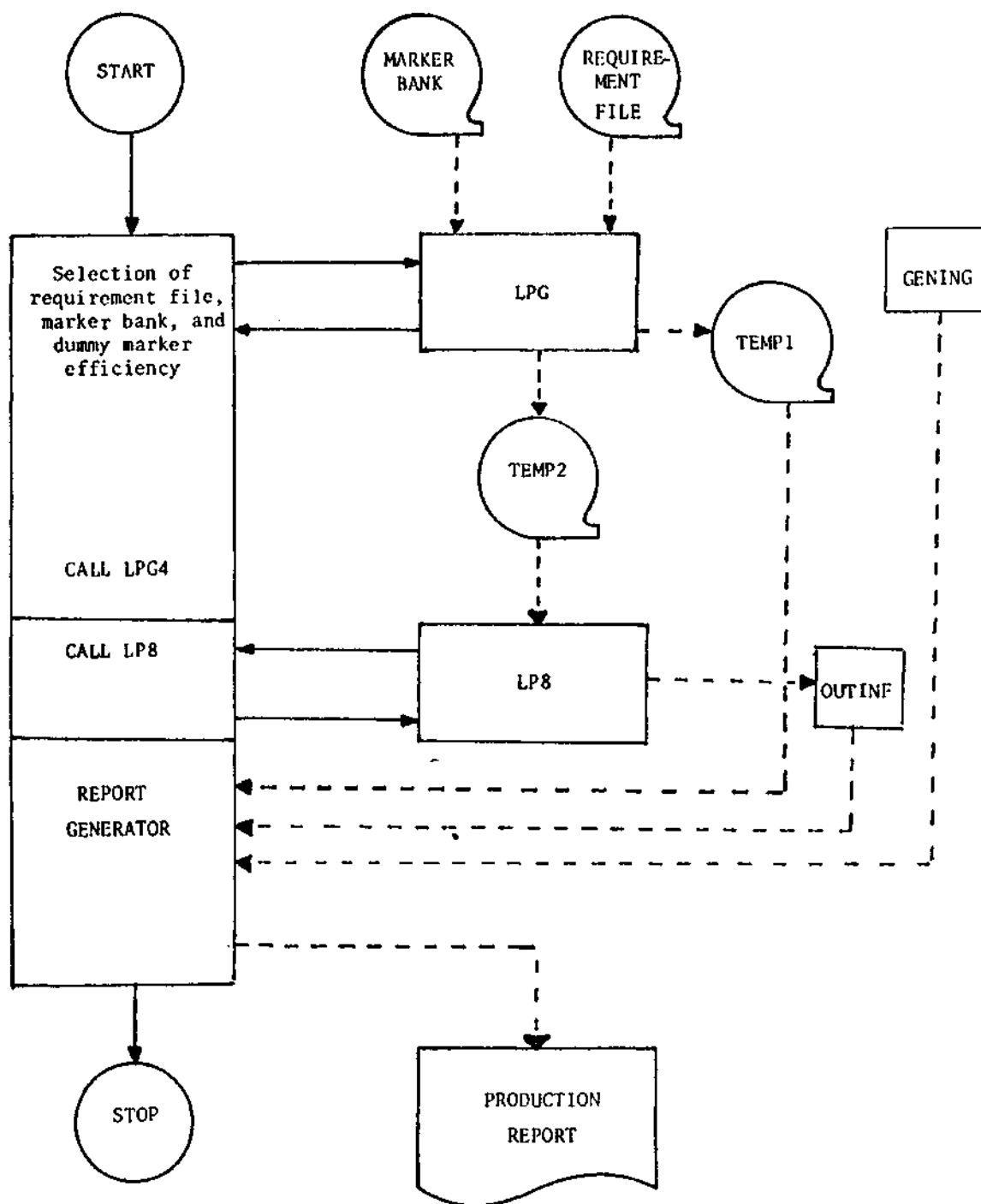


Figure 4. Block Diagram of Cut Order Scheduling Program Package

adds the fabric stock constraints, and writes the LP problem in string representation into the file TEMP2; the original matrix (without dummy markers, cutdowns and stock constraints) is written into the file TEMP1 for future use by LPM2 when generating the production output. More information on marker and size names, etc. for the same purpose is transferred via common block GENINF.

LP8: It is a subroutine built on the basis of the program LP7 discussed in Chapter III. It reads repeatedly the information on the fundamental part of the LP problem from TEMP2 and it solves the series of the real value problems as directed by the branch and bound algorithm (taking into account additional constraints at every branch and bound iteration step. It uses a common block OUTINF for communicating the results of the solution back to the LPM2.

Illustrative examples of input files, temporary files and a production report are given in Appendices E, F and G. The marker bank (Appendix E) consists of two segments. The first contains the values of pattern areas in square yards (second number in each line, i.e. 1.4389, 1.4708, ...) for every size (size identification, first number in each line, i.e. 2926, 2927, ...). This segment is terminated by /. The second segment is divided into subsegments (one for every

regular marker) separated by /END. In the first line of each subsegment is the marker identification (e.g. DUA900201), style name (DUA), fabric width in inches and marker length in yards. Each of the following lines of the subsegment gives size identification and the corresponding number of sets of patterns in the marker. The marker bank is terminated by &. The format of the marker bank is temporary and it is supposed to adjust to local conditions of the up-keeping of the marker banks.

The requirement file (Appendix E) consists of the name line (CT...), style identification (ST...), garment requirements (SZ, size identification, number of garments required, optional Y if the cut-down to the previous size is allowed) and fabric stock constraints (PG, width of the fabric in inches, linear yards of the fabric available). The requirement file is terminated by /.

Temporary file TEMP1 consists of three segments: right hand values (constraints), problem matrix coefficients in integer, dense matrix representation, and identifiers of the kind of constraints. The first two segments reflect the situation before adding the dummy markers, cut-down adjustments and adding the fabric stock constraints. In our example we have 21 size-quantity constraints and 13 unknown problem variables (markers, selected as compatible with requirement file). Correspondingly, there are 21 values in both 1st and 3rd segments and $21 \times 13 = 273$ values in the second segment.

In the temporary file TEMP2 the first line gives the file name and in the second line there are three integers: total number of constraints (in our example 21 for size-quantity requirements and 3 for fabric-widths limits = 24), total number of problem variables (13 regular markers and 21 dummy markers = 34) and number of regular markers (13). In the next six segments there is full information on the problem: objective function coefficients (34), right hand side or constraint values (24), pointers LQ to the last-row-in-column matrix coefficients (34), string of non-zero matrix coefficients Q (114), string of row indices IQ (114) and identifiers of the type of constraints (24).

The printout of the USER-computer conversation when running the program LPM2, including a production report, is given in Appendix G. The production report is self-explanatory.

5.2. HP Version and Industrial Implementation of the Package

The systems for computer-aided marker making are usually based on and built around a dedicated mini-computer. It is expected that the cut-order planning system will be implemented as an additional function of the marker-making system and it will reside in a mini-computer used for this purpose. One reason for this is that cut order planning will use directly the bank of markers developed and

accumulated by the marker-making system.

Therefore the cut-order planning program package developed on Georgia Tech Cyber 74, and described in Section 4.3 of Chapter IV, was transferred to a minicomputer HP21MX. The bulk of the FORTRAN source remains the same although there were changes due to:

- (a) differences in Cyber and HP Fortran dialects and word length,
- (b) entirely different file writing and reading conventions and procedures
- (c) limited storage space on HP

The HP version was divided into 5 modules (Figure 5), controlled by an external monitor, the function of which is to activate (overlay) the proper module depending on the control card (user input).

The communication of the modules is accomplished through permanent disc files, as shown in Figure 6. The information in these files is stored by style and cut identification (user input) on a first in-first out basis. The access to the information related to a particular cut identification (style) is accomplished by supporting directory files.

The above data structure provides the user with a direct access to any of the modulus; therefore repetition of any previous performed operation will require a call only to the module associated with the last phase of the operation.

