TWO-DIMENSIONAL STOCK CUTTING PROCESSES

AN ALGORITHM TO OPTIMIZE TWO-DIMENSIONAL STOCK CUTTING

AT NEW RIVER VALLEY WORKSHOP, INC.

by

Jerome W. Hogge, III

Report submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Master of Science

in

Industrial and Systems Engineering

APPROVED:

Paul T. Kemmerling, Chairman

Marvin H. Agee

Subhash C. Sarin

December, 1991

Blacksburg, Virginia

TWO-DIMENSIONAL STOCK CUTTING PROCESSES

AN ALGORITHM TO OPTIMIZE TWO-DIMENSIONAL STOCK CUTTING

AT NEW RIVER VALLEY WORKSHOP, INC.

by

Jerome W. Hogge, III

Paul T. Kemmerling, Chairman

Industrial and Systems Engineering

(ABSTRACT)

This research addresses a two-dimensional stock cutting problem encountered at the New River Valley Workshop (NRV), Inc. in Radford, VA. This research considers recent literature on two-dimensional stock cutting techniques, pattern generating methods, solution approaches, and practical problem considerations in addressing the NRV cutting problem. Linear Programming analysis is applied to NRV's two-dimensional cutting problem. A computer-based implementation of the LP solution methodology is developed and used to solve NRV cutting problems. Test data and sample problems are used to demonstrate the efficiency, speed and accuracy of the computer-based system. Solution results from test data are compared to current NRV performance, illustrating the significant improvements in solution time and stock utilization that can be attained through the use of LP techniques and this computer-based implementation.

Acknowledgements

This work is dedicated to my wife in gratitude for her love, support and encouragement.

To Paul T. Kemmerling, my advisor, thank you for assistance and guidance in my graduate career at Virginia Tech. I have thoroughly enjoyed knowing you, and I will always remember the things you have done for me.

I would like to extend my sincere thanks to Dr. Marvin Agee for serving on my committee. Dr. Agee provided crucial guidance when I had nowhere else to turn. It has been a pleasure knowing you, and I wish you long life and health.

I would also like to thank Dr. Subhash Sarin for serving on my committee. His work in the field of Linear Programming provided an extensive background on which to base this research. Dr. Sarin's willingness to engage in this research and his support in reviewing the literature were instrumental to the successful completion of this project.

Table of Contents

Table of Contents (cont'd.)

List of Figures

1. Introduction

Improving efficiency continues to be a driving force in today's fiercely competitive marketplace. In the manufacturing industry, increasing raw material utilization and improving manufacturing processes form the basis for maximizing efficiency. Through process automation, the use of advanced equipment, and the application of sophisticated analytical techniques, today's manufacturing facility can outproduce its predecessors with less energy, fewer people, and fewer resources.

Not all manufacturing organizations have taken full advantage of recent advances in manufacturing processes, equipment, and methods to elevate their operation's efficiency and productivity. Among the many techniques available are automation and Mathematical Programming. Automation is a term that has been open to much interpretation, but in this instance is described as the application of computer technology to a task otherwise conducted by a human. Mathematical programming is a collection of analytical tools for modeling and solving manufacturing, process, and many other practical problems.

These two techniques will be applied to a process/manufacturing problem at the New River Valley (NRV) Workshop. Specifically, the process by which NRV processes orders for its products, and the manner by which it allocates its raw material in the production of those

products will be examined. A complete discussion of NRV's operation and a statement of the problem are presented in the following sections.

1.1 Company Description

The New River Valley Workshop, Incorporated in Radford, Virginia, is a small-scale manufacturing facility that employs mentally and physically handicapped individuals in the production of various industrial and commercial products. NRV is a non-profit organization, and as such, receives funding from the Department of Rehabilitative Services, the Department of Social Services, as well as other state and local agencies. The Workshop employs approximately one hundred people, encompasses over 23,000 square feet of manufacturing space, and has outdoor storage totaling approximately five acres. NRV provides products to more than sixty companies and government agencies.

Central to the Workshop's operation is the production of wood strips used to manufacture drawer side and back assemblies. NRV currently operates a single, seven-step production line (Figure 1) to manufacture luaun, pine, and oak drawer sides and backs. NRV manages this production process by manually translating order requests into milling patterns and instructions. This manual operation is time consuming, often requiring six to eight hours of hand

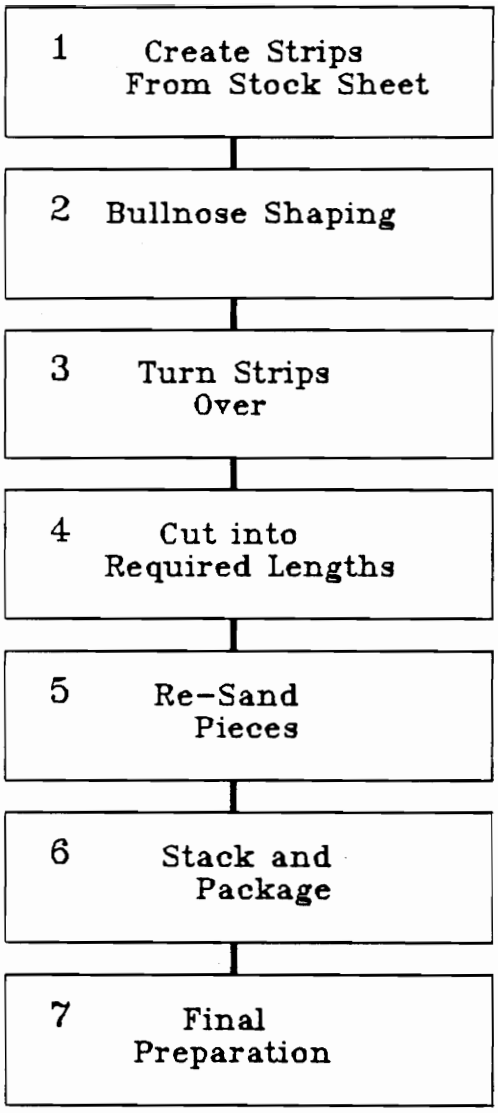


Figure 1 - NRV Seven Step Process

calculation, and has been identified as a priority for improvement. The manual method produces average stock utilization of approximately seventy-five percent which is acceptable to NRV management. While current stock utilization rates are acceptable, any improvement that can be achieved will mean higher profits for the workshop. Industry standards for two-dimensional stock cutting range from a low of 88 percent to a high of 98.7 percent [10] . NRV's goal is to maximize its utilization given its unique set of operating circumstances.

Faced with the current situation, NRV's management has sought to: 1) reduce the amount of time required to convert order specifications into cutting instructions, and 2) improve stock utilization. Accomplishing these two goals requires improvement of the order translation process and the application of more sophisticated analytical techniques in determining cutting patterns.

1.2 Problem Statement

In order for a manufacturing organization to be competitive, it must efficiently utilize its raw materials and follow effective operating procedures. In consideration of these two notions and the aforementioned goals of NRV management, two problems must be addressed. First, in order to expedite its manufacturing process, NRV needs a more efficient means of translating orders into

cutting patterns. Second, to make maximum use of its raw materials, NRV needs to improve the analytical technique by which cutting patterns are determined. Accordingly, the two objectives of this research are: 1) to automate the process by which orders are translated into cutting patterns and instructions, and 2) to apply Linear Programming techniques to the problem of determining optimal cutting patterns.

1.3 Report Layout

This report is organized to present a review and discussion of two-dimensional stock cutting literature. Contained in the review is a general discussion of two-dimensional stock cutting, methods for generating cutting patterns, various solution techniques and practical considerations involved in two-dimensional stock cutting problems (section 2). Following the literature review is an in depth description of New River Valley's cutting process highlighting solution implications, limitations and problems. Also included in this section is a description of the general approach taken in applying Linear Programming to NRV's operation. Concluding this section is a complete specification of NRV's cutting problem, a Linear Programming problem formulation, and a description of the computer-based system that was developed to solve NRV's cutting problems (section 3). Following this section are

test data and solution results (section 4). In the following section, order translation times and stock utilization rates are evaluated and compared to current New River Valley performance (section 5). Recommendations for improving the cutting process and future research are presented in the final section (section 7).

2. Literature Review

2.1 Two-Dimensional Stock Cutting

This literature review was adapted from a paper written by Dr. Subhash C. Sarin [22]. Two-dimensional stock cutting problems involve determining the "best" way to cut stock material into the required numbers, and sizes of smaller pieces. Two dimensions, usually length and width of required pieces are matched against the length and width of the stock for proper cuts (Figure 2). A one-dimensional problem of this nature involves cutting required lengths from a given length of stock material. Two-dimensional stock cutting problems occur frequently and can be found in the glass, paper, fabric, leather, film, plastic, and woodworking industries. Loading trucks, railway freight cars, or pallets is equivalent to two-dimensional stock cutting in that optimal stacking patterns are desired.

The two-dimensional stock cutting problem which is of main interest in the literature and also which occurs quite

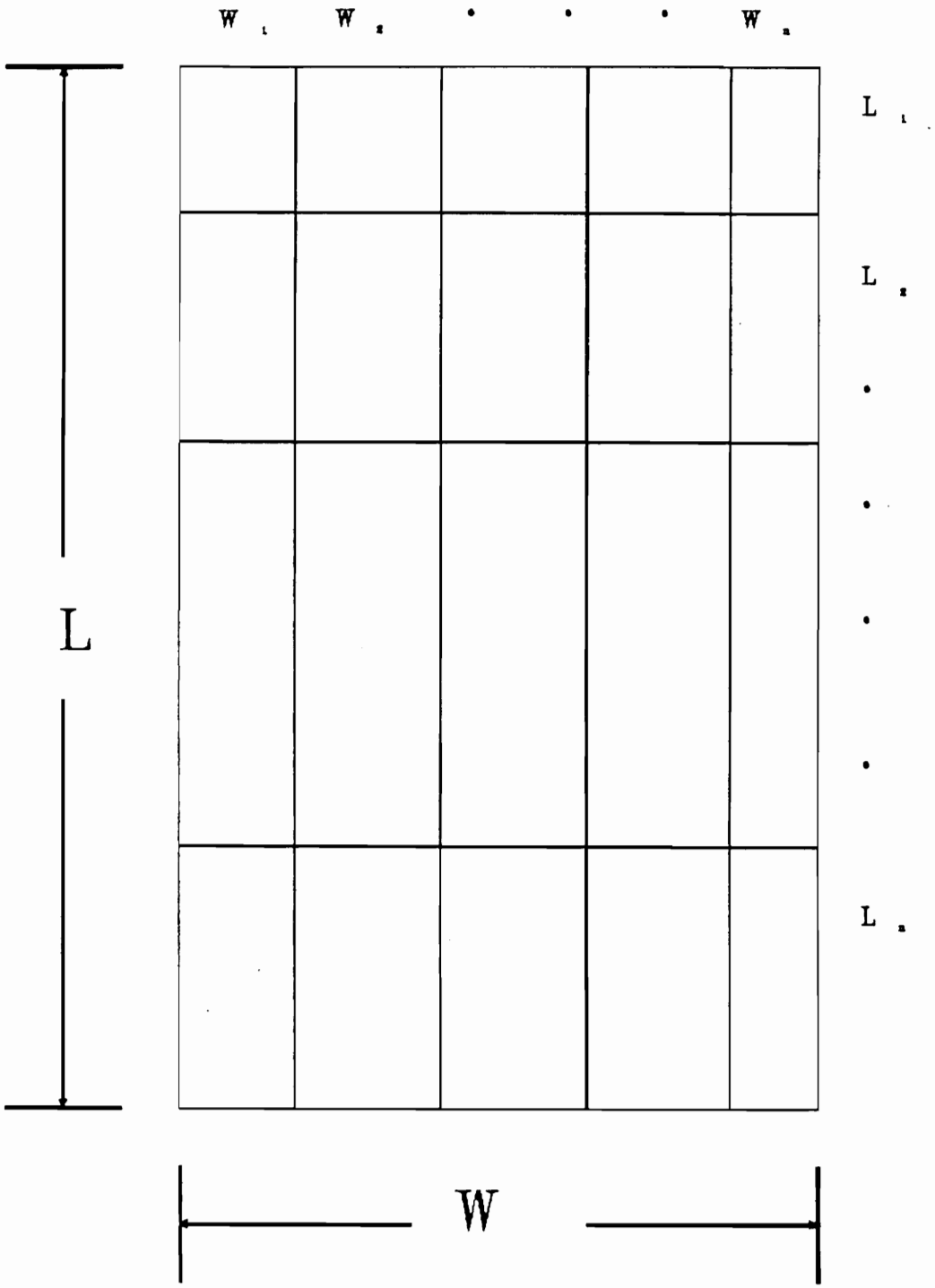


Figure 2 – Two-Dimensional Stock Cutting

frequently in practice, is that of cutting rectangular pieces from rectangular stock. While this problem might seem to be a straightforward extension of the one-dimensional (length cutting) problem, the two-dimensional case presents some unique difficulties. Factors which complicate two-dimensional problems are variations that can be introduced due to practical issues, such as type of material and production process limitations. An evaluation of these concerns will be discussed later in the literature review.

Mathematically stated, the two-dimensional stock cutting problem is to determine how to cut a given number of rectangular pieces of sizes (l_i, w_i), $i = 1, \ldots, m$, from an unlimited supply of rectangular stock of size (L,W). The objective generally chosen is to minimize total waste. However, consideration of various practical issues gives rise to other objectives as well. These include maximization of total value of pieces (where value depends upon the size and quality of the piece), and minimization of time (or cost) of setups required in executing the cutting patterns.

To formulate the problem let,

Q = matrix of feasible patterns (a column of Q is a pattern); a pattern $(q_1, \ldots, q_m)$ represents q_1 pieces of (l_1,w_1), etc.
N = a $(m \times 1)$ vector of demands; N_i = demand for pieces of size (l_1,w_1).

X = a $(m \times 1)$ vector; x_j represents the number of times pattern type j is cut from the stock.
$a_j = (l_i, w_i)$ area of the ith piece, $i = 1, \ldots, m$.

Then for the criterion of minimizing total waste (maximize utilization), the problem is to determine x_j, $j = 1, \ldots, m$ so as to

(A1)

$$\min \sum_{j=1,\ldots,m} x_j$$

$$\text{s.t. } QX = N$$
$$x_j \geq 0, \text{ integer}$$

This formulation is predicated upon the generation of cutting patterns designated as $(q_1, \ldots, q_m)$. In one-dimensional problems these patterns are obtained by solving an appropriate one-dimensional knapsack problem. Knapsack problems seek to determine the most beneficial way of cutting or packing a given quantity of material such that the cutting or packing exactly fits within certain physical dimensions specified in the problem. For a more detailed discussion of knapsack problems see [9, 19]. In the two-dimensional case, generating cutting patterns is complicated by the fact that solving knapsack simplifications of the problem can result in patterns that do not fit within the boundaries of the given stock. The requirement for a pattern to fit within the boundaries of a stock is termed a feasibility requirement. Because of this requirement, the activity of generating feasible patterns for the two-dimensional case is no longer a trivial activity. Before discussing methodologies for solving the

problem presented above, it is necessary to first discuss methodologies for generating feasible patterns.

