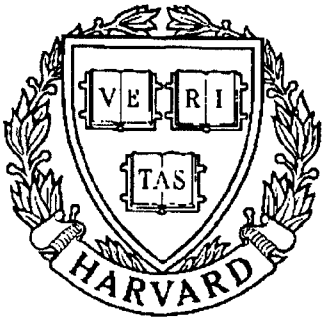
THESIS REPORT
Master's Degree

S Y S T E M S
R E S E A R C H
C E N T E R



Supported by the National Science Foundation Engineering Research Center Program (NSFD CD 8803012), Industry and the University

Heuristic Optimization of Rough-Mill Yield with Production Priorities

by M. Srinivasan
Advisor: G. Harhalakis

M.S. 90-13

HEURISTIC OPTIMIZATION OF ROUGH-MILL YIELD WITH PRODUCTION PRIORITIES

by

Murali Srinivasan

Thesis submitted to the Faculty of the Graduate School
of The University of Maryland in partial fulfillment
of the requirements for the degree of
Master of Science
1990

Advisory Committee:

Associate Professor George Harhalakis, Chairman
Associate Professor Shapour Azarm
Assistant Professor Ioannis Minis

ABSTRACT

Title of Thesis: Heuristic Optimization of Rough-Mill Yield With Production Priorities

Name of Degree Candidate: Murali Srinivasan

Degree and Year: Master of Science, 1990

Thesis directed by: Associate Professor George Harhalakis
Associate Professor Shapour Azarm
Department of Mechanical Engineering
University of Maryland
College Park, Md 20742

Efficient lumber utilization at the saw is a key issue in the woodworking industry because of shrinking supply and increasing raw material prices. In this thesis, the formulation of the cross-cut-first method of cutting defects out of lumber, as a one-dimensional stock cutting problem, is discussed, with the objective to maximize yield. Monte-Carlo simulations were used for generating boards of a given grade, to test and compare alternative solution procedures. A fast heuristic for solving the above problem is introduced to enable a real-time computerized implementation. The heuristic is shown to compare favorably with the algorithmic solution obtained, using Kolesar's knapsack algorithm, in terms of solution time and yield. The system is also capable of automatically assigning cutting priorities to reflect demand and production needs. In addition, the cut-rip defect removal strategy is introduced. The results of a set of experiments designed to evaluate the heuristic procedure as applied to the cut-off and cut-rip strategies are reviewed, in comparison to the operator's

performance. Finally, the implementation issues on an automated cut-off saw are presented.

ACKNOWLEDGEMENT

I wish to offer my sincere thanks to Drs. S. Azarm and G. Harhalakis for the helpful suggestions and the time they spent in discussions during the course of this project. This project would not have produced the fruitful results but for their constant guidance, criticisms, and the insights into the problem provided by them. I am deeply indebted to them for this.

I also wish to thank Dr. Minis for serving on the advisory committee and his helpful suggestions for improvement during his review of the thesis material.

I would also like to thank all my friends in the CIM lab at the University of Maryland, College Park for the help and encouragement they gave me during the preparation of the thesis material.

The work reported in this thesis was supported by the Maryland Industrial Partnerships (MIPS) program of the Engineering Research Center, and the Statton Furniture Company in Hagerstown, Maryland. This support is gratefully acknowledged. My special thanks go to Mr. Philip J. Statton, Mr. Dan Moore and Mr. Paul Hough from Statton Furniture Company for their many helpful suggestions and comments during the course of this project and the use of plant facilities.

Finally, I would like to thank my wife, Su, for her help and encouragement that enabled me to finish this work on time.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Wood has been a valuable material for a variety of constructional uses. It is easily worked upon with tools and machines. It has a large variety of species that provides a wide range of physical characteristics.

To some extent, wood has been replaced by many newer materials. But, at the same time, other new wood products have been added to the forest industry as a result of scientific research. These include particle board, hardboard and nonwoven wood cellulose fiber-based disposable products. The increasing demand for wood products will intensify the need for better utilization of what was formerly considered as unavoidable waste.