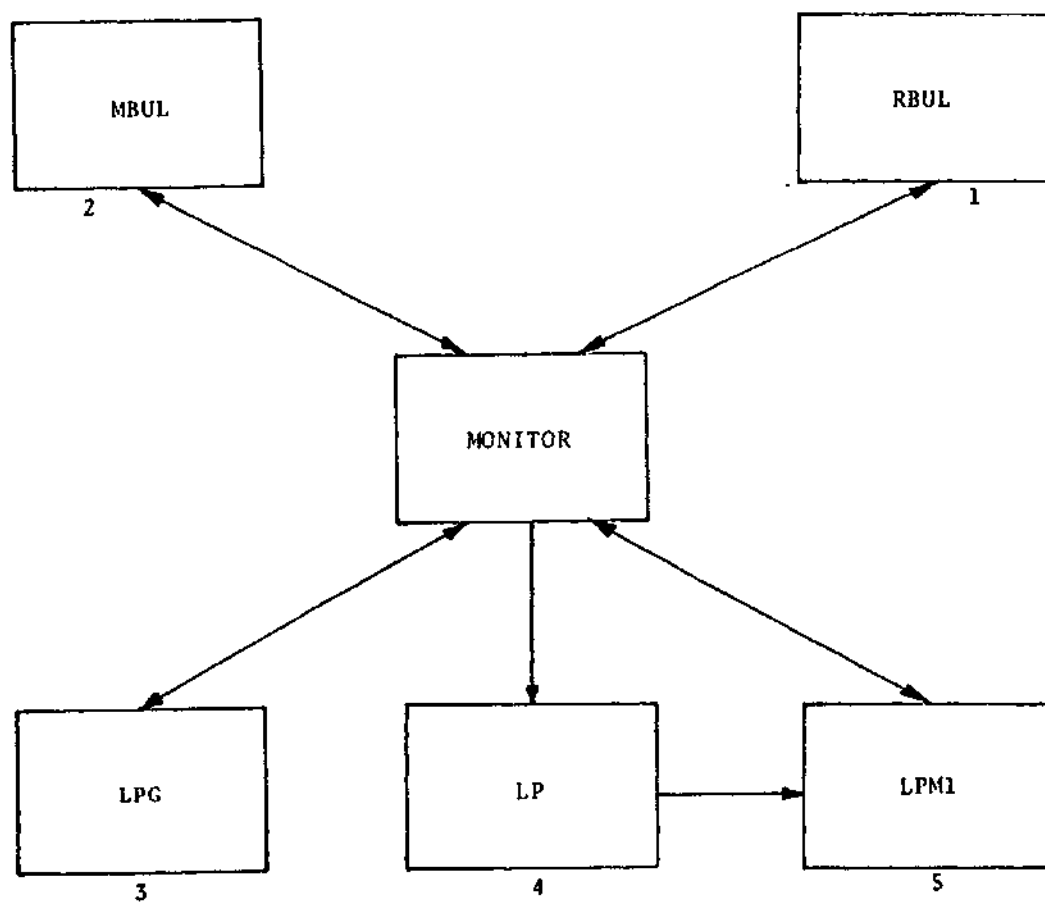


Figure 5. Relationship of the Five Modules of the HP Version of the System

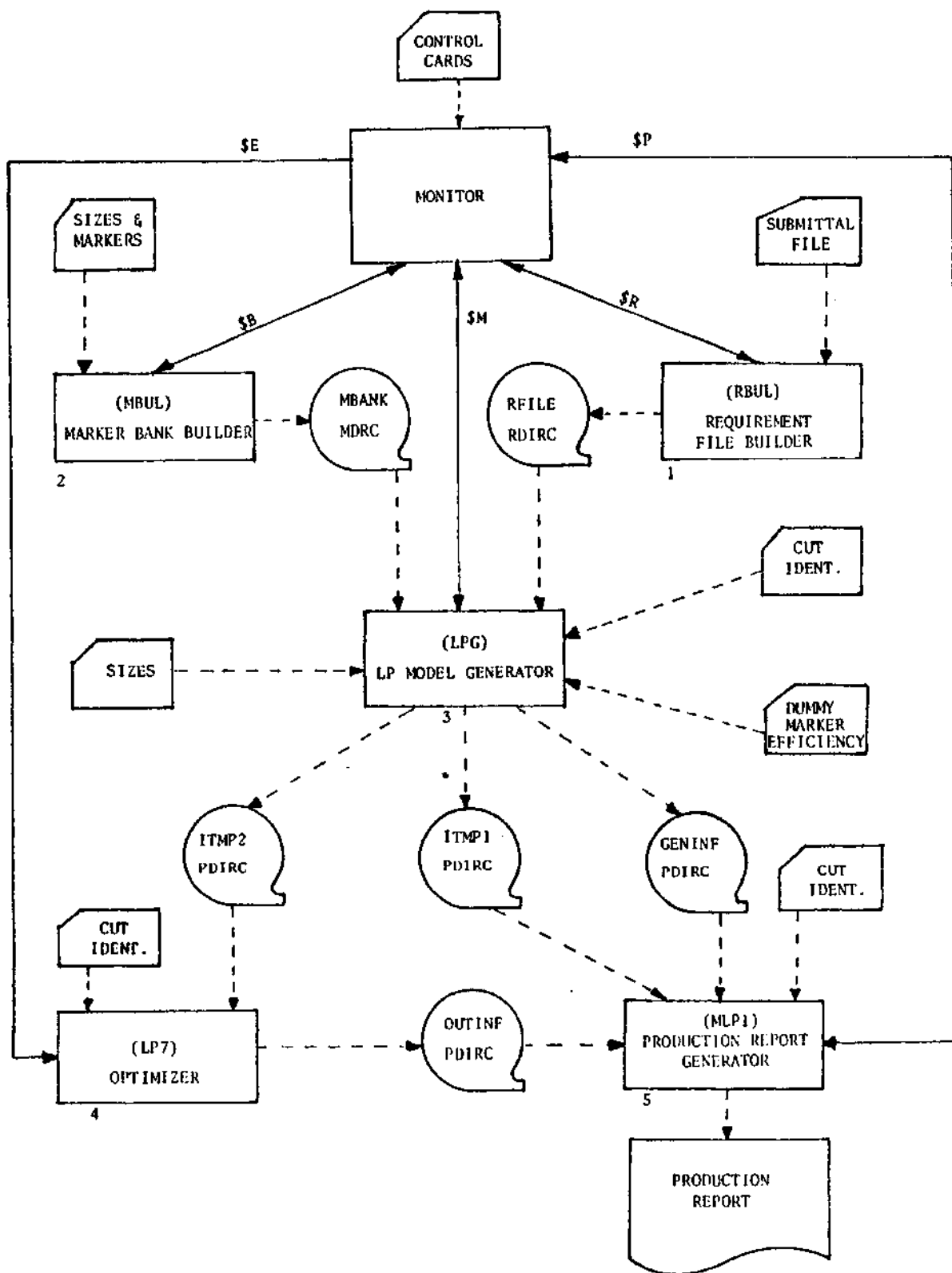


Figure 6. Program Relationship and File Organization of the System

The process is decomposed with respect to time and therefore all the phases can be executed independently. For instance, the user enters a particular submittal file and he generates an LP model with a given dummy marker efficiency. In the sequence the problem is solved and the associated files are updated. The user can resolve the problem for a different dummy marker efficiency by directly accessing module 3 (LPG). The execution of the program goes through the following five phases:

Phase: 1

Program: RBUL--returns control to minitor

Function: Requirement file builder

Control Card: \$R

Description: It accepts as input the submitted file and through RDIRC (directory file) it updates the requirement file (RFILE).

Phase: 2

Program: MBUL--Returns control to monitor

Function: Marker bank builder

Control Card: \$B

Description: It accepts as input the area sites and the marker specifications and through MDRC (directory file) it updates the marker bank (MBANK).