2.2 Pattern Generation Methods

If v_i represents the value of piece type i, then a general statement of the pattern generating problem is as follows:

(B1)

$$\max\ v_1 q_1 + \ldots + v_m q_m$$
$$\text{s.t.}$$
$$a_1 q_1 + \ldots + a_m q_m \leq A = L \times W$$
$$q_1, \ldots, q_m \geq 0, \text{ integer}$$

An important consideration in generating patterns is the type of cut to be used in cutting each pattern from the stock. A cut which goes from one edge of the rectangular stock to its opposite edge is called the "guillotine" cut. With guillotine cuts, a stock is cut in stages (first into strips of a given width and then into pieces of given lengths). This type of cut occurs in situations where cutting is to be performed by presses or by simple manufacturing lines. The milling process that is used at the NRV facility uses Guillotine cutting as the first step in its process. A more general type of cut that does not go from one edge to the other is called a "non-guillotine" cut. Non-guillotine cuts are possible when cutting is done with a mechanical saw, flame, or with lasers. Additionally, cuts which are made parallel to one edge of the rectangular stock are called "orthogonal" cuts. Orthogonal cuts are used at the NRV workshop. Cuts which are not parallel to any

one edge of the rectangular stock are called "non-orthogonal" cuts. Each of these different types and orientations is illustrated in Figure 3. In consideration of NRV's "Guillotine-Orthogonal" cutting process, it is necessary to explore the methods by which cutting patterns can be created to suit this situation. There are six widely used methods for generating two-dimensional cutting patterns:

1. Two-Stage Method
2. Dynamic Programming
3. Recursive Method
4. Tree Search Method
5. Heuristic Method
6. Interactive Graphics Method

Each of these methods will be discussed in the general context of two-dimensional guillotine cutting. Specific implications for NRV's cutting problems will be discussed in section three.

Two-Stage Method: The first method to be discussed is a technique developed by Gilmore and Gomory [8]. In their work they propose a two-stage procedure for solving pattern generating problems. By assuming guillotine-orthogonal cutting Gilmore and Gomory can represent a two-dimensional problem (B1) as two one-dimensional problems (B2) where optimal length and width patterns are determined independently. In their two-stage process, stock is first cut into strips and then the strips are cut into required

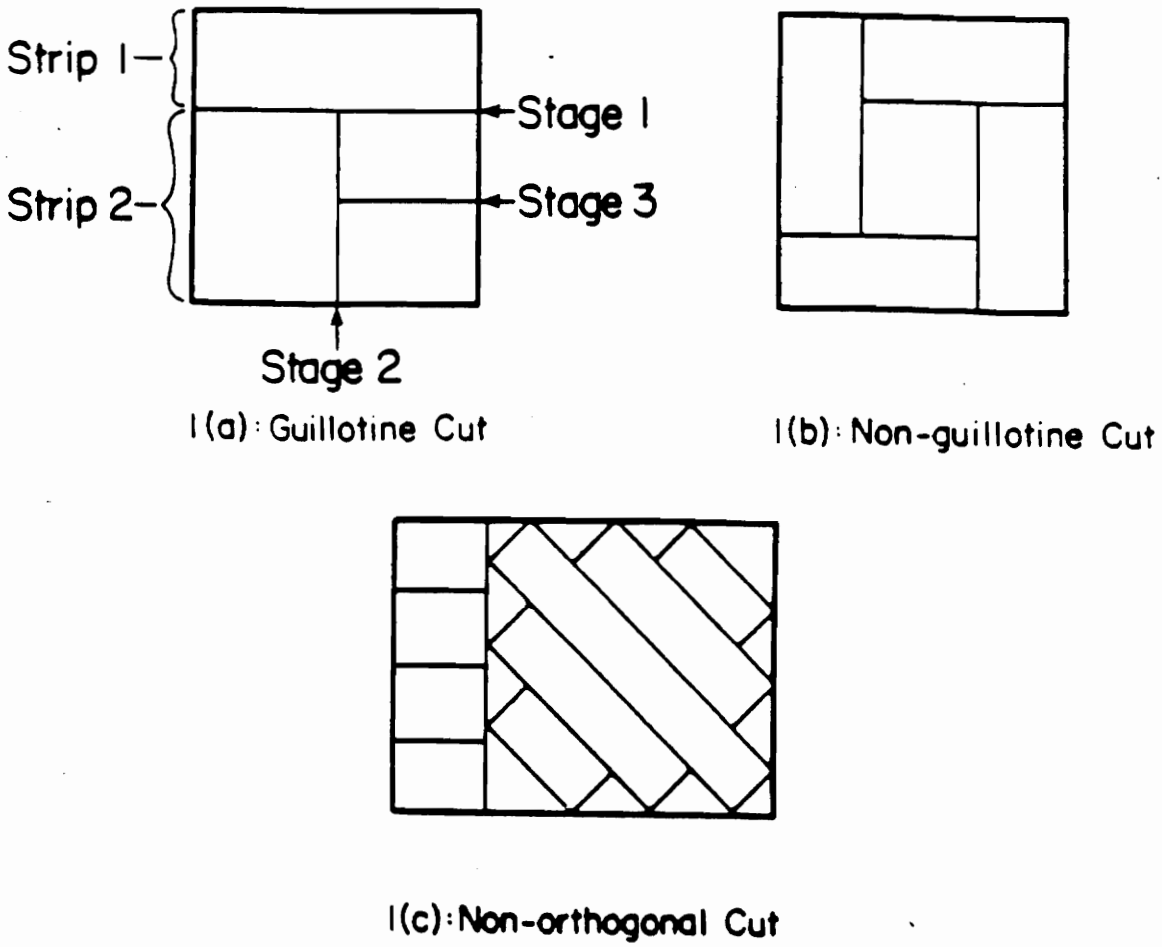


Figure 3 – Types and Orientations of Cuts

pieces. This method is appropriate for the NRV operation because their manufacturing process also first creates strips and then pieces from the strips. To solve each of these one-dimensional problems a one-dimensional knapsack problem is formulated and solved. Mathematically stated, in stage 1, for each width w_i, feasible cutting patterns for fitting rectangles of width $w_j \leq w_i$ in strips of size w_i x L are determined by solving the following knapsack problem:

$$\begin{aligned} &\max\ v_1q_1 + \ldots + v_tq_t \\ &\text{s.t.} \\ &\quad l_1q_1 + \ldots + l_tq_t \leq L \end{aligned}$$

where $l_1, \ldots, l_i$ and $q_1, \ldots, q_t$ are respectively the lengths and number of pieces for which $w_j \leq w_i$, $j = 1, \ldots, t$. In stage 2, a knapsack problem is created to fit strips of size w_i x L into rectangular stock of size W x L so as to

$$\begin{aligned} &\max\ v_1q_1 + \ldots + v_mq_m \\ &\text{s.t.} \\ &\quad w_1q_1 + \ldots + w_mq_m \leq W \end{aligned}$$

Several methods are available for the solution of the one-dimensional knapsack problem [9, 19].

Dynamic Programming: A second technique that has been used extensively in the solution of pattern generation problems is Dynamic Programming (DP) [2, 5, 17]. Gilmore

and Gomory [9] present such a procedure for B2 when there is no limitation on the number of rectangles of one kind that can be cut. If v_i is the value of the ith piece, then the DP recursion equation for two-stage cutting is as follows:

$$
\begin{aligned}
F_0(x,y) &= \max \{0,\ v_i,\ l_i \le x,\ w_i \le y\} \\
F_1(x,y) &= \max \{F(x,y-1),\ F(x1,y) + F(x2,y); \\
&\qquad x \ge x_1 + x_2,\ 0 < x_1 \le x_2\} \\
F_2(x,y) &= \max \{F(x-1,y),\ F(x,y_1) + F(x,y_2); \\
&\qquad y \ge y_1 + y_2,\ 0 < y_1 \le y_2\}
\end{aligned}
$$

where,

$$
\begin{aligned}
&F_1(x,0) = 0,\ F_2(0,y) = 0 \text{ and} \\
&F(x,y) = \max \{F_0(x,y),\ F_1(x,y),\ F_2(x,y)\}
\end{aligned}
$$

Here it is assumed that l_i and w_i are all integers. The implementation of the foregoing recursive equations is straightforward and is carried out as follows. Initially, $x_2 = y_2 = 1$ and they are varied one at a time. For any x_2 and y_2 values, x_1 is varied first starting from $x_1 = 1$. Its value is increased by one, keeping y_2 constant, and the best of $F(x_1,y_2) + F(x_2,y_2)$ and $F(x_1+x_2,y_2)$ is computed. This is carried out until x_1 exceeds x_2 or $x_1 + x_2 > L$ with x_2 fixed at its current value. Then y_1 is varied, starting from $y_1 = 1$, by increasing its value by one, keeping x_2 fixed at its current value, and the maximum of $F(x_2,y_1) + F(x_2,y_2)$ and $F(x_2,y_1+y_2)$ is computed. This process is continued until y_1 exceeds y_2 or $y_1+y_2 > W$. Then x_2 is increased by 1 and the foregoing process is repeated keeping y_2 fixed until $x_2 > L$, at which time y_2 is increased by one, and the whole process

is repeated starting from $x_2 = 1$. The procedure stops when y_2 exceeds W.

Recursive Method: A third method for solving pattern generation problems as represented by B2 is the recursive method. J. C. Herz [14] has developed a procedure based on the recursive property which says: either the large rectangle is already one of the desired rectangles or the first dissection line yields two rectangles, each of which is optimally dissected. This property implies trying every possible first dissection of the large rectangle, R. However, it can be easily shown that: a) only those dissections of R which are integer multiples of lengths or widths of pieces need to be considered because otherwise any other dissection can be reduced to a dissection of the same value by reducing its size to the nearest integer multiples of lengths or widths of pieces. Such a dissection is called a canonical dissection. A dissection involving only one type of small rectangle is called a homogeneous dissection. It can also be shown that: b) for every canonical nonhomogeneous dissection of R, there exists a canonical dissection of equal value, the coordinate of the first line of which is at most equal to half of the corresponding dimension of R. This reduces the number of dissections to be considered by half. The recursive method is better in terms of computation time and storage requirements than

Gilmore and Gomory's iterative procedure [9] described above.

Tree Search Method: A tree search procedure which is similar to Hertz's recursive technique was developed by Christofides and Whitlock [4] for problem B2, under the constraints of limited and known requirement for each type of piece. The tree search procedure proceeds from the "root" of a mathematical tree. At the root node of the tree, the branches are considered to be all possible cuts that can be made on the rectangular stock. A node at the end of each branch represents the rectangles produced by the corresponding cut. At each subsequent node, a particular rectangle is selected to make further cuts. So, a node in the tree represents all the rectangles produced by cuts according to the paths from the root node to that node. Cuts can be made along the length or width of the stock. Cuts are also made such that a combination of lengths or widths of one or more pieces exactly fits in the resulting length or width of the rectangle. These cuts are called normal cuts. The procedure for generating normal cuts is based on the two properties (a and b) presented in the discussion of the recursive method. Also in making these cuts, caution is taken to insure that no two cuts result in the same pattern. Additionally, the option to make no cuts is always available at a node if no cut is desired or

possible. If all rectangles, corresponding to a particular node, are given no-cut status, then no further branching can take place from that node. An elegant feature of this procedure is the determination of bounds. An upper bound on the value of the solution is obtained at every node. This is done by looking at the rectangles that have been designated no-cut and those that have been chosen for further branching. For rectangles which are not to be branched from, a transportation problem [11] routine is used to make the best feasible allocation of pieces to these rectangles. Each of the rectangles which are still to be branched from are analyzed by a dynamic programming (DP) procedure of the type described in an earlier section. The DP determines the best allocation of pieces on the given rectangle by relaxing the constraint on the maximum number of pieces of each type desired. By doing this, the allocation given by the DP may be infeasible, but if it is not infeasible, then the allocation obtained is the best possible and no further branching is required from that node. Otherwise branching continues comparing the current best feasible solution with each branching solution. Nodes with an upper bound value lower than the current best feasible solution are truncated. Various rules can be imposed to direct branching from node to node. The tree branch procedure stops when no more nodes remain to be branched from.

Heuristic Method: Another method for generating patterns is the Heuristic Method. This method is essentially a collection of individual rules by which required pieces are fitted into available stock. Several heuristics are: a) start by packing the largest sizes first, then next to largest and so on; b) start by packing smallest sizes first, then next to smallest and so on; c) start from a corner (usually lower left) and fit pieces in; d) pack along the edge first and move inward or e) pack from the center and move outward. In each of these heuristics, packing simply means fitting the length of a piece into the length of the stock strip until no additional strips can be added. A heuristic procedure based on c) is proposed by Steudel [23] for a version of problem B1 concerning the generation of arrangements for loading pallets. Only one size of piece is considered for loading. The heuristic consists of generating four arrangements along the four edges of the pallet. These arrangements are determined using DP. A piece can be placed lengthwise or widthwise along an edge. Once the arrangements along edges are found, they are built up inwards following these arrangements such that there is no overlap in the center portion. If overlap occurs in the center portion, the depth of inward projections is modified.

Interactive Graphics Method: Another method for generating patterns for guillotine type cutting procedures is the Interactive Graphics Method. As was discussed earlier, the difficulty in solving B2 is the requirement to create feasible patterns. However, if the constraint in B2 is replaced by:

(B3) $$a_1q_1 + \ldots + a_mq_m \leq A - e$$

then it is easy to see that feasible patterns can be generated by solving B3 for some $e \geq 0$ [21]. For rectangular or triangular figures it is possible that e, the error term, could equal zero, but for other figures, e is likely to be greater than zero which limits explicit consideration of many feasible patterns. Various solutions to B3 can be generated by varying e. The feasibility of these solutions can be tested on a graphics terminal and an optimal feasible solution obtained. A feasible solution of B2 is thus determined interactively through the use of a graphics terminal. An advantage of this method is that it is applicable to figures of any shape. Knowledge about the upper and lower bounds on e for the shapes of pieces to be cut can be very beneficial in this interactive procedure. Available bounds on e for some types of figures are given in [20] and an application of this procedure is given in [21].

2.3 Solution Techniques

Linear Programming: The first solution technique to be discussed is Linear Programming. LP was developed in 1947 by George Dantzig in response to a need for a more efficient and expedient military planning tool. LP created a means for optimizing (maximizing or minimizing) an objective while satisfying large sets of linear constraints. LP problems consist of an explicit goal, explicit problem constraints expressed as inequalities, decision variables and problem parameters. In NRV's case, the goal is to minimize the amount of stock material required to fill customer orders for drawer components--or simply stated, minimize waste. The constraints are expressions which relate cutting patterns to the different quantities and lengths of required pieces. In NRV's case, there are also operational limitations which constrain the complexity of feasible patterns. This limitation will be discussed in more detail in a following section. The decision variables are the number of times to use each feasible cutting pattern in the production of order requirements. The parameters of the problem are the number and size of pieces required, pattern complexity and the sizes of available stock.