Phase: 3

Program: LPG--Returns control to monitor

Function: LP model generator

Control Card: \$P

Description: LPG operates on RFILE and MBANK. The appropriate markers are selected and introduced as decision variables. LPG interacts with the user to obtain the dummy marker efficiency and any pattern areas missing from MBANK. After the model is built in a matrix form:

- (i) dummy markers are introduced as decision variables,
 - (ii) the module is adjusted by the cut-down specifications,
 - (iii) fabric stock constraints are added to the model,
 - (iv) the two-dimensional matrix is converted to a string by a sparse matrix technique.
- Finally the model in string form and the supporting tables are stored in permanent files through PDIRC (directory file).

Phase: 4

Program: LP8--Passes control to Module 5 (LPM1)

Function: LP Optimizer

Control Card: \$E

Description: The optimizer is an Integer Linear Program algorithm which solves the model. The solution and the supporting tables are stored in the permanent file OUTINF through PDIRC.

Phase: 5

Program: LPM1--Returns control to monitor

Function: Production report generator

Control Card: \$P

Description: Format specifications for input and control cards.

The following commands are available on the HP version.

1. \$REQUEST
2. \$BUILD
3. \$MODEL
4. \$EXECUTE
5. \$PRINT

Each of these commands directs the monitor to activate the proper module respectively. Certain data must be entered after the command is accepted.

The following two-character mnemonics are entered in the first two columns.

CT	Cut identification
ST	Style
SZ	Size
PG	Piece goods

The format of the data entered after the commands is shown below. The following conventions are used.

- (i) Brackets indicate that the information contained is optional,
- (ii) Parentheses indicate that the information

- contained is a numeric entry,
- (iii) Angled brackets indicate that the information contained is a name,
- (iv) The character b indicates a blank,
- (v) /E terminates the current input stream.

1. \$ R[EQUEST]

CT <cut identification>

SZ <size name>, (quantity)[, Y or, N]

PG (width), (quantity)

/E

2. \$ B[UILD]

ST <style name>

<size name>, (area)

<size name>, (area)

.

.

.

/E

<marker name>, (width), (length), [(quantity)*],

<size name>, [(quality)*], <size name>, ... /

<marker name>, ... /

.

.

.

/@

/E

3. \$ M[ODEL]

CT <cut identification>

4. \$ E[XECUTE]

CT <cut identification>

5. \$ P[RINT]

CT <cut identification>

CHAPTER VI

CONCLUSION

The existing systems for computer-assisted marker making facilitate the development and accumulation, over the period of its operation, of a large number of markers for various garment styles, size distributions, fabric widths, and production conditions.

The preparation of a set of markers for every new cutting order typically includes:

- (a) manual selection of those of the old markers which can cover a part of the order;
- (b) marking the rest of the order.

The natural limitations of human ability to process large quantities of information impedes manual retrieval and implementation of old markers in new situations. Consequently, the results of past efforts are not fully utilized. The demands on new markers may exceed available operator and system capacity, and the resulting distribution of the order in sizes and quantities, over the set of markers and over the fabric in stock is not as good as it could be.

The developed linear programming system and supporting computer program package for cut order planning performs a large part of the evaluation and decision making which leads

to the optimum proportion of (a) and (b) above. The main features of the system are as follows:

- retrieving a marker bank and selecting all the existing markers compatible with a particular cut order;
- formulation and solution of a linear programming problem which gives an optimum composition of the existing markers from the viewpoint of fabric utilization or overall operational costs;
- introduction of a concept of "dummy marker", i.e. an imaginary one-size-one garment marker with assumed efficiency: by varying this efficiency one can cover larger or smaller proportion of the new order by existing markers depending on current priorities;
- provision for cut-downs: system automatically adjusts the required size distribution according to the indicated cut-down options and the output information includes detailed instructions for cut-downs from particular markers and sizes;
- provision for complying with indicated fabric stock constraints: system performs optimum distribution of the order considering the available quantities of the fabrics with different widths;
- flexibility in editing the input information in response to intermediate solutions: if the solution violates some obvious but difficult-to-define

limitations, only a part of the solution may be accepted. The original requirements are then adjusted accordingly so that the computer can offer a modified solution, covering the remaining part of the order;

- modular design of the system allows for modification and expansion in compliance with various local conditions and constraints different from those mentioned above.

During the research a series of conversational computing programs for solving linear programming problems has been developed and used. Two of these, LP4 (basic program, dense matrices) and LP6 (sparse matrices, integer programming option) represent a separate by-product of this research and may be easily used for educational and research purposes as shown by Syen [11]. The programs offer all the essential features of interactive computing systems, such as complete freedom and flexibility in manipulating given data (i.e. inserting, altering, or removing variable, constraints, and matrix coefficients.

- Simple means of checking the status of the conversation by displaying current formulation of the problem.
- Means of preventing the corruption of the conversation and basic diagnostics indicating the user's errors and ways of their correction.

- Several levels tracing the iteration procedures for diagnostics and other purposes.
- Simple means of preserving current status of the conversation by writing all the information on the linear programming problem into a data file with an option to recall it whenever necessary.

CHAPTER VII

RECOMMENDATIONS

The results of the research do not solve all the problems of cut order planning as they arise in the real industrial environment. There are two groups of problems future research should be directed towards.

First, there are many additional considerations besides the fabric utilization which in the present cut order planning system may be taken into account only by reformulating and re-entering the problem by the human operator. In the next stage of the research or during the implementation of its results in the industry, some of these considerations should be incorporated in the system so that they would be taken care of automatically. They typically include the following.

The number of markers necessary to satisfy a particular cut order should be minimized for two reasons. Obviously when fewer markers are created, the costs of marking decrease along with the storage requirements in the computer. The second reason the number of markers created should be minimized (or number of garments per marker maximized) is that the larger "bundle size" (garments per marker) usually results in better efficiency. This holds true up to a certain

point and then the efficiency levels off.

In conflict with attempting an optimal efficiency in the way stated above, the mechanics of "spreading" or laying out the fabric are such that a minimum of "two-bundles" (two garments per marker) in a spread (total length of fabric on the cutting table) are required.

The pile height, or number of layers of spread fabric, chosen should be as large as possible to reduce the cost in cutting and the table space requirements. This also keeps down the number of markers needed. However, the pile height is restricted by the cutting device used, yarn characteristics (for example textured yarn reduces pile height allowed) and fabric construction (woven or knitted).

Another factor which must be considered is the number of cut-downs allowed for each cut order and the degree to which the patterns are allowed to be cut down. Some plants have their own restriction as to how much a particular panel can be cut down because of the extra waste involved, which is not accounted for in computer assisted marker making and in standard marker characteristics. This aspect is, of course, already considered by the developed program which minimizes the total fabric consumption. However, there is also an additional labor cost involved.

The width of fabric is critical for utilization because some of the patterns closely pack together forming large compound panels or strips; if these cannot be marked on and

cut from fabric of compatible width, the cutting waste rises disastrously. On the other hand, ordering, keeping and managing large stock of fabrics with near-to-optimal width distribution is costly and has to be judged on the merits of its potential benefits.

In an ideal situation, the cut order planning system would consider and minimize the total cost of cutting operations and related activities including the cost of the material, rather than to minimize the material consumption only. It is difficult to see how this could be done in the near future. However, at least some of the aspects mentioned above may be incorporated in the present system in the form of additional constraints or penalty functions.

Secondly, the available means of predicting the composition and efficiency of non-existing markers should be incorporated into the system. At some plants and in production of certain apparel products, there is a considerable amount of know-how, accumulated over the period of years, about the ways of combining sizes and styles in a marker. The rules are used for the selection of an efficient composition of styles/sizes entering individual markers, depending on the width of the fabric available, so that the cutting waste does not exceed some given limit. Supposing the rules for the selection of good combinations of sizes for a marker are previously formulated and incorporated in the cut order planning system. In this case the predicted "potential

markers" may appear in the linear programming problem along with regular markers and dummy markers. Considering potential markers with trustworthy value of predicted efficiency, rather than filling all the deficiencies by dummy markers, should increase the power of the system and make more competitive the options for cutting the parts of the order, which are not supported by regular markers.

The introduction and utilization of the concept of potential markers will require setting up additional subroutines which will be called from the LP problem generator LPG4 after the information from the requirement file is entered. These subroutines will select the combinations of style/sizes and calculate the expected efficiency according to given rules and a chosen efficiency limit. These subroutines will have to be created individually depending on the local conditions and type of production.