Linear Programming can be used to solve the problem of A1 (page 8) if the integrality requirements are relaxed. Call this problem A2. When N is large, rounding the x_j values to the nearest integer has little effect on the

objective function value, and thus LP is a good approximation to the integer problem expressed in A1. Therefore, solving A2 is in most cases a reasonable approximation to solving A1.

Problem A2 can be solved by the method of column generation [15] which was presented by Gilmore and Gomory [8] for guillotine cuts. A column $(q_1, \ldots, q_m)$ represents a feasible pattern. If $(v_1, \ldots, v_m)$ represent the simplex multipliers associated with an optimal solution of A2, then a column $(q_1, \ldots, q_m)$ is determined by solving the pattern generating problem B2. Gilmore and Gomory use the two-stage procedure presented in a preceding section to solve B2. Problem A2 can, therefore, be solved by a linear programming procedure, each iteration of which requires the solution of a pattern generating problem B2.

Gilmore and Gomory [8] present another LP formulation of the two-stage guillotine problem which is treated as a two-stage linear programming problem. The first stage corresponds to the process of slitting the stock rectangles into strips with widths corresponding to the widths of demanded rectangles, and the second stage corresponds to the process of chopping the strips into demanded lengths. This LP program can then be solved directly using the approach for one-dimensional stock cutting problems or by applying the Dantzig-Wolfe decomposition procedure [6].

Heuristic Procedures: Heuristic procedures have also been used to solve two-dimensional problems. A general heuristic procedure for problem A1 is as follows.

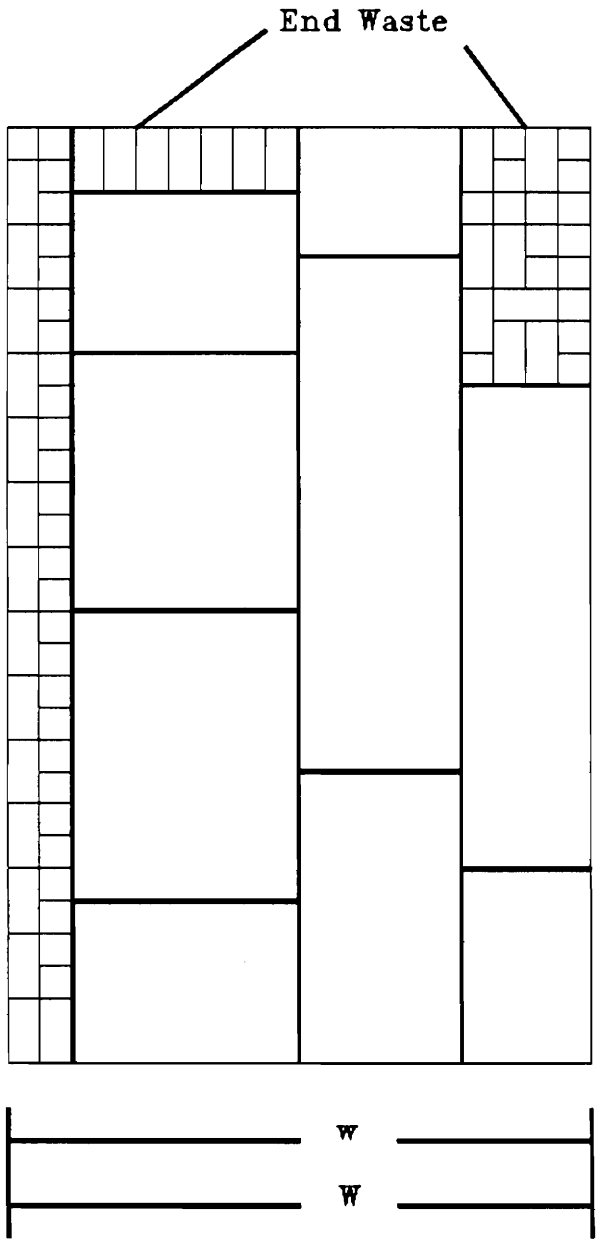
Step 1 : Select a criterion for acceptance of a pattern. One such criterion for a homogeneous pattern to satisfy is based on the percentage of maximum width of stock occupied and the percentage of end waste. An acceptable pattern has to satisfy both the selected percentages of width and end waste. This acceptance criterion is shown in Figure 4. Other appropriate acceptance criteria can be selected for other cuts.

Step 2 : Combinations of pieces of one kind alone are taken and patterns which satisfy the selected criterion are selected, if any.

Step 3 : Combinations of two kinds of pieces are then tried. The patterns which satisfy the selected criterion are chosen.

Step 4 : Combinations of three kinds of pieces are tried next and the patterns which satisfy the selected criterion are chosen. Higher order combinations can be tried, if desired. An example of a homogeneous pattern involving three kinds of pieces obtained as a result of this procedure is shown in Figure 5.

Step 5 : If the best patterns obtained cover all required pieces, stop; otherwise relax the criterion and go to step



w/W = Width Criterion

Figure 4 – Heuristic Acceptance Criterion

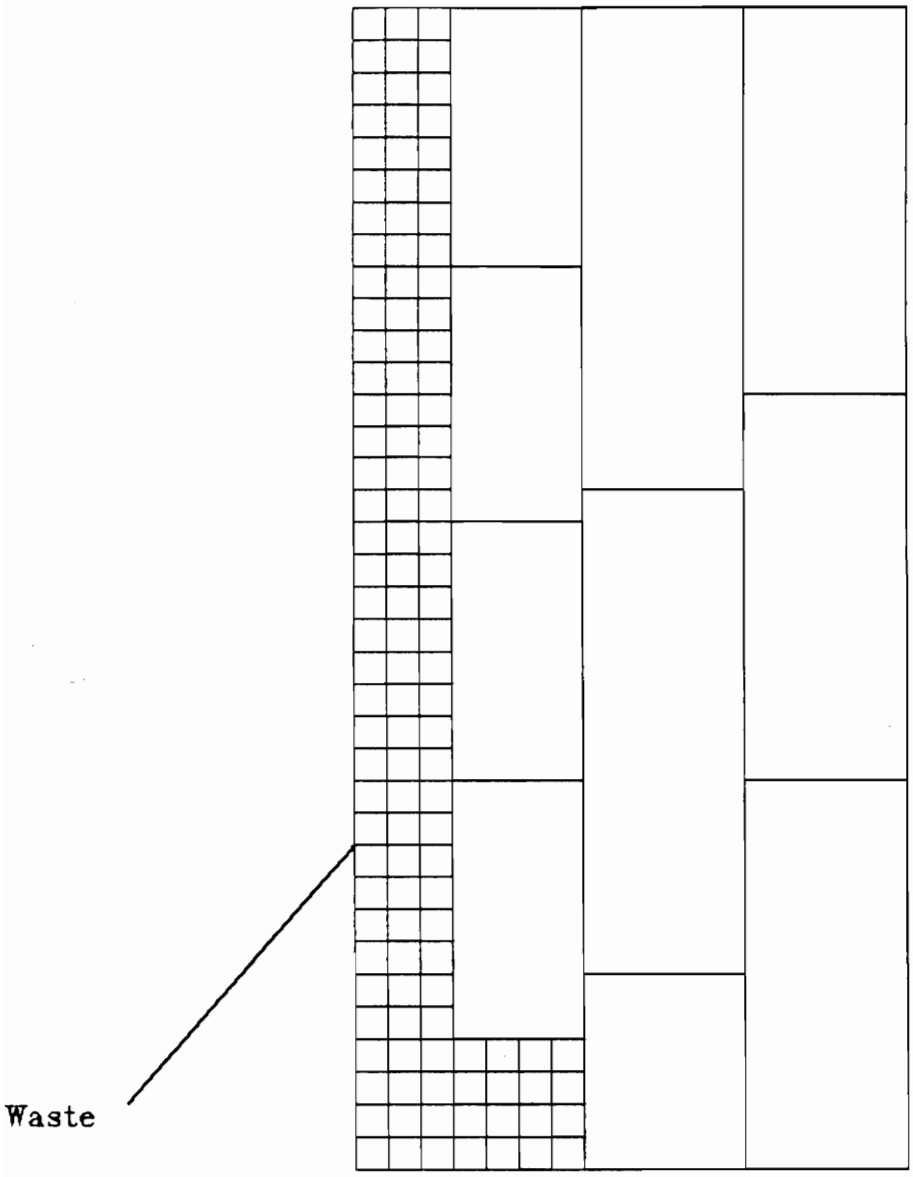


Figure 5 – Patterns Created by Heuristics

2. Continue until enough patterns are obtained to cover all pieces.

Such a heuristic procedure is used by Pegel [18] and by Adamowicz and Albano [1] for problems involving homogeneous patterns. Adamowicz and Albano [1] assume a restriction on the desired number of pieces of each type and modify the aforementioned heuristic. Their procedure differs from that described above in that if an arrangement cannot be found in the sheet of given size then a subsheet is considered and combinations of pieces are tried for a fit in the subsheet. Within a subsheet, only guillotine type arrangements are considered. Arrangements obtained are of the form given in Figure 6. After considering all combinations, the best combination for the subsheet is selected using a procedure based on dynamic programming.

Interactive Optimization: An interactive optimization procedure to solve two-dimensional stock cutting problems is a combination of the LP procedure presented above and the interactive graphics procedure for generating a pattern (column) also presented earlier. An application of this procedure to the case of packing circular disks on a circular plate is given in [21]. This interactive optimization procedure is also applied to cutting rectangular pieces from a rectangular stock. The arrangements obtained are shown in Figure 7. There are

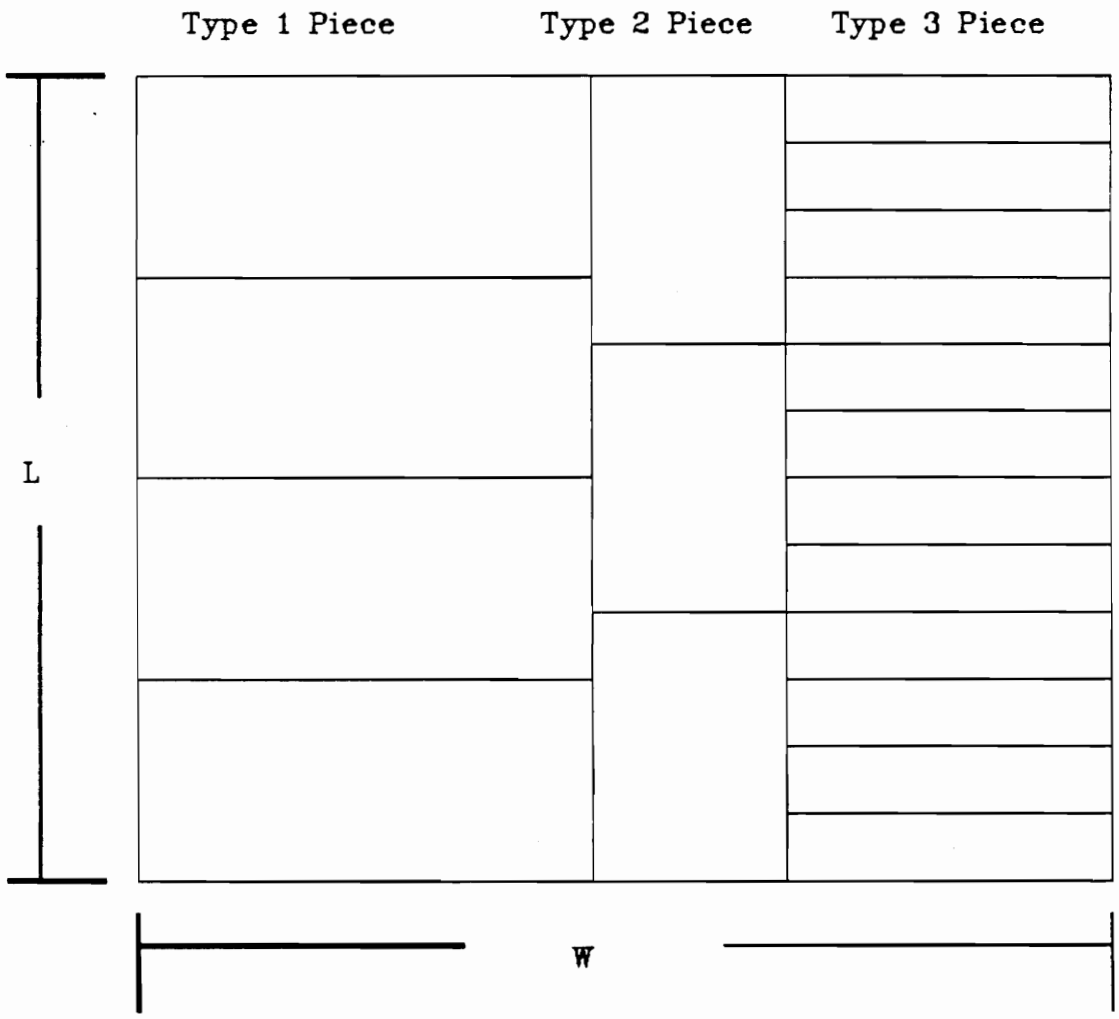


Figure 6 – Patterns from Modified Heuristic

Example: Size of Outside Rectangle = (8,5)

Size of Smaller Rectangles		Demand
1.	(3,2)	70
2.	(5,3)	40
3.	(6,2)	60

Optimal Patterns:

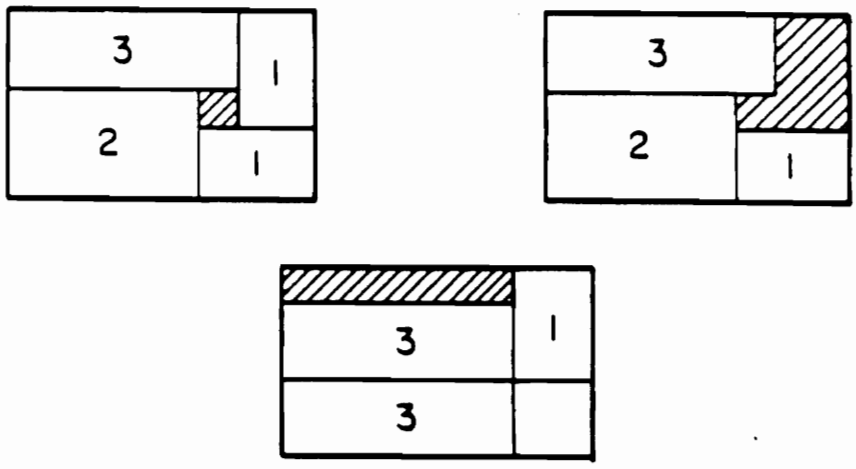


Figure 7 – Interactive Optimization Patterns

several advantages to this method. First, it can be applied to figures of any shape. Second, it is very similar to the way patterns are generated in practice except that this procedure generates for the designer the best available pattern to use at any stage rather than allowing the designer to determine the patterns heuristically. Third, this procedure allows any type of pattern to be generated, including patterns based on general cuts.

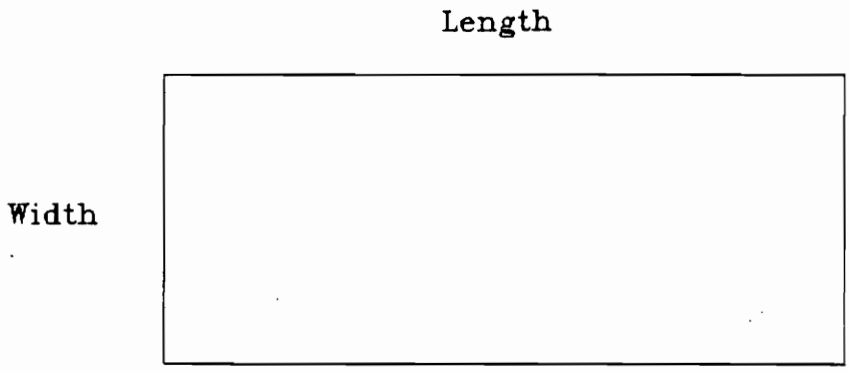
2.4 Practical Considerations and Implications for Solution Techniques

The pattern generating techniques and the solution methods discussed in this report provide optimal solutions to two-dimensional problems. However, these techniques fail to consider some of the practical concerns that are part of two-dimensional cutting. While failure to address these concerns does not preclude a precise analysis of a two-dimensional problem, it is important to be aware of some of the more common difficulties. Sheet orientation (or grain), non-uniform sheet quality (defective portions), non-rectangular pieces to be cut (as in fabric, leather and sheet metal cutting), variation in stock sizes, limitations on set-up times or number of set-ups, and the sequence of cutting patterns are among the most frequently encountered aberrations in two-dimensional problems.

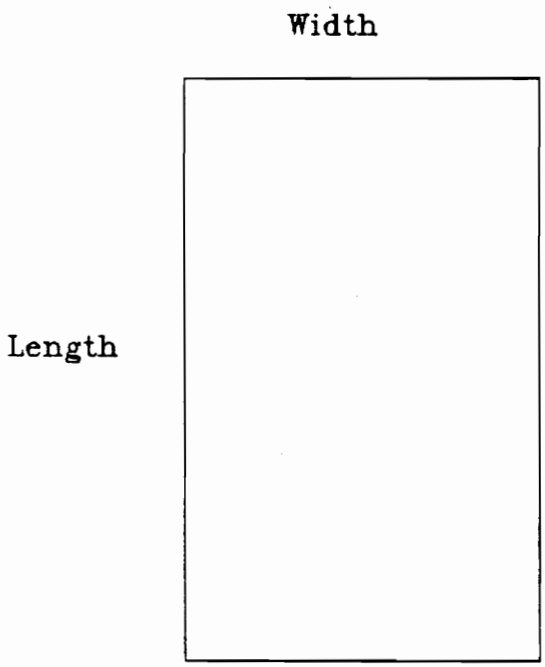
Sheet Orientation: The orientation of the stock

material is important because it indicates where on a sheet of stock each cut is to be performed (Figure 8). However, once the orientation of the sheet has been determined, an appropriate methodology can be used to determine how a given pattern is to be cut from the sheet.

Non-Uniform Sheet Quality: A relatively more difficult situation to handle is when the rectangular sheet from which pieces are to be cut is defective. That is, some spots of the sheet are defective and can be located before the cutting is undertaken (Figure 9). We shall assume that these defects are enclosed by rectangles and are not to be part of the required pieces. The objective is to cut as many pieces as possible to minimize waste. A DP procedure can be used to find the best patterns given the existence and location of defects. It is difficult to use the DP recursion equation of problem B2 when there are defects on the sheet. The problem can be solved using a one-dimensional version of the preceding recursion equation. In the one-dimensional formulation, the sheet is divided along its length into vertical sections which, in turn, are divided into strips. The value of a section is determined by packing only one kind of piece in that section. The location of defects is taken into consideration during the evaluation of a section to determine the number of pieces that can be packed in that section. Sections are



Horizontal Orientation



Vertical Orientation

Figure 8 – Sheet Orientation

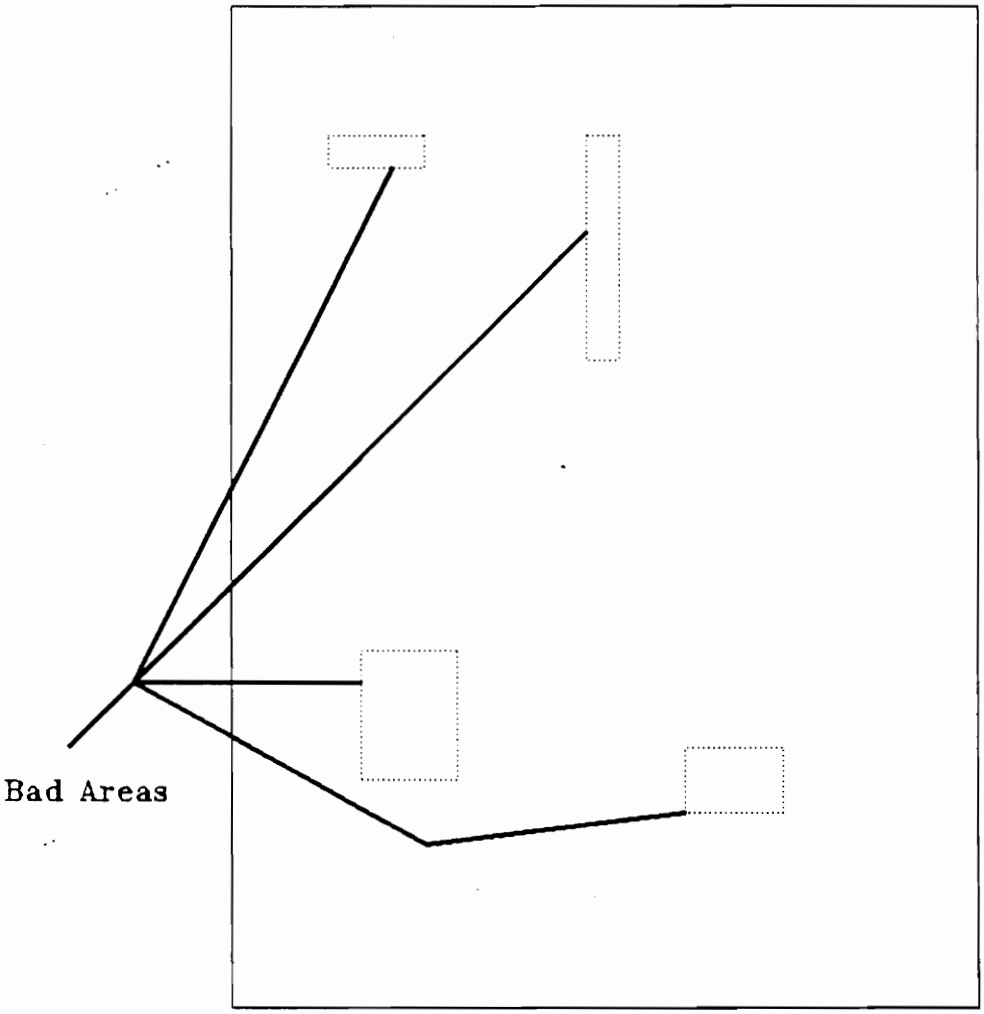
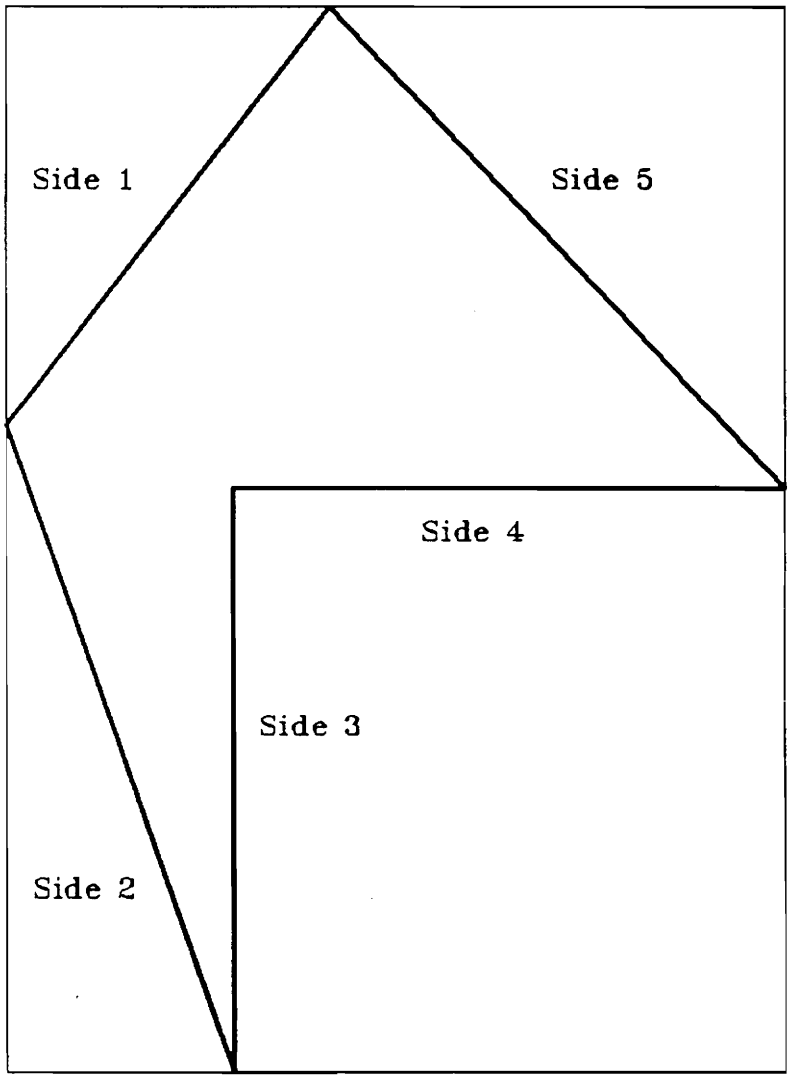


Figure 9 - Non-Uniform Sheet Quality

sequentially evaluated to determine maximum values. A detailed procedure for this problem is given by Hahn [12]. Also, some short cuts are suggested to speed up the procedure for on-line applications.

Non-Rectangular Shapes: When irregular shapes are involved, the types of cuts involved are general (non-guillotine) in nature (Figure 10). This problem is discussed by Haims and Freeman [13] in connection with the template layout problem. The difficulty in confronting problems of this type is easily seen when one considers the means by which feasible cutting patterns are generated. The rectangular piece cutting problem simply grows as the number of sides on the pieces grows.

Variation in Stock Sizes: When multiple sizes of stock are available, the problem is to determine how to cut a required number of rectangular pieces from multiple pieces of stock. An application of this is discussed in [18]. Assuming that guillotine cuts are made, the problem becomes an extrapolation of problem A2. This amounts to solving a larger number of knapsack problems at every iteration. If the choice among stock sizes is large, this procedure will require extensive computation time. Techniques are available for addressing this problem which significantly reduce the amount of time required to solve the problem.



Stock Sheet

Figure 10 – Irregular Shape Cutting

Limitations on Set-Up Times: Limitations on set-ups can be divided into two general categories. The first category deals with the sequence in which patterns are cut, and the second deals with process limitations on the number of different length and width cuts that can be made without adjusting the production line. Two approaches can be used to address limitations of the first type. The first technique is to solve the cutting problem via LP and then sequence the cutting patterns such that set-up cost (time) is minimized. For this approach, a "traveling salesman" type procedure [16] can be used to minimize set-up time. In this situation, each pattern is considered a city and the cost matrix for sequencing one pattern after another is determined based upon the number of new pieces to be cut in successive patterns. A second method is to generate patterns using a different procedure that simultaneously minimizes total waste and set-up time. For the second method, some heuristic procedures that minimize set-up time and waste are discussed in [7]. Set-up limitations due to production line constraints can be incorporated in the pattern generating problem formulation to prevent infeasible patterns from being generated.

3. Application of Linear Programming to New River Valley Workshop's Stock Cutting Problem.

3.1 General Approach

As discussed in previous sections, the principal elements involved in applying Linear Programming to a given two-dimensional cutting problem are:

- Type of cutting process
- Requirements for each size of piece
- Objective to be accomplished.

At NRV cuts are made parallel to a stock edge (orthogonal) and cuts are made continuously from one edge to another (guillotine cuts). Requirements for each size of piece come in the form of orders specifying order quantity and length and width dimensions. The objective in NRV's two-dimensional cutting process is to minimize stock waste while satisfying demanded sizes and quantities of pieces. Other elements can become part of a two-dimensional problem as practical issues, such as those presented earlier, are taken into account. In the following section, practical problem issues are discussed in the context of NRV's operation and their implications for a Linear Programming approach.

3.2 NRV Process Implications and LP

For NRV, only one of the issues presented earlier (number of set-ups) will be explicitly incorporated into the problem formulation. The rationale for excluding the other

issues is as follows: Sheet orientation is not a concern because NRV cuts its stock using only one orientation; lengthwise and with the grain. Non-Uniform sheet quality is not a concern because sheets are inspected prior to cutting and are either discarded or manually compensated for in the cutting process. NRV's operation produces only rectangular pieces from rectangular stock which precludes consideration of non-rectangular shapes. Variation in stock sizes is insignificant because dimensional tolerances on NRV stock are tightly controlled. Limitations on set-up times or the number of set-ups is a concern because of two process limitations at NRV. For width (or strip) cutting, NRV can only cut one width in any given set-up. This is because of a limitation on the saw used in the first step of the manufacturing process and because NRV does not wish to create a material handling problem by moving and recutting sheets of stock. That is to say, multiple widths could be cut from a given sheet of stock, but a single width would have to be cut on each sheet first, then the saw would have to be repositioned for the next width and each sheet recut. Thus width cutting patterns will be limited to one per sheet. Also, NRV can cut only three different lengths from a given strip without performing a production line adjustment. Therefore, the number of lengths in any single cutting pattern (for a given width strip) will be limited to

three (Figure 11).

3.3 Problem Specification and LP Formulation

The next step in solving NRV's cutting problem is to express each of the constraints and limitations discussed previously in mathematical terms. Once each element of NRV's problem has been numerically or algebraically expressed, a Linear Programming problem can be formulated and solved. As stated previously, two-dimensional problems can be solved in two stages. In NRV's case, these problem will be represented as two sub-problems S1 and S2. The first sub-problem (S1) determines feasible cutting patterns, subject to the dimensions of the stock, sizes of required pieces, and the number of different cuts that can be made in the length and width directions. The second sub-problem (S2) calculates the number of times each pattern will be used to cut the stock, subject to the demanded quantities.

Sub-Problem One (S1): Since NRV is cutting two-dimensional shapes from two-dimensional stock, the pattern generating problem involves determining both length and width cutting patterns. However, because NRV can only cut one width from a given sheet of stock without adjusting its production line, the width pattern problem reduces to dividing the stock width by each required width of board.