It is envisioned that automatic or semi-automatic marker-making systems could in part play the role of the specialized subroutines. In the future automatic marker-making and cut order planning would merge into one comprehensive system which would accept requirements in terms of pattern shapes and production plan (plus relevant information on additional constraints), and return a complete set of markers together with the production schedule.

APPENDICES

APPENDIX A

SIMPLE EXAMPLE SOLVED BY XLINEAR

```

XLINEAR
THIS PROGRAM SOLVES LP PROBLEMS.
DO YOU DESIRE ADDITIONAL INFORMATION. (YES OR NO)
? NO
DO YOU HAVE A PROBLEM TO RUN (YES OR NO)
? YES
ENTER THE NUMBER OF CONSTRAINTS (INTEGER)
? 2
ENTER THE NUMBER OF VARIABLES (INTEGER)
? 4
DO YOU DESIRE THAT THE TABLEAU FOR EACH
ITERATION BE PRINTED (YES OR NO)
? NO
ENTER THE OBJECTIVE FUNCTION(INCLUDING SLACKS AND ARTIFICIALS)
? 1.,2.,0.,0.
ENTER THE      2      ROWS OF CONSTRAINTS AS EQUALITIES (DECIMAL)
? 1.,1.,1.,0.,6.
? 2.,1.,0.,1.,8.
0 ITER NO=      0 OBJ FCN=      0.00000
0 REPLACING COL      3 WITH COL      2
0 ITER NO=      1 OBJ FCN=     12.00000
0-----O P T I M A L   S O L U T I O N-----
0ROW  SOLN COL  SOLN VAL      DUAL VAL
      1      2      6.0000      2.0000
      2      4      2.0000      0.0000
      .042 CP SECONDS EXECUTION TIME

```

APPENDIX B

ENTERING, DISPLAYING, SOLVING, STORING,
AND TERMINATING THE SIMPLE EXAMPLE
PERFORMED BY LP4

PROBLEM ? NEW
 NO OF CONSTR ? 2
 NO OF VRBLS? 2
 INSERT 2 COEFF OF OBJECT FUNCTION:
 ? 1, 2
 INSERT KIND OF CONSTRAINT AND VALUE:
 1? 1, 6
 2? 1, 8
 INSERT MATRIX ROW BY ROW, 2 ELEMENTS FOR EACH ROW
 1 ROW ? 1, 1
 2 ROW ? 2, 1

NEXT ? DISPLAY

2 2

OBJECTIVE FUNCTION
 1.0000E+00 2.0000E+00

M A T R I X

1ROW: LE 6.0000
 1.0000E+00 1.0000E+00

2ROW: LE 8.0000
 2.0000E+00 1.0000E+00

NEXT ? SOLVE

PRINT TRACE ? YES

0 ITER NO= 0, OBJ FCN= 0.

0-----O P T I M A L S O L U T I O N-----0

0ROW	VARIABLE	SOLN VAL	DUAL VAL	O F COEFF
1	3	6.0000	0.	0.0000
2	4	8.0000	0.	0.0000

NEXT ? STORE

DATA FILE NAME ? EX LP4

NEXT ? END

.105 CP SECONDS EXECUTION TIME

APPENDIX C

SOLUTION PRINTED BY UTILIZING "ALL"

OPTION OF PROGRAM LP6

/RUNLP6

PROBLEM ? BBMAX

NEXT ? SOLVE INT

PRINT ALL-INT-BEST ? ALL

PRINT TRACE ? NO

0 ITER NO= 3,OBJ FCN= -.160000E+02

0-----O P T I M A L S O L U T I O N-----0

0ROW VARIABLE SOLN VAL DUAL VAL O F COEFF

1 1 2.6667 .100E+01 -3.0000

2 2 2.6667 .100E+01 -3.0000

1 1 -1 3.

0 ITER NO= 2,OBJ FCN= .333317E+06

0-----O P T I M A L S O L U T I O N-----0

0ROW VARIABLE SOLN VAL DUAL VAL O F COEFF

1 1 2.6667 .222E+06 -3.0000

2 2 2.6667 .111E+06 -3.0000

3 6 .3333 0. 1000000.0000

1 -1 1 2.

0 ITER NO= 3,OBJ FCN= -.157143E+02

0-----O P T I M A L S O L U T I O N-----0

0ROW VARIABLE SOLN VAL DUAL VAL O F COEFF

1 1 2.0000 .905E+00 -3.0000

2 2 2.0000 .429E+00 -3.0000

3 3 .2857 .952E+00 -13.0000

1 -1 1 2.

2 3 1 0.

0 ITER NO= 4,OBJ FCN= -.130000E+02

0-----O P T I M A L S O L U T I O N-----0

0ROW VARIABLE SOLN VAL DUAL VAL O F COEFF

1 1 2.0000 0. -3.0000

2 2 2.3333 .450E+01 -3.0000

3 3 0.0000 .950E+01 -13.0000

4 5 3.0000 .500E+00 0.0000

1 -1 1 2.

2 3 1 0.

3 2 1 2.

0 ITER NO= 4,OBJ FCN= -.120000E+02

0-----O P T I M A L S O L U T I O N-----0

0ROW VARIABLE SOLN VAL DUAL VAL O F COEFF

1 1 2.0000 0. -3.0000

2 2 2.0000 .300E+01 -3.0000

3 3 0.0000 .130E+02 -13.0000

4 4 2.0000 0. 0.0000

5 5 2.0000 .300E+01 0.0000


```

1 -1 1 2.
2 3 1 0.
3 -2 -1 3.
0 ITER NO= 2,OBJ FCN= .666654E+06
0-----O P T I M A L S O L U T I O N-----0
0ROW VARIABLE SOLN VAL DUAL VAL O F COEFF
1 1 2.0000 .500E+06 -3.0000
2 2 2.3333 .167E+06 -3.0000
3 5 3.0000 0. 0.0000
4 7 .6667 0. 1000000.0000
5 9 0.0000 0. 0.0000

1 -1 1 2.
2 -3 -1 1.
0 ITER NO= 4,OBJ FCN= -.150000E+02
0-----O P T I M A L S O L U T I O N-----0
0ROW VARIABLE SOLN VAL DUAL VAL O F COEFF
1 1 .3333 .100E+01 -3.0000
2 2 .3333 .100E+01 -3.0000
3 3 1.0000 .100E+07 -13.0000
4 6 1.6667 0. 0.0000

1 -1 1 2.
2 -3 -1 1.
3 1 1 0.
0 ITER NO= 4,OBJ FCN= -.148571E+02
0-----O P T I M A L S O L U T I O N-----0
0ROW VARIABLE SOLN VAL DUAL VAL O F COEFF
1 1 0.0000 .905E+00 -3.0000
2 2 0.0000 .429E+00 -3.0000
3 3 1.1429 .100E+07 -13.0000
4 8 .1429 .952E+00 0.0000

1 -1 1 2.
2 -3 -1 1.
3 1 1 0.
4 3 1 1.
0 ITER NO= 3,OBJ FCN= -.135000E+02
0-----O P T I M A L S O L U T I O N-----0
0ROW VARIABLE SOLN VAL DUAL VAL O F COEFF
1 1 0.0000 .450E+01 -3.0000
2 2 .1667 .500E+00 -3.0000
3 3 1.0000 .950E+01 -13.0000
4 5 1.5000 0. 0.0000

```

1 -1 1 2.
 2 -3 -1 1.
 3 1 1 0.
 4 3 1 1.
 5 2 1 0.

0 ITER NO= 3,OBJ FCN= -.130000E+02

0-----O P T I M A L S O L U T I O N-----0

ROW	VARIABLE	SOLN VAL	DUAL VAL	O F COEFF
1	1	0.0000	.300E+01	-3.0000
2	2	0.0000	.300E+01	-3.0000
3	3	1.0000	.130E+02	-13.0000
4	4	1.0000	0.	0.0000
5	5	1.0000	0.	0.0000

1 -1 1 2.
 2 -3 -1 1.
 3 1 1 0.
 4 3 1 1.
 5 -2 -1 1.