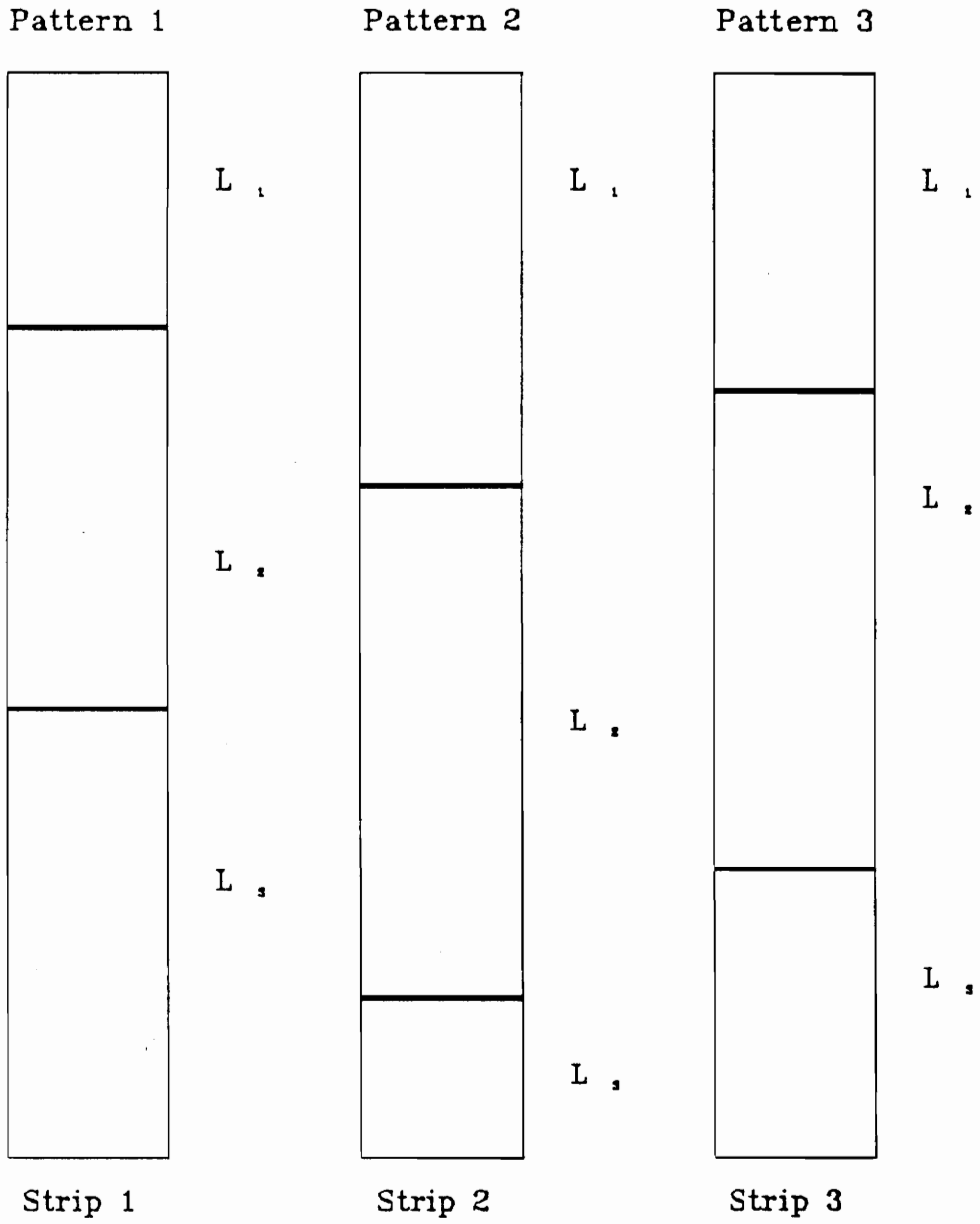


Figure 11 – Length Pattern Limitation

The width pattern problem is mathematically expressed as:

(S1) $Pw_i = [SW_i / Bw_i]$

where Pw_i = width cutting pattern (number of times to cut each sheet for width i).
SW = width of stock sheet in inches.
Bw_i = width of the ith required board.
[ ] : indicate integer value of expression

Pwi represents the number of times to cut a given sheet of stock to produce a given width board (Bw_i). Sub-problem S1 is repeated for each required width (Bw_i). Once S1 has been solved, the number of times to cut each sheet of stock along its width is known and the problem becomes (n) one-dimensional problems involving strips of stock cut to width(s) Bw_i (where i = 1 to n, the number of different widths contained on the work order). Now the problem is to determine for each width (Bw_i), how to combine the different required lengths (l_i) in each of these strips to minimize waste. From the NRV production constraints listed previously, the number of different length combinations that can be combined in any given strip is limited to three. Even with this limitation, the problem pattern generation problem can be quite difficult to formulate. As the number of different requested lengths grows, the number of three-way combinations of those lengths grows exponentially. There are, however, several methods for determining useful patterns which generate minimum waste. One such method involves setting up a primary Linear Programming problem

which specifies an initial set of patterns. The initial set of patterns is a diagonal matrix given by the ratio of the length of stock to each different requested length. Once a solution to the primary problem is found, a secondary problem is formulated using the dual prices for each constraint in the primary problem. Dual prices represent the amount by which the objective function changes in value for a one unit change in the right-hand side of a given constraint. These secondary problems are termed knapsack problems. Knapsack problems seek to maximize the value of items included in a solution (packed in a theoretical knapsack), subject to capacity and requirement constraints. Once an appropriate knapsack problem has been formulated and solved, the output (decision variable values) is used to create a new column (pattern) for the primary problem. The new column is added to the primary problem, the problem is resolved, and the process is repeated. Repetition of this process continues until the knapsack problem solution yields an objective function value equal to or less than one. When this occurs, the pattern under evaluation by the knapsack problem has a "cost" which makes it an unattractive alternative. The mathematical representation of this formulation is as follows:

(Primary Problem) Min. ΣP_j
s.t.

$$\Sigma a_{ij}P_j \geq N_i$$

where P_j = number of pieces cut using pattern j.
N_i = number of required pieces for each length i.
a_{ij} = number of pieces/strip of length i cut using pattern j.

(Secondary Problem) Min. $\Sigma Y_i DP_i$
s.t.

$$\Sigma l_i y_i \leq L$$

where Y_i = are integer variables.
l_i = length of ith piece.
L = length of strip of stock.
DP_i = Dual Price for each constraint in the primary problem.

The pattern matrix (a_{ij}'s) created by the above process contains all possible cutting patterns to the primary problem. This is matrix Q in the problem A1 (page 8). Once Q has been determined, A1 can be solved to determine how many times each pattern will be used in producing the required number of each different length.

In NRV's cutting process, there are additional issues which are worthy of consideration before proceeding with this method. While a knapsack solution algorithm is efficient in terms of the patterns it generates, it is not necessarily efficient in terms of the required procedures and computer code. Specifically, knapsack solution methods are useful, if not mandatory, when large numbers of different and complex patterns can be cut. As mentioned

previously, NRV can only cut three different lengths in any single pattern. Also, NRV rarely encounters work which requests more than four to six different lengths for a given width board. Because of these two factors, a more straightforward enumeration of feasible cutting patterns is called for. Specifically, simple enumeration of cutting patterns (i.e. combinations of different length boards) by way of nested looping is both effective and computationally efficient. Incorporated in the nested loops are logical expressions which eliminate patterns that have waste greater than the shortest board being considered in the pattern, and patterns which include more than three lengths. These two restrictions in conjunction with multiple nested loops, permit feasible cutting patterns to be generated without solving a knapsack problem at each iteration. This looping structure is shown in Figure 12. It is important to note that a pattern generation problem must be solved for each different required width board. Once feasible patterns have been created, sub-problem S2 can be formulated.

Sub-Problem 2 (S2): Once all feasible cutting patterns have been generated for a given width board, a problem formulation for optimal cutting can be created. This problem is most clearly represented in mathematical form, and is as follows:

```
450 FOR X = (11-NL) TO 10
460 IF L(X) <> 0 THEN NUM(X) = INT (TSTOCK/L(X))
470 NEXT X
480 S = 1
490 FOR E = 0 TO NUM(1)
500 FOR F = 0 TO NUM(2)
510 FOR G = 0 TO NUM(3)
520 FOR H = 0 TO NUM(4)
530 FOR I = 0 TO NUM(5)
540 FOR J = 0 TO NUM(6)
550 FOR K = 0 TO NUM(7)
560 FOR M = 0 TO NUM(8)
570 FOR N = 0 TO NUM(9)
580 FOR O = 0 TO NUM(10)
590 TLENGTH = (E*L(1) + F*L(2) + G*L(3) + H*L(4) + I*L(5) +
    J*L(6) + K*L(7) + M*L(8) + N*L(9) + O*L(10))
600 IF TLENGTH > TSTOCK THEN 840
610 IF (TSTOCK - TLENGTH) >= L(11-NL) THEN 840
620 IF E<>0 THEN DE = E/E
630 IF F<>0 THEN DF = F/F
640 IF G<>0 THEN DG = G/G
650 IF H<>0 THEN DH = H/H
660 IF I<>0 THEN DI = I/I
670 IF J<>0 THEN DJ = J/J
680 IF K<>0 THEN DK = K/K
690 IF M<>0 THEN DM = M/M
700 IF N<>0 THEN DN = N/N
710 IF O<>0 THEN DO = O/O
720 IF (DE + DF + DG + DH + DI + DJ + DK + DM + DN + DO) > 3
    THEN 840
730 IF NUM(1) <> 0 THEN A(S,1) = E
740 IF NUM(2) <> 0 THEN A(S,2) = F
750 IF NUM(3) <> 0 THEN A(S,3) = G
760 IF NUM(4) <> 0 THEN A(S,4) = H
770 IF NUM(5) <> 0 THEN A(S,5) = I
780 IF NUM(6) <> 0 THEN A(S,6) = J
790 IF NUM(7) <> 0 THEN A(S,7) = K
800 IF NUM(8) <> 0 THEN A(S,8) = M
810 IF NUM(9) <> 0 THEN A(S,9) = N
820 IF NUM(10) <> 0 THEN A(S,10) = O
830 S = S + 1
840 DE=0:DF=0:DG=0:DH=0:DI=0:DJ=0:DK=0:DM=0:DN=0:DO=0
850 NEXT O
860 NEXT N
870 NEXT M
880 NEXT K
890 NEXT J
900 NEXT I
910 NEXT H
920 NEXT G
930 NEXT F:NEXT E
```

Figure 12 – Pattern Generation Looping Structure

(S2) Min. ΣX_i
s.t.

$$QX_i \geq N_i$$

where X = vector, xi indicates the number of times pattern i is cut from stock.
Q = the matrix of feasible patterns.
N = vector, ni = demand for pieces of size i.

The solution to S2 indicates the number of times (x_i) each cutting pattern (i) will be used to satisfy the number (n_i) of demanded lengths (l_i). This solution represents the combination of cutting patterns that uses the minimum amount of stock in satisfying demanded quantities for each length and width board.

3.4 Computer-Based Implementation of LP

A computer-based Linear Programming system was developed to apply the Linear Programming techniques and methods that have been discussed, to NRV's cutting problem. This system consists of three modules:

1. Front-End - Data Entry
   Generate Pattern File
   Generate Model File
2. LINDO Solver - Commercially available LP/IP Solver
3. Report Generator - Read LINDO solution File
   Read Pattern File
   Generate Cutting Instructions
   Generate Summary Statistics

These three modules are controlled by a batch file that executes them in sequential order. The batch file is executed by typing the word "cut", followed by the return

key.

Module one, is written in the BASIC programming language and is approximately one hundred and fifty lines long (Appendix B). This module accepts user data in a format that is identical to the format used on NRV customer orders. The NRV format was adopted in the system to facilitate data entry and mitigate confusion. Once user data has been entered into the system, the user is queried to insure the accuracy of the information, and corrections are permitted. Once all data have been verified, the system generates feasible cutting patterns that meet the two restrictions for utilization and pattern complexity described earlier. Once the matrix of feasible cutting patterns is complete, the system generates an Integer Programming model in the format required by LINDO and writes the model to a file.

The second module is a commercially available Linear/Integer Programming solver called LINDO. LINDO is developed and marketed by LINDO Systems, Inc. of Chicago, Illinois. The version of LINDO that has been selected for this implementation can solve a matrix that has a maximum of one hundred equations, and a maximum of two hundred variables. NRV's largest problem has approximately seventy equations and seventy variables. LINDO has been configured to read the model file that is generated by module one, the first BASIC program, and write its solution values to a

file. It should be noted that the variables in the cutting models are integer variables. LINDO is equipped to solve LP problems with integer variables (Integer Programming) by using a branch and bound technique.

The third module in this implementation is a report generation program written in the BASIC programming language. This module reads the solution values generated by LINDO and reads the patterns that were created by module one. Based upon this data, module three creates a cutting order and summary statistics (see Appendix F). The cutting order is organized to tell the manufacturing floor what purchase order they are working, what type of stock is required, the date, the patterns to be used, and the number of times to use each pattern. The summary statistics indicate how many strips are to be cut for a given width, the number of linear inches of material required, the number of linear inches of waste produced, the strip utilization, the number of sheets needed, and the amount of edge waste that is created on each sheet. Appendix A contains a sample New River Valley work order. Three sample problem formulations for this work order are presented in the next section. The results from each sample problem including all feasible cutting patterns, number of times each pattern is used, number of strips of each width required, number of sheets required, pattern determination time and stock

utilization data are presented and discussed.

4. Test Data and Results

This section presents three sample New River Valley cutting problems. In each problem, the required quantities, widths, and lengths are shown. Following that information is the list of patterns generated using the algorithm in Appendix B. Following the list of patterns is the solution to the Linear Programming model (with integer variables) for the given problem. Following that is summary data which specifies the number of sheets of stock required, solution time, and strip and sheet utilization values generated using LINDO and the algorithm in Appendix C. Following the solution to each problem is a brief discussion and analysis which highlights the solution time and stock utilization attained by the system.

Sample Problem 1

Sample problem one is for 4.5 inch wide boards. This problem involves three different demanded lengths and quantities. The stock used in problem one is 96.75 X 48.75 inches.

Board Width	Quantity Demanded	Length(s) Demanded
4.50 "	2,034	14.25 "
4.50 "	1,019	20.25 "
4.50 "	619	28.25 "

Feasible Cutting Patterns for 4.50 " Boards

Pattern No.	14.25 "	20.25 "	28.25 "
1	0	0	3
2	0	3	1
3	1	1	2
4	1	2	1
5	1	4	0
6	2	0	2
7	2	3	0
8	3	1	1
9	3	2	0
10	4	0	1
11	5	1	0
12	6	0	0

Pattern No.	Number of Strips to Cut with Pattern i
1	0
2	1
3	0
4	0
5	95
6	0
7	0
8	618
9	0
10	0
11	18
12	0

Number of Sheets of Stock Required: **73.2**

Pattern Generation + Solution Time: **1.1 minutes**

Strip Utilization: **94.85 %**

Sheet Edge Waste: **7.69 %**

Analysis and Discussion of Sample Problem 1 Data

For this problem, twelve different cutting patterns were produced. These patterns represent all possible ways of cutting the three lengths (14.25, 20.25, 28.25) from stock 96.75 inches long. Patterns which combined the three lengths in ways that would result in more than 14.25 inches of waste were discarded as inefficient. An illustration of the cutting patterns generated for this problem is shown in figure 13. These patterns were used to generate a Linear Programming model (integer variables) that was solved by a linear optimizer (LINDO). As with most linear optimizers, LINDO seeks to optimize an objective function over a feasible solution space. For this problem, the possible solutions are described by the equations which represent all of the ways each length of board can be cut subject to the quantity demanded for that length. These equations are derived from the patterns and are as follows:

Objective:

Minimize $X1 + X2 + X3 + X4 + X5 + X6 + X7 + X8 + X9 + X10 + X11 + X12$

Subject to:

14.25" $\quad 0X1 + 0X2 + 1X3 + 1X4 + 1X5 + 2X6 + 2X7 + 3X8 + 3X9 + 4X10 + 5X11 + 6X12 \geq 2034$

20.25" $\quad 0X1 + 3X2 + 1X3 + 2X4 + 4X5 + 0X6 + 3X7 + 1X8 + 2X9 + 0X10 + 1X11 + 0X12 \geq 1019$

28.25 " $\quad 3X1 + 1X2 + 2X3 + 1X4 + 0X5 + 2X6 + 0X7 + 1X8 + 0X9 + 1X10 + 0X11 + 0X12 \geq 619$

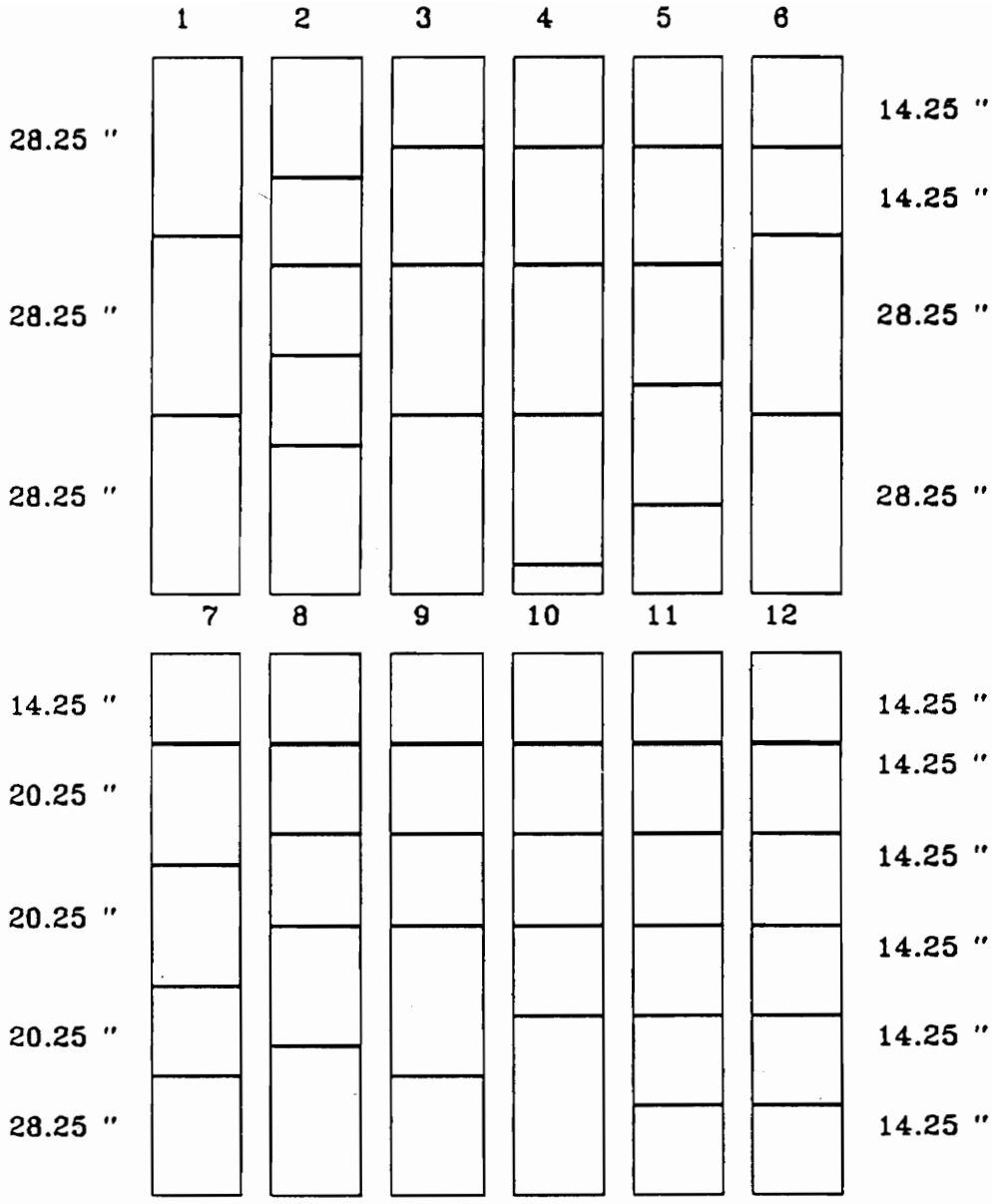


Figure 13 – Sample Problem One Cutting Patterns

$X_i \geq 0$, integer

LINDO solves this system of inequalities by minimizing the objective function over the three dimensional space defined by the equations for 14.25 inch, 20.25 inch, and 28.25 inch boards. The solution to this optimization (minimization) problem is a set of values for each X_i. Using the solution values, the number of strips of stock required, the number of sheets of stock required, and the amount of waste are determined. In this problem, the optimal solution requires 732 strips of stock, 73.2 sheets of stock (i.e. 10 strips/sheet), and utilizes 94.85 percent of each strip of stock. It should be noted that while 73.2 sheets of stock are required for this problem, the 0.8 sheet of stock that remains is not considered waste since it can be reused on subsequent cutting orders. It is also important to note that since exactly ten 4.50 inch strips can be cut from a 48.75 inch wide sheet of stock, there is no accumulation of edge waste from sheet to sheet.

The solution to Sample Problem One demonstrates the value of Linear Programming analysis for this type of problem. In approximately one minute, feasible cutting patterns, an optimal problem solution, and cutting instructions are created using LP and two input/output routines (see Appendices B, C, D, and E). Using their manual methods, NRV management would have spent hours

determining optimal cutting instructions and would not be assured that their solution was truly optimal. Also, through the use of the LP solution technique, NRV's objective stock utilization of 90% has been exceeded by 4.85% on a per strip basis. Utilization is calculated on a per strip basis since sheet optimization is not possible. That is, since NRV can cut only one width per sheet of stock, optimization across the width of a sheet of stock is prohibited. Thus, strip utilization is a more revealing and meaningful performance measure than sheet utilization. Now consider sample problem two.

Sample Problem 2

Sample problem two also involves three different demanded lengths and quantities, but the required width is 6.50 inches. The stock used in problem two is 96.75 X 48.75 inches.

Board Width	Quantity Demanded	Length(s) Demanded
6.50 "	721	18.25 "
6.50 "	1,029	25.00 "
6.50 "	2,464	28.25 "

Feasible Cutting Patterns for 6.50 " Boards
(note only those patterns used are shown)

Pattern No.	18.25 "	25.00 "	28.25 "
1	0	0	3
3	1	2	1
5	2	0	2

Pattern No.	Number of Strips to Cut with Pattern i
1	581
3	515
5	103

Number of Sheets of Stock Required: 171.29

Pattern Generation + Solution Time: 1.0 minutes

Strip Utilization: 93.55 %

Sheet Edge Waste: 6.67 %

Analysis and Discussion of Sample Problem 2 Data

Sample problem two data generated 10 possible cutting patterns. Three of the ten patterns were selected by LINDO to meet the required demand and minimize waste. Again, all possible combinations of the 18.25, 25.00, and 28.25 inch long boards were generated and used to formulate a Linear Programming problem for LINDO. The equations generated are:

Objective:

Minimize X1 + X2 + X3 + X4 + X5 + X6 + X7 + X8 + X9 + X10

Subject to:

18.25" $0X1 + 0X2 + X3 + X4 + 2X5 + 2X6 + 2X7 + 3X8 + 3X9 + 5X10 \geq 721$

25.00" $0X1 + X2 + 2X3 + 3X4 + 0X5 + X6 + 2X7 + 0X8 + X9 + 0X10 \geq 1029$

28.25" $3X1 + 2X2 + X3 + 0X4 + 2X5 + X6 + 0X7 + X8 + 0X9 + 0X10 \geq 2464$

$X_i \geq 0$, integer

LINDO evaluated the objective function over the three dimensional space defined by the 18.25, 25.00, and 28.25 inch equations, and determined the least wastage solution to be: use pattern (1) 581 times; use pattern (3) 515 times; and use pattern (5) 103 times. Based upon these solution values, the number of sheets of stock to be cut is 171.29 which results in 93.55 % strip utilization and 6.67% sheet waste. Again, strip and sheet utilizations are reported separately since the NRV cutting process does not allow width optimization. Now consider sample problem three.

Sample Problem 3

Sample problem three involves two different demanded lengths and quantities. The stock for sample problem three is 96.75 X 48.75 inches.

Board Width	Quantity Demanded	Length(s) Demanded
6.75 "	2,259	13.00 "
6.75 "	6,978	14.75 "

Feasible Cutting Patterns for 6.75 " Boards

Pattern No.	13.00 "	14.75 "
1	0	6
2	1	5
3	2	4
4	4	3
5	5	2
6	6	1
7	7	0

Pattern No.	Number of Strips to Cut with Pattern i
1	881
4	564
7	1

Number of Sheets of Stock Required: 206.57

Pattern Generation + Solution Time: 0.75 minutes

Strip Utilization: 94.60 %

Sheet Edge Waste: 3.08 %

Analysis and Discussion of Sample Problem 3 Data

Sample problem three data generated 7 possible cutting patterns. Three of these patterns were selected by LINDO in optimizing the problem. Again, all possible combinations of the 13.00 and 14.75 inch long boards were generated and used to formulate a Linear Programming problem for LINDO. The equations generated are:

Objective:

Minimize X1 + X2 + X3 + X4 + X5 + X6 + X7

Subject to:

13.00"	$0X1 + 1X2 + 2X3 + 4X4 + 5X5 + 6X6 + 7X7$	≥ 2259
14.75"	$6X1 + 5X2 + 4X3 + 3X4 + 2X5 + X6 + 0X7$	≥ 6978

$X_i \geq 0$, integer

LINDO evaluated the objective function over the two dimensional space defined by the 13.00 and 14.75 inch equations, and determined the least wastage solution to be: use pattern (1) 881 times; use pattern (4) 564 times; and use pattern (7) 1 time. Based upon these solution values, the number of sheets of stock to be cut is 206.57 which results in 94.60 % strip utilization and 3.08% sheet waste. Again, strip and sheet utilizations are reported separately since the NRV cutting process does not allow width optimization. The significance and difference between strip and sheet utilization will be discussed in more detail in section five.

5. Conclusions and Comparison of Results

5.1 Conclusions

The three sample problems presented in the previous section clearly demonstrate the power and usefulness of Linear Programming and automation in solving NRV's production management problems. By using this computer-based solution system and Linear Programming to translate customer orders into cutting instructions, the time to process an order has decreased from six to eight hours, to little more than one minute. Simultaneously, through the application of Linear Programming analysis, NRV's stock utilization has been improved and the consistency and accuracy of solutions have been greatly enhanced.

While stock utilization improvement was not a critical objective for NRV management, the improvements illustrated in the sample problems translate into improved profit margins for NRV's operation. The principal and overwhelming benefits of the automated solution presented here are the vast improvement in processing time and consistent analytical evaluation of the production problem. The savings in terms of labor hours, reduced frustration, increased quality and consistency are indeed large.

5.2 Comparison of Results

The first performance data to be compared is computation time. As reported by NRV management, a single

work order normally requires six to eight hours of uninterrupted hand calculation (not including the additional time needed to manually create the cutting instructions). In all three of the sample problems (and other problems that have been solved at NRV using this system), the average elapsed time from entering the order data into the system to having a complete cutting order printed is approximately sixty seconds for each different width board contained on the customer order. Customer orders contain an average of three different width boards. Thus, the total average computation time per order is approximately three minutes. The magnitude of this improvement makes this comparison speak for itself. It should be noted that not only is computation time vastly reduced, but computational consistency and accuracy have also been greatly improved. Because each problem is formulated and evaluated using the same Linear Programming approach, there is no longer a need to question whether the solution obtained is indeed truly optimal.

The second performance criteria to be compared is stock utilization. From data provided by NRV, the average sheet utilization achieved through the manual process is approximately seventy-five percent. The utilizations determined by the LP method are approximately ninety-five percent for strip utilization, and approximately ninety

percent for sheet utilization. The sheet utilization improvement represents an average reduction in material cost of fifteen percent on a per order basis. Based upon NRV's estimate that seventy percent of their product cost is material, this improvement should save NRV a considerable amount of capital by reducing stock expenses. Other cutting problems have been solved using this system and have consistently resulted in strip utilization in the low to mid ninety percentile range.

Strip and sheet utilization are two distinct performance measures. Sheet utilization specifies the percentage of the entire sheet of stock that is utilized for useful production. Strip utilization specifies the percentage of strips that is utilized for useful production. Because NRV can only cut one width strip from any given sheet of stock, there is no opportunity to optimize cutting across the width of the stock. That is, the width utilization for a sheet of stock is solely determined by the width of the desired strip. This concept is illustrated in figure 14. Since optimization in the width direction is precluded by this limitation, the most meaningful performance measurement is strip utilization.