0 ITER NO= 3,OBJ FCN= -.671429E+01

0-----O P T I M A L S O L U T I O N-----0

ROW	VARIABLE	SOLN VAL	DUAL VAL	O F COEFF
1	1	0.0000	.857E+01	-3.0000
2	2	1.0000	.100E+07	-3.0000
3	3	.2857	.186E+01	-13.0000
4	5	9.0000	0.	0.0000
5	9	.7143	0.	0.0000

1 -1 1 2.
 2 -3 -1 1.
 3 1 1 0.
 4 -3 -1 2.

0 ITER NO= 2,OBJ FCN= .857128E+06

0-----O P T I M A L S O L U T I O N-----0

ROW	VARIABLE	SOLN VAL	DUAL VAL	O F COEFF
1	1	0.0000	.476E+05	-3.0000
2	3	1.1429	.952E+05	-13.0000
3	6	0.0000	0.	0.0000
4	7	.8571	0.	1000000.0000

1 -1 1 2.
 2 -3 -1 1.
 3 -1 -1 1.

0 ITER NO= 4,OBJ FCN= .285699E+06

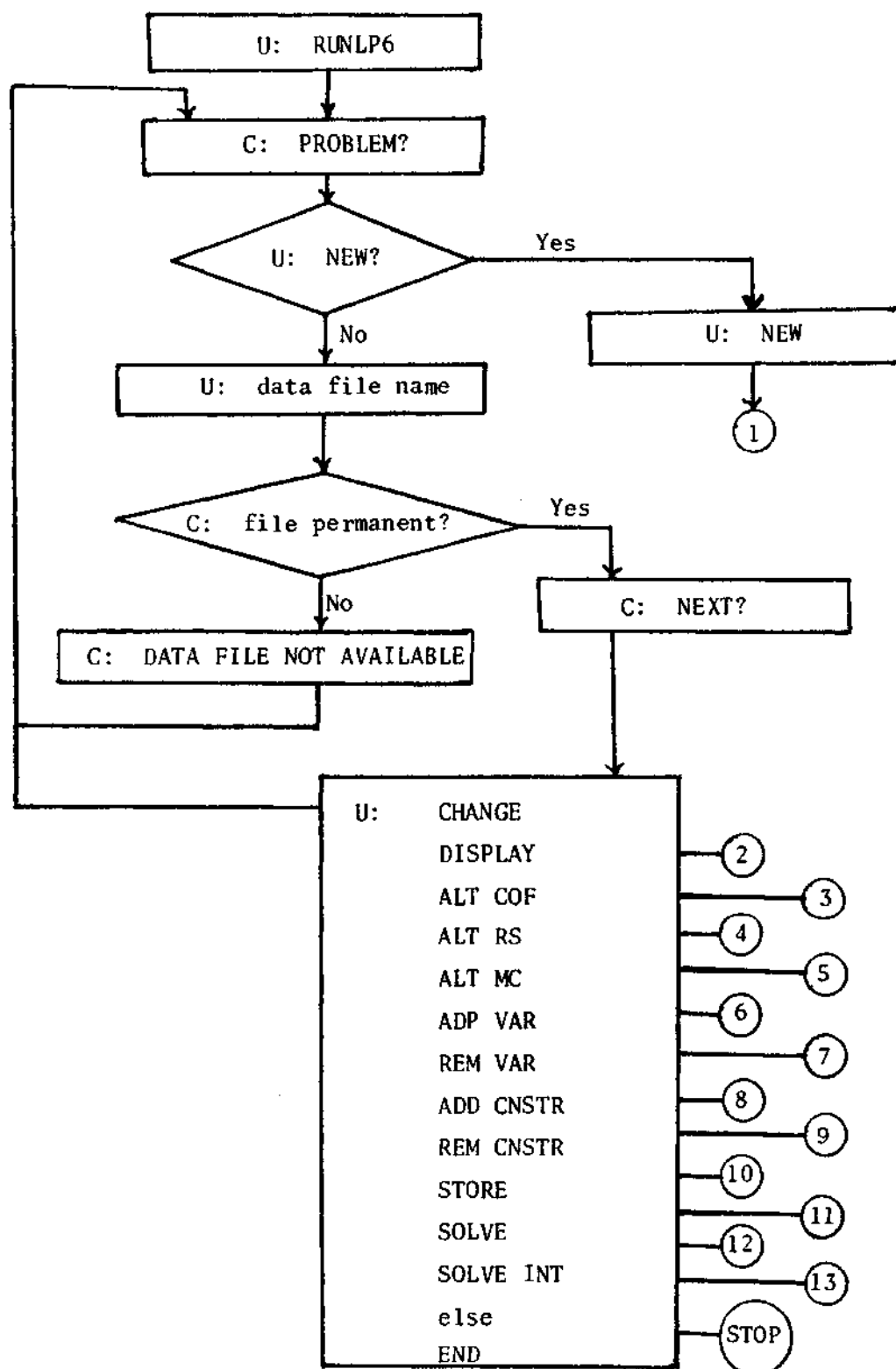
0-----O P T I M A L S O L U T I O N-----0

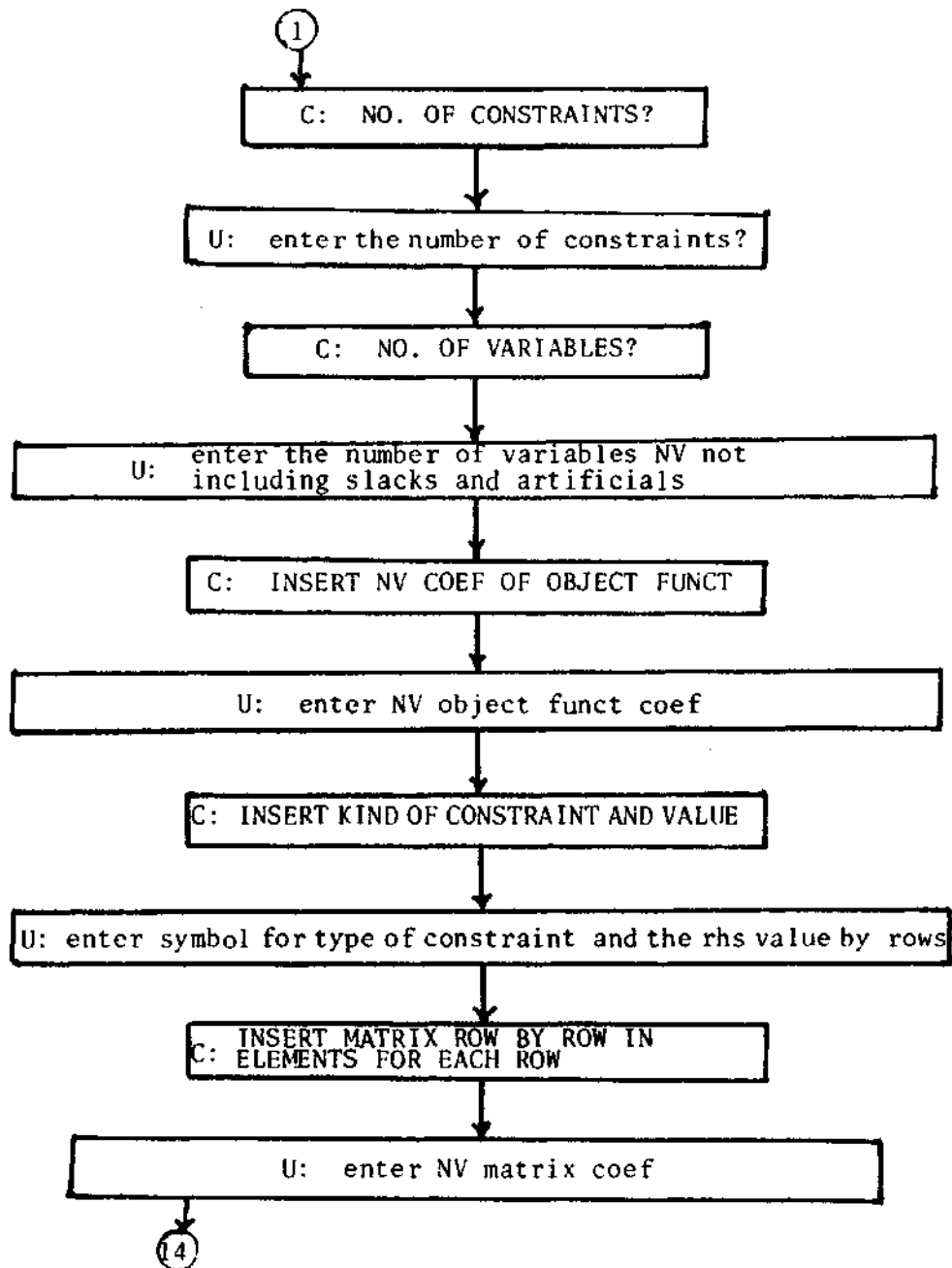
ROW	VARIABLE	SOLN VAL	DUAL VAL	O F COEFF
1	1	1.0000	.952E+05	-3.0000
2	2	1.0000	.476E+05	-3.0000
3	3	.7143	0.	-13.0000
4	8	.2857	.571E+06	1000000.0000

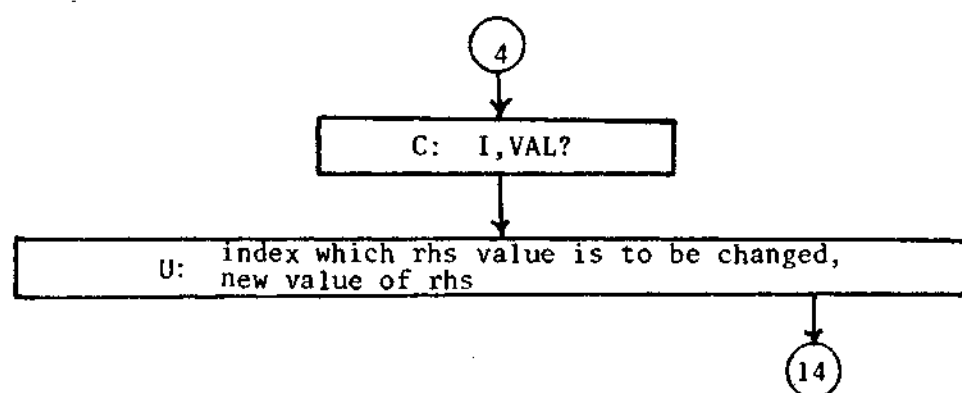
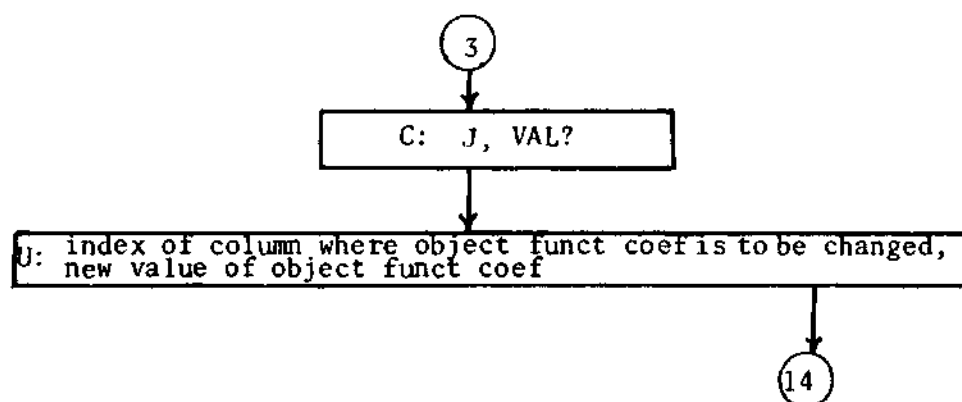
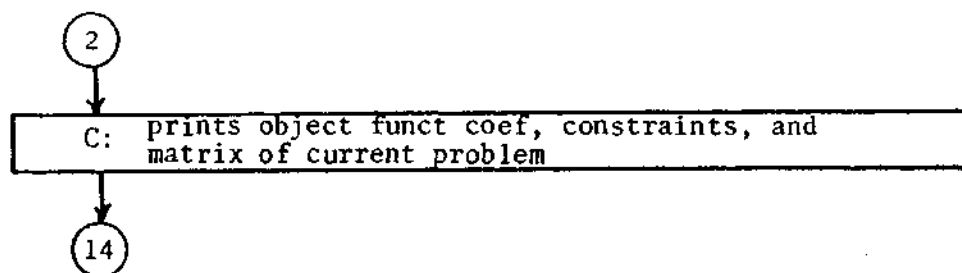
13 REAL AND INT SOLUTIONS

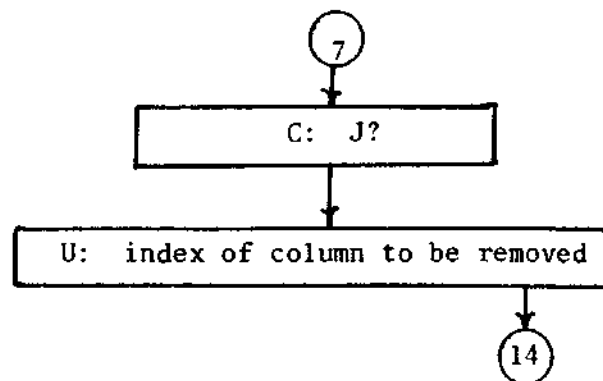
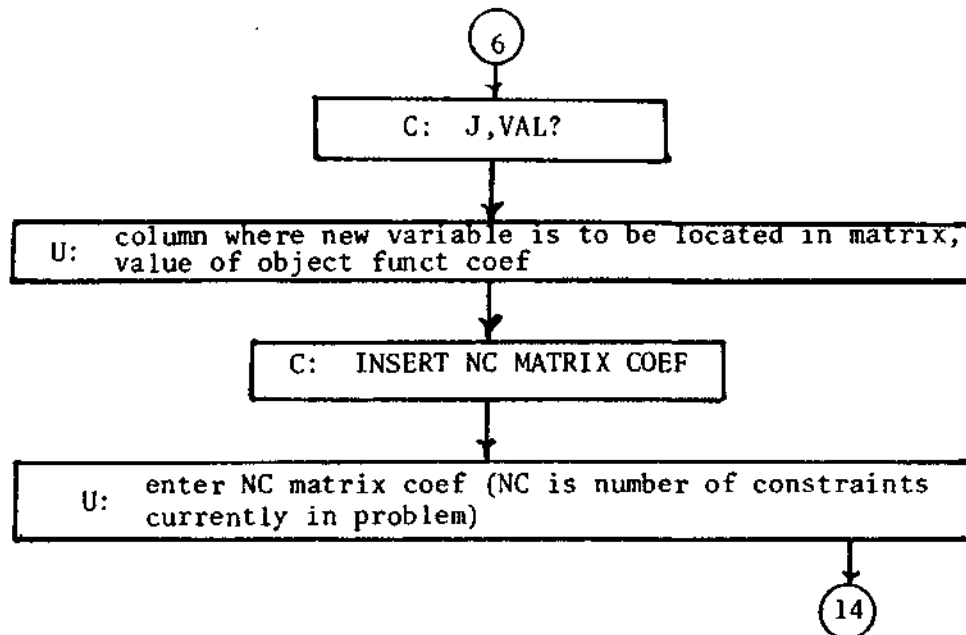
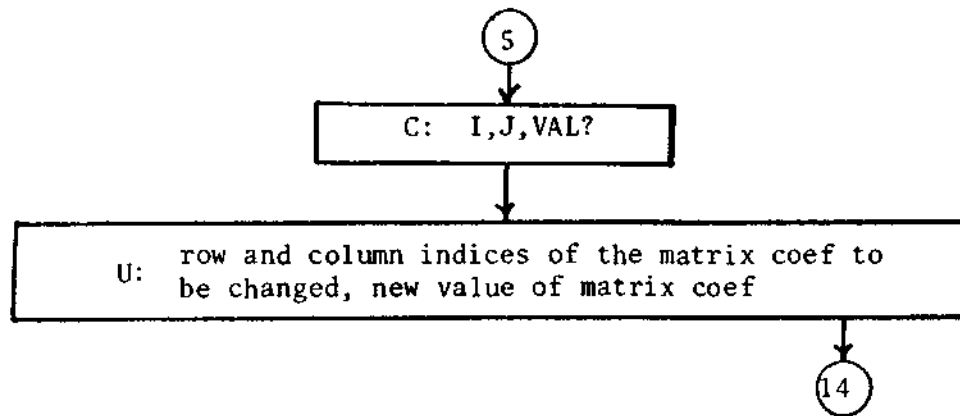
APPENDIX D