Strip utilization opportunities exist since NRV can cut up to three different lengths from any given strip. By combining required lengths in the best possible way, length optimization can be achieved. This optimization problem was

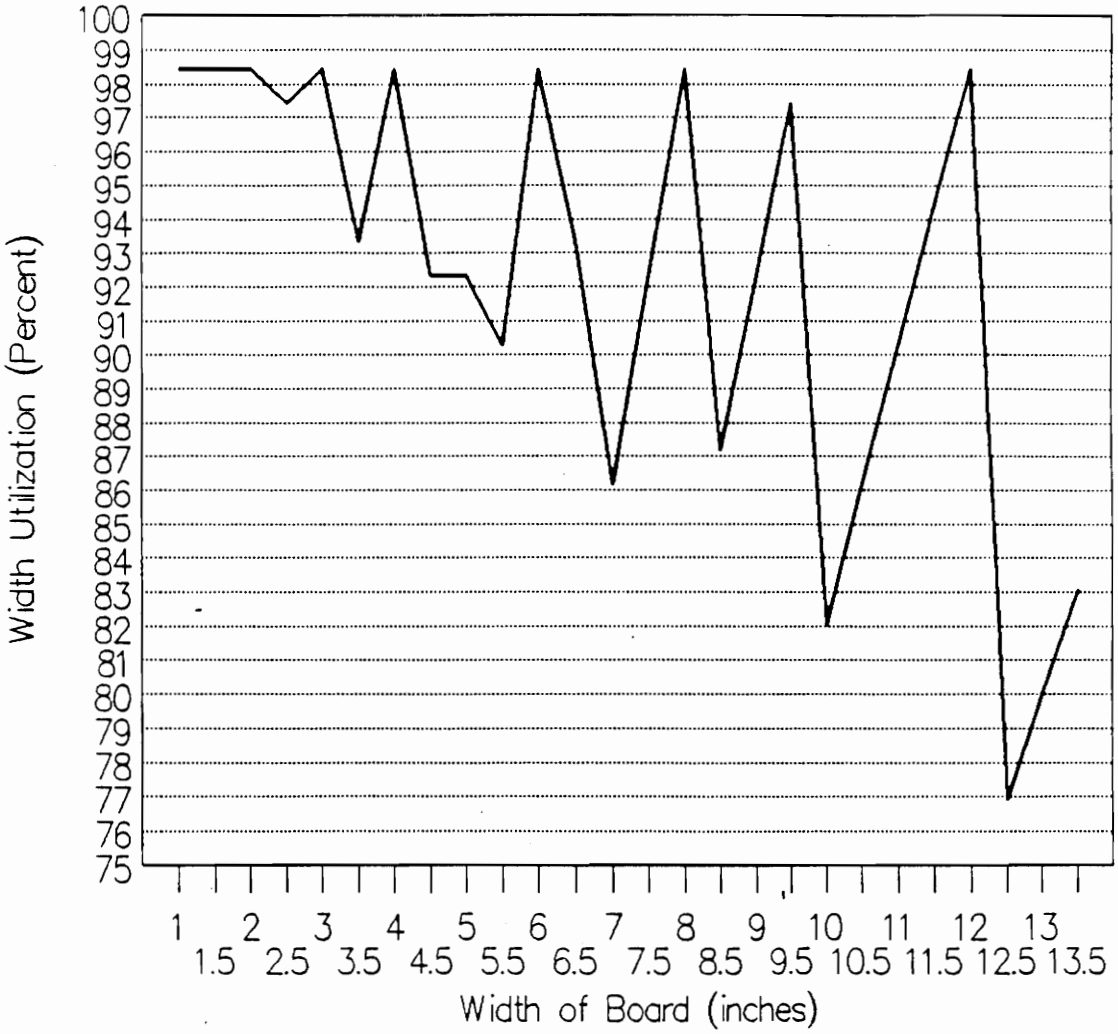


Figure 14 – Sheet Utilization vs. Strip Width

the focus of this research. Through the use of the routines contained in appendices B, C, D, and E, length optimization was obtained. For a description of each algorithm, see the corresponding appendix.

6. Recommendations for Future Research

Because of the limitations of the NRV production process, width optimization is not possible. Future research should be conducted to either upgrade the NRV process to permit multiple width cutting, or specify material handling procedures which would permit easy recutting of stock sheets. Either of these changes would allow NRV to truly optimize sheet utilization and thus minimize material costs.

The algorithms developed in this research have been implemented at the NRV Workshop. Operational results for the system have been overwhelmingly positive. NRV management has been delighted with the reduced computational burden that the system allows, and the speed with which it creates cutting instructions for the shop floor. See appendix F for the cutting instructions from sample problems one, two, and three. NRV operations has reacted very positively to the format and content of the cutting instructions and has been able to easily incorporate them into their operating procedures.

7. REFERENCES

1. Adamowicz, M., and Albano, A., "A Solution of the Rec-tangular Cutting-stock Problem," IEEE Trans. System, Man,Cybernetics, Vol. smc-6, No. 6, 1976, pp. 302-310.

2. Bellman, R. E., Dynamic Programming, Princeton University Press, Princeton, 1957.

3. Cani, Phillip De, "A Note on the Two-Dimensional Rectangular Cutting-stock Problem," J. opl. Res. Soc., Vol. 29, No. 7, 1978, pp. 703-706.

4. Christofides, N., and Whitlock, C., "An Algorithm for Two Dimensional Cutting Problem," ops. Res., Vol. 25, No. 1, 1977, pp. 30-44.

5. Cooper, L., and Cooper, M. W., Introduction to Dynamic Programming, Pergamon Press, 1981.

6. Dantzig, A. B., and Wolfe, P., "Decomposition Principle for Linear Programs," opns. Res., Vol. 2, 1954, pp. 275-288.

7. Dyson, R. G., and Gregory, A. S., "The Cutting Stock Problems in the Flat Glass Industry," Opl. Res. Quart., Vol. 25, No. 1, 1974, pp. 41-53.

8. Gilmore, P. C., and Gomory, R. F., "Multistage Cutting-Stock Problem of Two and More Dimensions," opns. Res., Vol. 13, 1965, pp. 94-120.

9. Gilmore, P. C., and Gomory, R. F., "The Theory and Computation of Knapsack Functions," ops. Res., Vol. 14, 1966, pp. 1045-1074.

10. Golden, B. L., "Approach to Cutting Stock Problem," AIIE Trans., 1976, pp. 256-272.

11. Hadley, G., Linear Programming, Addison Wesley, Reading, Massachusetts, 1962.

12. Hahn, S. G., "On the Optimal Cutting of Defective Sheets," Ops. Res., Vol. 16, No. 6, 1968, pp. 1100-1114.

13. Haims, M. J., and Freeman, H., "A Multistage Solution of the Template-Layout Problem," IEEE Transactions on Systems Science and Cybernetics, SSC-6, No. 2, 1970, pp. 145-151.

14. Herz, J. C., "Recursive Computational Procedure for Two-Dimensional Stock Cutting," IBM J. of Res. and Dev., Vol. 16, No. 5, 1972, pp. 462-469.

15. Lasdon, L. S., Optimization Theory for Large Systems, McMillan, 1970.

16. Little, J. D. C., Murty, K. G., Sweeney, D. W., and Karel, C., "An Algorithm for the Travelling Salesman Problem," opns. Res., Vol. 11, 1963, pp. 979-998.

17. Nemhauser, G. L., Introduction to Dynamic Programming, Wiley, New York, 1966.

18. Pegels, C. C., "Heuristic Scheduling Models for Variants of the Two-Dimensional Cutting-stock Problem," Tappi, Vol. 5, No. 11, 1967, pp. 532-535.

19. Salkin, H. M., and Dekleyver, C. A., "The Knapsack Problem: A Survey," Nav. Res. Log. Quart., Vol. 22, No. 7, 1975, pp. 127-144.

20. Sarin, S. C., "The Mixed Disc Packing Problem: Part I: Some Bounds on Density," IIE Transactions, Vol. 15, No. 1, March, 1983, pp. 37-45.

21. Sarin, S. C., "The Mixed Disc Packing Problem: Part II. An Interactive Optimization Procedure," IIE Transactions, Vol. 15, No. 2, June, 1983, pp. 91-98.

22. Sarin, S. C., "Two-Dimensional Stock Cutting Problems and Solution Methodologies," Transactions of the ASME, Vol. 105, 1983, pp. 155-160.

23. Steudel, H. F., "Generating Pallet Loading Patterns: A Special Case of the Two-Dimensional Cutting Stock Problem," Mgt. Sc., Vol. 25, No. 10, 1979, pp. 997-1004.

8. APPENDICES

PURCHASE ORDER

WEBB FURNITURE ENTERPRISES, INC.

P.O. BOX 1277; GALAX, VA 24333

Phone: (703) 236-2984 Fax: (703) 236-2999

File 339-W-02

. River Valley Workshop
53 Duncan Lane
Radford, VA 24141

SHIP TO: Plant #2

Promised Complete 6/1/90

DATE	DELIVERY DATE	SHIP VIA	PAGE	OF	P.O. NUMBER
5/ 5/90	6/13/90		1	1	H41 - 794

OTAL IECES ORDER	ITEM NUMBER	ITEM DESCRIPTION	LENGTH		WIDTH		THCKNS
	*****	DRAWER SIDES & BACKS					
		7/16'LUAUN DRAWER SIDES (UNFINISHED)					
(1) 2259→27	3926	ALL	13		~~6~~	~~3/4~~	
(2) 7316 →	3905	TOP & CENTER	14	3/4	4	3/4	840 pcs Fa... 75
	3911	TOP	14	3/4	4	3/4	
	3914	ALL	14	3/4	4	3/4	
(3) 6978→	3905	BOTTOM	14	3/4	~~6~~	~~3/4~~	
	3911	LOWER	14	3/4	6	~~3/4~~	
		7/16' LUAUN DRAWER BACKS (UNFINISHED)					
(4) 2034 →	3905	END	14	1/4	4	1/2	
(5) 721→5	3926	ALL	18	1/4	~~6~~	~~1/2~~	
(6) 1019→11	3905	CENTER	20	1/4	~~4~~	~~1/2~~	
(7) 1029→11	3905	BOTTOM	25		~~6~~	~~1/2~~	
(8) 619→7	3911	TOP	28	1/4	~~4~~	1/2	
(9) 2464 →	3911	LOWER	28	1/4	~~6~~	~~1/2~~	
0	3914	ALL	32		4	1/2	

APPENDIX B. - PATTERN GENERATING ALGORITHM

```
10 KEY OFF
20 DIM L(20): DIM NUM(20):DIM A(100,100)
30 COLOR 15,1,10
40 CLS
50 PRINT"Enter the P.O. Number from the Order."
60 INPUT PO
70 PRINT"Enter the width of the required pieces."
80 INPUT PWDTH
90 PRINT"Enter the number of different lengths for ";:PRINT
   USING "###.##";PWDTH;:PRINT" width boards."
100 INPUT NL
110 PRINT"Enter each different quantity and length."
120 FOR X = 10 TO (11-NL) STEP -1
130 INPUT Q(X),L(X)
140 NEXT X
150 CLS
160 PRINT"Line No.     Quantity    Length":PRINT
170 FOR X = 10 TO (11-NL) STEP -1
180 PRINT USING "  ##";X;:PRINT USING "        ######";Q(X);:
    PRINT USING "        ###.##";L(X)
190 NEXT X
200 PRINT:PRINT"Is this data correct?"
210 INPUT CD$
220 IF CD$ = "y" THEN 290
230 IF CD$ = "Y" THEN 290
240 PRINT"Enter the line number you wish to change."
250 INPUT LC
260 PRINT"Re-enter the quantity and length now."
270 INPUT Q(LC),L(LC)
280 GOTO 170
290 PRINT"Enter the length of the stock material in inches."
300 INPUT TSTOCK
310 PRINT"Enter the width of the stock material in inches."
320 INPUT WDTH
330 CLS:PRINT:PRINT:PRINT:PRINT:PRINT:PRINT:PRINT:
    PRINT"Generating solution.  Please wait ..."
340 FOR Y = (11-NL) TO 10
350 FOR X = (11-NL) TO 10
360 IF L(Y) >= L(X) THEN 400
370 D=L(X): Q=Q(X)
380 L(X) = L(Y): Q(X) = Q(Y)
390 L(Y) = D: Q(Y) = Q
400 NEXT X
410 NEXT Y
420 REM FOR X = 10 TO (11-NL) STEP -1
430 REM PRINT "Length ";:PRINT USING"### ";X;:PRINT"= ";:PRINT
    USING      "###.##";L(X);:PRINT"       Quanity      =";
   :PRINT USING"#####";Q(X)
```

```
440 REM NEXT X:PRINT
450 FOR X = (11-NL) TO 10
460 IF L(X) <> 0 THEN NUM(X) = INT (TSTOCK/L(X))
470 NEXT X
480 S = 1
490 FOR E = 0 TO NUM(1)
500 FOR F = 0 TO NUM(2)
510 FOR G = 0 TO NUM(3)
520 FOR H = 0 TO NUM(4)
530 FOR I = 0 TO NUM(5)
540 FOR J = 0 TO NUM(6)
550 FOR K = 0 TO NUM(7)
560 FOR M = 0 TO NUM(8)
570 FOR N = 0 TO NUM(9)
580 FOR O = 0 TO NUM(10)
590 TLENGTH = (E*L(1) + F*L(2) + G*L(3) + H*L(4) + I*L(5) +
    J*L(6) + K*L(7) + M*L(8) + N*L(9) + O*L(10))
600 IF TLENGTH > TSTOCK THEN 840
610 IF (TSTOCK - TLENGTH) >= L(11-NL) THEN 840
620 IF E<>0 THEN DE = E/E
630 IF F<>0 THEN DF = F/F
640 IF G<>0 THEN DG = G/G
650 IF H<>0 THEN DH = H/H
660 IF I<>0 THEN DI = I/I
670 IF J<>0 THEN DJ = J/J
680 IF K<>0 THEN DK = K/K
690 IF M<>0 THEN DM = M/M
700 IF N<>0 THEN DN = N/N
710 IF O<>0 THEN DO = O/O
720 IF (DE + DF + DG + DH + DI + DJ + DK + DM + DN + DO) > 3
    THEN 840
730 IF NUM(1) <> 0 THEN A(S,1) = E
740 IF NUM(2) <> 0 THEN A(S,2) = F
750 IF NUM(3) <> 0 THEN A(S,3) = G
760 IF NUM(4) <> 0 THEN A(S,4) = H
770 IF NUM(5) <> 0 THEN A(S,5) = I
780 IF NUM(6) <> 0 THEN A(S,6) = J
790 IF NUM(7) <> 0 THEN A(S,7) = K
800 IF NUM(8) <> 0 THEN A(S,8) = M
810 IF NUM(9) <> 0 THEN A(S,9) = N
820 IF NUM(10) <> 0 THEN A(S,10) = O
830 S = S + 1
840 DE=0:DF=0:DG=0:DH=0:DI=0:DJ=0:DK=0:DM=0:DN=0:DO=0
850 NEXT O
860 NEXT N
870 NEXT M
880 NEXT K
890 NEXT J
900 NEXT I
910 NEXT H
920 NEXT G
```

```
930 NEXT F
940 NEXT E
950 OPEN "pats.dat" FOR OUTPUT AS #2:PRINT #2, TSTOCK,WDTH,PWDTH,PO
960 REM PRINT"Pattern No.";
970 REM PRINT:PRINT
980 FOR X = 1 TO (S-1)
990 REM PRINT USING"##";X;
1000 PRINT #2, USING"##";X;
1010 FOR Y = 1 TO 10
1020 REM PRINT USING "## ";A(X,Y);:PRINT USING"##.##";L(Y);
1030 PRINT #2, USING "## ";A(X,Y);:PRINT #2, USING"##.##";L(Y);
1040 NEXT Y
1050 PRINT #2, ""
1060 NEXT X
1070 CLOSE #2
1080 REM
1090 REM / This section generates the set of linear inequalities
     for LINDO
1100 REM
1110 REM PRINT"min ";
1120 REM FOR X = 1 TO (S-1)
1130 REM PRINT "1 x";:IF X < 10 THEN PRINT USING "#";X;
     ELSE PRINT USING"##";X;
1140 REM IF X < (S-1) THEN PRINT " +";
1150 REM NEXT X
1160 REM PRINT
1170 REM PRINT"st"
1180 REM DNL = NL
1190 REM FOR X = (11-NL) TO 10
1200 REM FOR Y = 1 TO (S-1)
1210 REM PRINT USING"#";A(Y,X);:PRINT "x";:IF Y < 10 THEN PRINT
     USING"#";Y; ELSE PRINT USING"##";Y;
1220 REM IF Y < (S-1) THEN PRINT " + ";
1230 REM NEXT Y
1240 REM PRINT " >= ";:PRINT USING "#####";Q(11-DNL):PRINT
1250 REM IF DNL > 0 THEN DNL = DNL -1
1260 REM NEXT X
1270 REM PRINT"end"
1280 REM PRINT:PRINT"GIN ";:PRINT USING"##";(S-1)
1290 REM
1300 REM
1310 OPEN "saw.mod" FOR OUTPUT AS #1
1320 PRINT #1,"min "
1330 FOR X = 1 TO (S-1)
1340 PRINT #1,"1 x";:IF X < 10 THEN PRINT #1, USING "#";X;
     ELSE PRINT #1, USING"##";X;
1350 IF X < (S-1) THEN PRINT #1," +"
1360 NEXT X
1370 PRINT #1,"":PRINT #1,"st"
1380 DNL = NL
1390 FOR X = (11-NL) TO 10
```

```
1400 FOR Y = 1 TO (S-1)
1410 PRINT #1, USING"#";A(Y,X);:PRINT #1,"x";:IF Y < 10 THEN
     PRINT #1, USING"#";Y; ELSE PRINT #1, USING"##";Y;
1420 IF Y < (S-1) THEN PRINT #1," + "
1430 NEXT Y
1440 PRINT #1," >= ";:PRINT #1, USING "#####";Q(11-DNL)
1450 IF DNL > 0 THEN DNL = DNL -1
1460 NEXT X
1470 PRINT #1,"end"
1480 PRINT #1,"GIN ";:PRINT #1, USING"##";(S-1)
1490 CLOSE #1
1500 SYSTEM
1510 END
```

APPENDIX C. - RESULTS AND CUTTING ANALYSIS ALGORITHM

```
10 CLS
20 I = 1
30 DIM VNAME(100),NVL(100), NRD(100),D1(100),D2(100)
40 DIM PATTERN(100),PIECES(100),LENGTH(100),LUSED(100),WASTE(100)
50 DIM PL1(100),PL2(100),PL3(100),PL4(100),PL5(100),PL6(100),
   PL7(100),PL8(100),PL9(100),PL10(100)
60 DIM L1(100),L2(100),L3(100),L4(100),L5(100),L6(100),L7(100),
   L8(100),L9(100),L10(100)
70 OPEN "nrvinfo.out" FOR INPUT AS #1
80 INPUT #1,T1,T2,T3,T4
90 INPUT #1, VNAME(I),NVL(I),NRD(I),D1(I),D2(I)
100 IF EOF(1) THEN 130
110 I = I + 1
120 GOTO 90
130 CLOSE #1
140 REM FOR X = 1 TO I
150 REM PRINT VNAME(X),NVL(X),NRD(X),D1(X),D2(I)
160 REM NEXT X
170 I = 1
180 OPEN "pats.dat" FOR INPUT AS #2
190 INPUT #2, TSTOCK,WDTH,PWDTH,PO
200 IF TSTOCK > 96 THEN TYPE$ = "Luaun" ELSE TYPE$ = "Pine/Oak"
210 LPRINT "P.O. Number ";:LPRINT USING "#####";PO:LPRINT
220 LPRINT "DATE: ";DATE$:LPRINT
230 LPRINT "Stock type: ";TYPE$
240 LPRINT:LPRINT"                    **** ";:LPRINT
    USING "###.##";PWDTH;:LPRINT" INCH BOARDS  ****"
250 INPUT #2, PATTERN(I),PL1(I),L1(I),PL2(I),L2(I),PL3(I),L3(I),
    PL4(I),L4(I),PL5(I),L5(I),PL6(I),L6(I),PL7(I),L7(I),PL8(I),
    L8(I),PL9(I),L9(I),PL10(I),L10(I)
260 IF EOF(2) THEN 290
270 I = I + 1
280 GOTO 250
290 CLOSE #2
300 LPRINT"    ";
310 LPRINT
320 LPRINT"Pattern"
330 LPRINT"No."
340 FOR X = 1 TO I
350 IF NVL(X) <> 0 THEN LPRINT:LPRINT "--------------------------"
360 IF NVL(X) <> 0 THEN LPRINT USING " ##    ";PATTERN(X)
370 IF PL1(X) AND NVL(X) <>0 THEN LPRINT "    Cut ";: LPRINT
    usING "###";PL1(X);:LPRINT USING"###.##";L1(X);:
    LPRINT" boards."
380 IF PL2(X) AND NVL(X) <>0 THEN LPRINT "    Cut ";: LPRINT
    USING "###";PL2(X);:LPRINT USING"###.##";L2(X);:
    LPRINT" boards."
```

```
390 IF PL3(X) AND NVL(X) <>0 THEN LPRINT "    Cut ";: LPRINT
    USING "###";PL3(X);:LPRINT USING"###.##";L3(X);
    :LPRINT" boards."
400 IF PL4(X) AND NVL(X) <>0 THEN LPRINT "    Cut ";: LPRINT
    USING "###";PL4(X);:LPRINT USING"###.##";L4(X);
    :LPRINT" boards."
410 IF PL5(X) AND NVL(X) <>0 THEN LPRINT "    Cut ";: LPRINT
    USING "###";PL5(X);:LPRINT USING"###.##";L5(X);
   :LPRINT" boards."
420 IF PL6(X) AND NVL(X) <>0 THEN LPRINT "    Cut ";: LPRINT
    USING "###";PL6(X);:LPRINT USING"###.##";L6(X);
    :LPRINT" boards."
430 IF PL7(X) AND NVL(X) <>0 THEN LPRINT "    Cut ";: LPRINT
    USING "###";PL7(X);:LPRINT USING"###.##";L7(X);
    :LPRINT" boards."
440 IF PL8(X) AND NVL(X) <>0 THEN LPRINT "    Cut ";: LPRINT
    USING "###";PL8(X);:LPRINT USING"###.##";L8(X);
    :LPRINT" boards."
450 IF PL9(X) AND NVL(X) <>0 THEN LPRINT "    Cut ";: LPRINT
    USING "###";PL9(X);:LPRINT USING"###.##";L9(X);
    :LPRINT" boards."
460 IF PL10(X) AND NVL(X) <>0 THEN LPRINT "    Cut ";: LPRINT
    USING "###";PL10(X);:LPRINT USING"###.##";L10(X);
    :LPRINT" boards."
470 NEXT X
480 LPRINT "--------------------------"
490 LPRINT:LPRINT
500 TSTRIPS = 0:TWASTE = 0
510 FOR X = 1 TO I
520 IF NVL(X) = 0 THEN 550
530 LPRINT "Use pattern  ";:LPRINT USING"###  ";X;
    :LPRINT "to cut ";:LPRINT USING"####"
    ;NVL(X);:LPRINT"strip(s)."
540 TSTRIPS = TSTRIPS + NVL(X)
550 NEXT X
560 TLENGTH = TSTRIPS * TSTOCK
570 FOR X = 1 TO I
580 LUSED(X) = PL1(X)*L1(X)+PL2(X)*L2(X)+PL3(X)*L3(X)+PL4(X)
    *L4(X)+PL5(X)*L5(X)+PL6(X)*L6(X)+PL7(X)*L7(X)+PL8(X)*L8(X)
    +PL9(X)*L9(X)+PL10(X)*L10(X)
590 WASTE(X) = TSTOCK - LUSED(X)
600 TWASTE = TWASTE + NVL(X)*WASTE(X)
610 NEXT X
620 UTIL = (1 - (TWASTE/TLENGTH))*100
630 LPRINT CHR$(12)
640 LPRINT:LPRINT:LPRINT:LPRINT"              ******
    CUTTING SUMMARY ";:LPRINT USING "###.##";PWDTH;:LPRINT"
    INCH BOARDS  ******"
650 LPRINT"-------------------------------------------------------
    ------------------------":LPRINT
```

```
660 LPRINT"This order requires a total of";:LPRINT USING
    "######";TSTRIPS;:LPRINT " strips to be cut.":LPRINT
670 LPRINT"This cutting requires ";:LPRINT USING"######.##"
    ;TLENGTH;:LPRINT " linear inches of stock material.":LPRINT
680 LPRINT"This cutting produces ";:LPRINT USING"#####.##"
    ;TWASTE;:LPRINT " linear inches of wasted material.":LPRINT
690 LPRINT "This cutting yields";:LPRINT USING "###.##";UTIL;
    :LPRINT "% strip utilization.":LPRINT
700 STPSHT = INT(WDTH/PWDTH)
710 SHEETS = (TSTRIPS/STPSHT)
720 LPRINT"This cutting requires ";:LPRINT USING "######.##";SHEETS;
    :LPRINT" sheets of stock.":LPRINT
730 EWASTE = WDTH - (STPSHT * PWDTH):EPCT = (EWASTE/WDTH)*100
740 LPRINT"Each sheet has ";:LPRINT USING "###.##";EWASTE;:LPRINT"
    inches of edge waste.":LPRINT
750 LPRINT"Edge waste creates ";:LPRINT USING "###.##";EPCT;
    :LPRINT"% sheet waste.":LPRINT
760 LPRINT"-------------------------------------------------------
    -----------------------":LPRINT
770 LPRINT CHR$(12)
790 SYSTEM
```

APPENDIX D. CUTTING ROUTINE BATCH FILE

```
cd\dos33
basica pattern3.bas
echo off
cd\lindo
LINDO < input2.lin
cd\dos33
echo on
basica result5b.bas
```

APPENDIX E. - LINDO EXECUTION BATCH FILE

```
take saw.mod
terse
go
sdbc nrvinfo.out
rvrt
quit
```

APPENDIX F. Sample Cutting Instructions

P.O. Number 999

DATE: 11-03-1991

Stock type: Luaun

**** 4.50 INCH BOARDS ****

Pattern
No.

2
Cut 3 20.25 boards.
Cut 1 28.25 boards.

5
Cut 1 14.25 boards.
Cut 4 20.25 boards.

8
Cut 3 14.25 boards.
Cut 1 20.25 boards.
Cut 1 28.25 boards.

11
Cut 5 14.25 boards.
Cut 1 20.25 boards.

Use pattern	2	to cut	1	strip(s).
Use pattern	5	to cut	95	strip(s).
Use pattern	8	to cut	618	strip(s).
Use pattern	11	to cut	18	strip(s).

****** CUTTING SUMMARY 4.50 INCH BOARDS ******

This order requires a total of 732 strips to be cut.

This cutting requires 70821.00 linear inches of stock material.

This cutting produces 3643.75 linear inches of wasted material.

This cutting yields 94.85% strip utilization.

This cutting requires 73.20 sheets of stock.

Each sheet has 3.75 inches of edge waste.

Edge waste creates 7.69% sheet waste.

P.O. Number 998

DATE: 11-03-1991

Stock type: Luaun

**** 6.50 INCH BOARDS ****

Pattern
No.

1
Cut 3 28.25 boards.

3
Cut 1 18.25 boards.
Cut 2 25.00 boards.
Cut 1 28.25 boards.

5
Cut 2 18.25 boards.
Cut 2 28.25 boards.

Use pattern 1 to cut 581 strip(s).
Use pattern 3 to cut 515 strip(s).
Use pattern 5 to cut 103 strip(s).

****** CUTTING SUMMARY 6.50 INCH BOARDS ******

This order requires a total of 1199 strips to be cut.

This cutting requires 116003.30 linear inches of stock material.

This cutting produces 7487.00 linear inches of wasted material.

This cutting yields 93.55% strip utilization.

This cutting requires 171.29 sheets of stock.

Each sheet has 3.25 inches of edge waste.

Edge waste creates 6.67% sheet waste.

P.O. Number 997

DATE: 11-03-1991

Stock type: Luaun

**** 6.75 INCH BOARDS ****

Pattern
No.

1
Cut 6 14.75 boards.

4
Cut 4 13.00 boards.
Cut 3 14.75 boards.

7
Cut 7 13.00 boards.

Use pattern 1 to cut 881 strip(s).
Use pattern 4 to cut 564 strip(s).
Use pattern 7 to cut 1 strip(s).

****** CUTTING SUMMARY 6.75 INCH BOARDS ******

This order requires a total of 1446 strips to be cut.

This cutting requires 139900.50 linear inches of stock material.

This cutting produces 7556.00 linear inches of wasted material.

This cutting yields 94.60% strip utilization.

This cutting requires 206.57 sheets of stock.

Each sheet has 1.50 inches of edge waste.

Edge waste creates 3.08% sheet waste.

APPENDIX G. USER'S MANUAL

NRV CUTTING SOLVER - QUICK REFERENCE GUIDE

STEP 1: Change directory to **F:\cut.**

STEP 2: Type 'cut3' and press the RETURN key.

STEP 3: The system will say,

"Enter the P.O. Number from the Order."

You should enter the number and press RETURN.

STEP 4: The system will say,

"Enter the width of the required pieces."

You should enter the width you are interested in solving.

For example : enter 4.5 (and RETURN)

STEP 5: The system will say,

"Enter the number of different lengths for the 4.50 width boards."

You should enter the number of different lengths for 4.50 inch boards on the order.

STEP 6: The system will say,

"Enter each different quantity and length."

You should enter data in the following format.

```
1000,14.25
 500,18.25
2000,20.25
```

This tells the system that you need 1,000 14.25 inch boards, 500 18.25 inch boards, and 2,000 20.25 inch boards.

STEP 7: The system will show you what you entered.

STEP 8: The system will ask you,

"Is this data correct?"

You should respond 'y' or 'n'.

STEP 9: If you responded 'n', the system will say,

"Enter the line number you wish to change."

You should type the line number and RETURN.

STEP 10: The system will then say,

"Re-enter the quantity and length now."

You should re-enter the data in the format used earlier.

STEP 11: The system will show you the data again and ask you if it is correct. You should use the preceding process for correcting additional lines of data.

STEP 12: The system will say,

"Enter the length of the stock material in inches."

You should type (for example) 96.75 RETURN.

STEP 13: The system will say,

"Enter the width of the stock material in inches."

You should type (for example) 48.75 RETURN.

STEP 14: Now the system will tell you,

"Generating solution. Please wait ..."

The system is now calculating the optimal cutting patterns to use, based upon the data you have entered. The next action will be for the system to print the cutting instructions and summary statistics for that cutting data.

To re-run the system for another width board, simple type 'cut' and press RETURN.

9. VITA

Mr. Jerome W. Hogge, III was born March 18, 1965 in Newport News, Virginia. Mr. Hogge graduated from Homer L. Ferguson High School in 1983. Mr. Hogge earned a Bachelor of Science degree in Electrical Engineering from Virginia Polytechnic Institute and State University in 1987.