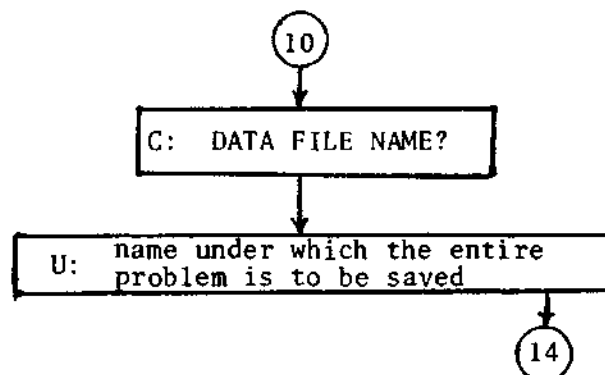
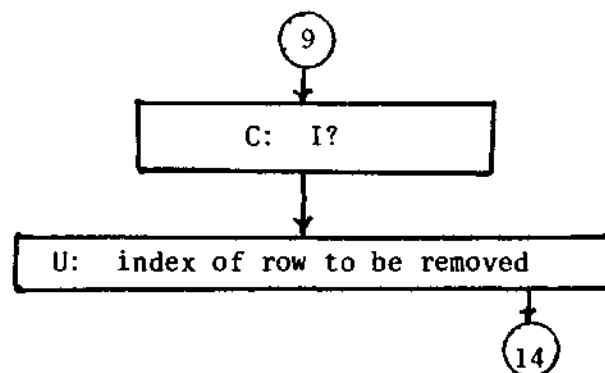
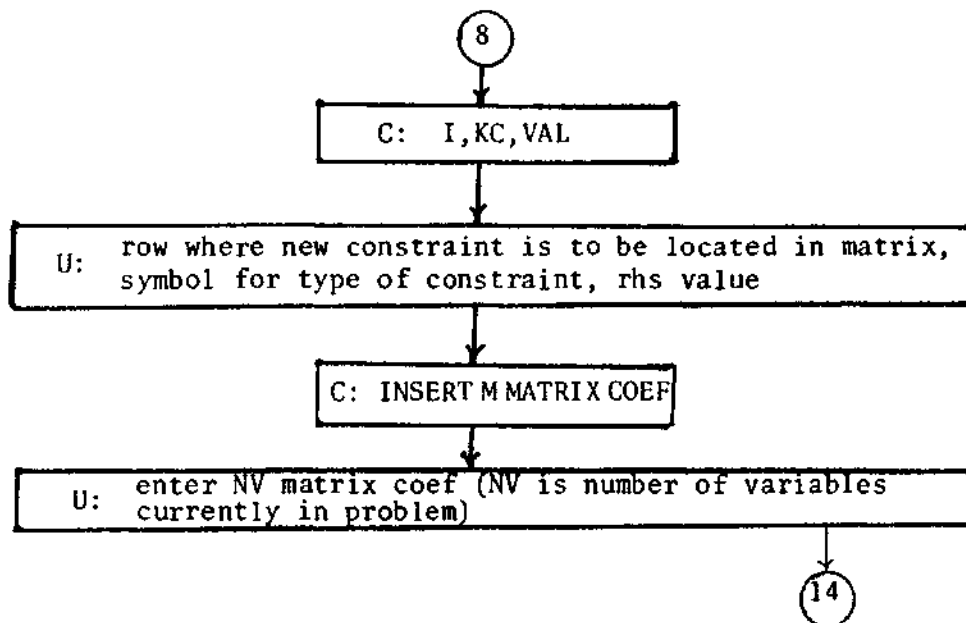
LP6 FLOWCHART

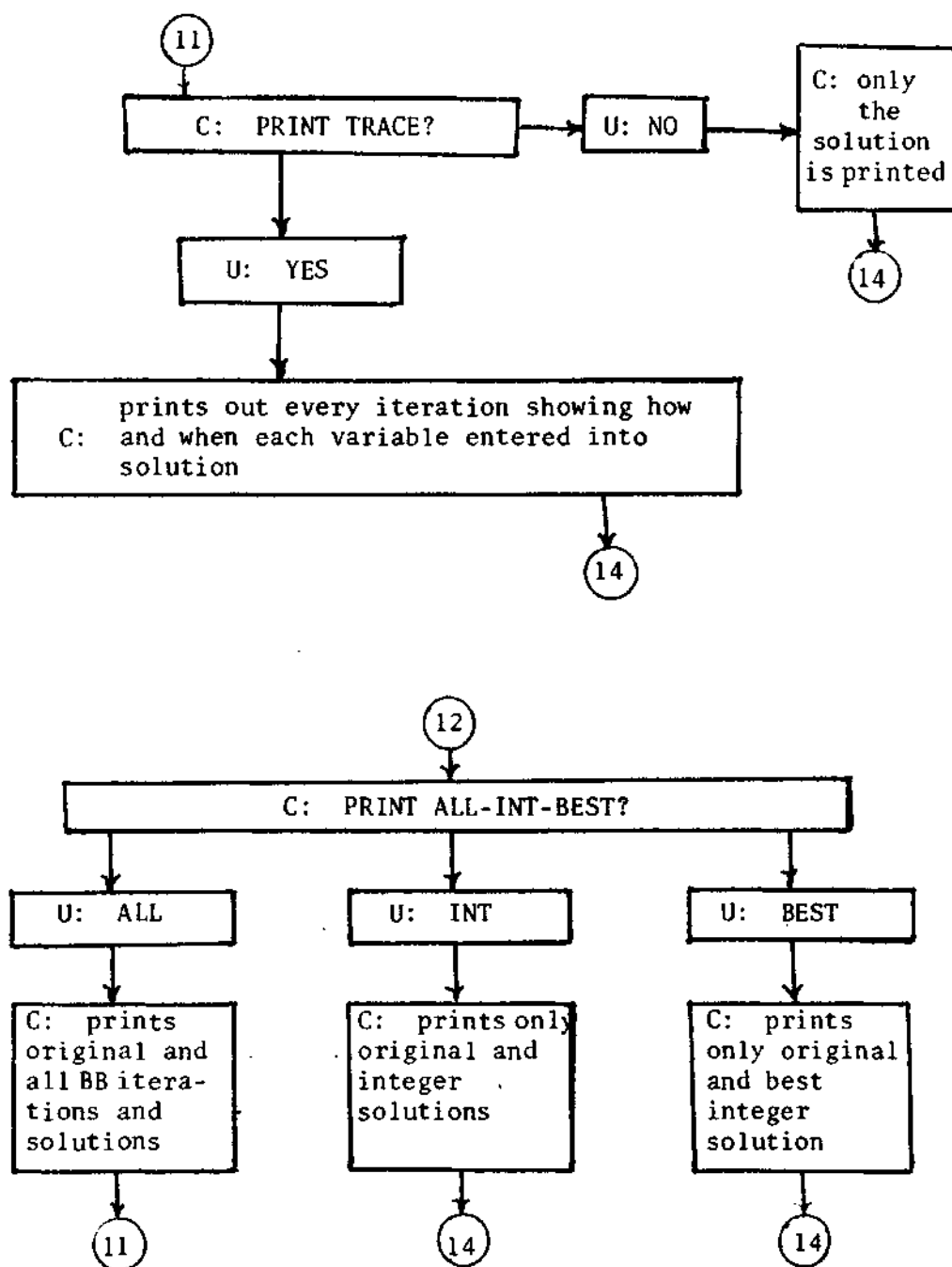


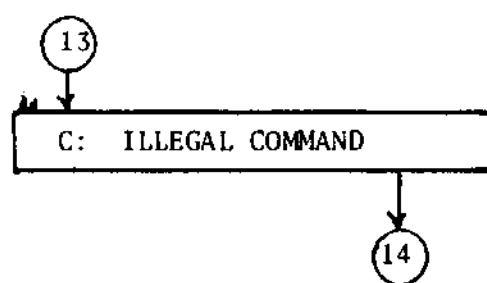












APPENDIX E

INPUT FILES

2926,1.4389	3232,1.7317	3629,1.7719
2927,1.4708	3233,1.7636	3630,1.8038
2928,1.5026	3234,1.7954	3631,1.8356
2929,1.5344	3326,1.5746	3632,1.8674
2930,1.5663	3327,1.6065	3633,1.8993
2931,1.5981	3328,1.6383	3634,1.9311
2932,1.6300	3329,1.6701	3726,1.7103
2933,1.6618	3330,1.7020	3727,1.7422
2934,1.6937	3331,1.7338	3728,1.7740
3026,1.4728	3332,1.7657	3729,1.8058
3027,1.5047	3333,1.7975	3730,1.8377
3028,1.5365	3334,1.8294	3731,1.8695
3029,1.5684	3426,1.6085	3732,1.9014
3030,1.6002	3427,1.6404	3733,1.9332
3031,1.6321	3428,1.6722	3734,1.9651
3032,1.6639	3429,1.7041	3826,1.7442
3033,1.6957	3430,1.7359	3827,1.7761
3034,1.7276	3431,1.7678	3828,1.8079
3126,1.5068	3432,1.7996	3829,1.8398
3127,1.5386	3433,1.8314	3830,1.8716
3128,1.5704	3434,1.8633	3831,1.9035
3129,1.6023	3526,1.6425	3832,1.9353
3130,1.6341	3527,1.6743	3833,1.9671
3131,1.6660	3528,1.7061	3834,1.9990
3132,1.6978	3529,1.7380	3926,1.7782
3133,1.7297	3530,1.7698	3927,1.8100
3134,1.7615	3531,1.8017	3928,1.8418
3226,1.5407	3532,1.8335	3929,1.8737
3227,1.5725	3533,1.8654	3930,1.9055
3228,1.6044	3534,1.8972	3931,1.9374
3229,1.6362	3626,1.6764	3932,1.9692
3230,1.6681	3627,1.7082	3933,2.0011
3231,1.6999	3628,1.7401	3934,2.0329
		/

```

/END
DUA5900215,DUA,59,2.52
3430,1
3432,1
/END
DUA5900216,DUA,59,2.61
3234,1
3534,1
/END
DUA5900217,DUA,59,2.43
3231,1
3331,1
/END
DUA5900232,DUA,59,2.54
3132,1
3333,1
/END
DUA5900233,DUA,60,2.64
3330,1
3931,1
/END
DUA5900234,DUA,60,2.53
3629,1
3732,1
/END
DUA5900235,DUA,60,2.58
3229,1
3634,1
/END
DUA5900240,DUA,60,2.54
3234,1
3628,1
/END
DUA5900241,DUA,60,2.45
3432,1
3627,1
/END
DUA5900242,DUA,60,2.43
3530,2
/END
DUA5900243,DUA,60,2.44
3429,1
3430,1
/END
DUA5900244,DUA,60,2.53
3230,1
3929,1
/END
DUA5900245,DUA,60,2.72
3532,1
3534,1
/END
DUA5900246,DUA,60,2.67
3433,1
3734,1
/END
&

```

```

DUA5900201,DUA,58,2.55
3432,1
3632,1
/END
DUA5900202,DUA,58,2.36
3329,1
3627,1
/END
DUA5900203,DUA,58,3.08
3633,1
3828,1
/END
DUA5900204,DUA,58,2.37
3030,1
3329,1
/END
DUA5900205,DUA,58,2.59
3432,1
3433,1
/END
DUA5900206,DUA,58,2.52
3431,2
/END
DUA5900207,DUA,58,2.65
3234,2
/END
DUA5900208,DUA,58,2.62
3234,1
3528,1
/END
DUA5900209,DUA,58,2.62
3234,1
3528,1
/END
DUA5900210,DUA,59,2.45
3132,1
3628,1
/END
DUA5900211,DUA,59,2.64
3533,1
3829,1
/END
DUA5900212,DUA,59,2.59
3334,1
3626,1
/END
DUA5900213,DUA,59,2.63
2934,1
3734,1
/END
DUA5900214,DUA,59,2.50
3627,2

```

CT OXREQT
ST DUA
SZ 2934,19
SZ 3132,14
SZ 3229,22
SZ 3230,29,Y
SZ 3231,12,Y
SZ 3234,10,Y
SZ 3331,18
SZ 3333,33,Y
SZ 3427,11
SZ 3429,19,Y
SZ 3430,25,Y
SZ 3431,17,Y
SZ 3432,21,Y
SZ 3433,16,Y
SZ 3434,17,Y
SZ 3628,15
SZ 3632,16,Y
SZ 3634,17,Y
SZ 3729,18
SZ 3734,23,Y
SZ 3930,5
PG 58,100
PG 59,150
PG 60,120
/

APPENDIX F

TEMPORARY FILES

TEMP 1

[illegible]

APPENDIX G

PRODUCTION REPORT

RUNLPM2

REQUIREMENT FILE? OXREQT

MARKER BANK ? OXMBSS

DUMMY MARKER EFFICIENCY ? .83

FIRST-BEST INTEGER SOLUTION ? FIRST

3 REAL AND INT SOLUTIONS

MARKAMATIC CUT ORDER PLANNING

76/11/19. RQ FILE: OXREQT

REGULAR MARKERS:

A - ACCEPTED

	MARKER ID	PLIES
1	DUA5900201	16.
2	DUA5900205	12.
3	DUA5900206	8.
4	DUA5900207	3.
5	DUA5900210	14.
6	DUA5900213	19.
7	DUA5900217	18.
8	DUA5900240	1.
9	DUA5900243	19.
10	DUA5900246	4.

B - REJECTED

DUA5900215 DUA5900232 DUA5900235

DUMMY MARKERS:

	SIZE	GMTS
1	3229	22.
2	3230	23.
3	3234	3.
4	3333	33.
5	3427	11.
6	3434	17.
7	3634	17.
8	3729	18.
9	3930	5.

BIBLIOGRAPHY

1. Abrahams, L., Computer Pattern Grading Techniques as Compared to Manual Grading, Proceedings of a Hatra Conference held in Nottingham, England, April, 1974.
2. Bazaraa, M. S. and J. J. Jarvis, Linear Programming and Network Flows, School of Industrial and Systems Engineering, Georgia Institute of Technology, 1974.
3. Camsco: Report to Industry, Richardson, Texas, 1976.
4. Eberly, N. D. and J. E. Troutman, New Technologies for the Cutting Department, AAMA Computer Assisted Apparel Production/Distribution Conference, October, 1975.
5. Gaetan, M., Computerized Marker Making, Bobbin, March, 1973.
6. Gaetan, M., A Trained Computer Makes Markers Automatically, Bobbin, January, 1974.
7. Garfinkel, R. S. and G. L. Nemhauser, Integer Programming, John Wiley & Sons, 1972.
8. Geoffrion, A. M., Integer Programming by Implicit Enumeration and Balas Method, SIAM Review, Volume 9, No. 2, April, 1967.
9. Hrnack, D. M., Single-Ply Cutting Systems, AAMA Computer Assisted Apparel Production/Distribution Conference, October, 1975.
10. Silverman, M., Marker-Making Manual, Michael Silverman, Hull, Mass., 1964.
11. Syen, S., M. S. Thesis, Georgia Institute of Technology, Atlanta, Georgia, 1976.
12. Wagner, H. M., Principles of Operation Research, Prentice-Hall, Englewood Cliffs, N. J., 1975.