The annual consumption of wood in the United States is about 398 million cubic meters [1]. Two-thirds of this consumption is in the housing and nonresidential construction industries. Another third goes into manufacture of paper and related products. The supply of wood, against the increasing demand, is not limitless. About one-half of the earth was once covered by forests. Today, only about a third of the land surface is forested. Increased farming is one of the factors for deforestation. Fires consume more than three million acres of forests and woodlands every year - an estimate being one fire every four minutes. Insects and disease kill about one-fifth of the annual timber growth.

Wood is one of the least recycled of our resources. Only proper planning and forest conservation measures can ensure a never ending supply of wood. It is a natural resource and as such it will tend to become precious with increasing demand.

It is clear that improvements in the processes that utilize wood, with regard to better utilization, can play a significant role in conserving this natural resource. In the next section we take a look at one such process with potential for improvement- the method used for removing defects from rectangular pieces of wood of uniform thickness termed as boards.

1.1 Motivation and Objective

The fact that wood is a renewable natural resource adds to its value as a raw material. Properties of wood such as wide range of color, finishing characteristics and durability, make it an ideal raw material for a variety of uses. Methods to improve utilization of this scarce resource play an invaluable role in the profitability of the woodworking industry. This will also reduce the impact of depleting forest resources on the ecology by reducing the consumption of this raw material. One of the main areas open for improvement in utilization is the cutting process used for removing defects from boards. Automated cutting saws are increasingly used by the furniture and other woodworking industries to improve the productivity and efficiency of the cutting process. Hence, there is a pressing need for procedures that can help in fast and efficient decision making at the

saw. In this thesis, the emphasis throughout is on the one-dimensional cutting stock problem, as it applies to the furniture industry. The objectives are to develop a practical, time-efficient solution procedure that can be used on a real-time basis and to maximize the yield of the cutting operation at the saw. The heuristic solution procedures discussed in this thesis were developed based on these objectives.

The work was carried out with the support of the Statton Furniture Company located in Hagerstown, Maryland. The company manufactures traditional American furniture using cherry, oak and mahogany. The current manufacturing operation involving removal of defects from boards is labor intensive and gives poor yield. The company is presently introducing automated saws for the defect removal operation with a view to improve yield and increase volume output. The use of automated saws requires solution procedures that are fast in terms of solution time and provide reasonably good yield. The development of such a procedure is one of the main objectives of this thesis.

A brief section on the basics of the woodworking industry is presented next, to provide a basis for understanding the material to be covered in the following chapters.

1.2 A Brief Note on the Woodworking Industry

Wood cut from trees in a form suitable for manufacture of parts is termed as lumber. The production of lumber begins with logging.

Trees are cut down and further cut into suitable lengths. These are further cut into rectangular pieces of uniform thickness called boards. These boards have varying characteristics with regard to dimensions and appearance, because of the presence of material imperfections. These two factors determine the utility of a board for a given purpose. For some types of usage it may not be permissible to have any form of imperfections in a given piece of wood. This may not be an important factor in some other applications. Hence, the classification of wood into different grades, based on dimensions and imperfections, is necessary. The grade of wood that can yield a larger quantity of clear wood will have a higher monetary value. The classification of wood into grades aids the selection of a particular grade for a given application based on certain specifications and constraints (eg; appearance, cost, etc).

1.2.1 Grading of Lumber

Hardwoods are graded visually - each individual piece is inspected. Except in the case of "selects", grading is based on the poorer side of the inspected piece. The grader visualizes a number of defect-free rectangles, known as cuttings, on the surface of the board. The number of cutting units in each rectangle is computed by multiplying the length of the rectangle in feet by its width in inches. The grader also computes the number of cutting units in the board if it were defect free. The grade classification is then made based on grading specifications. In the USA, for example, these include the percentage area of the board that must be in clear cuttings of specified minimum dimensions. As a rule of thumb, the better the grade, the

fewer the number of defects and the longer the clear sections available on a board. The following is a general guide on various hardwood grades [2].

Hardwood grades:

Some of the commonly used hardwood grades are First, Second, Select, Number 1 Common and Number 2 Common. This classification, as mentioned before, is based on the dimensions of the board and the imperfections in it. The imperfections, also called defects, are covered in more detail in section 1.2.2. These imperfections determine the grade in an indirect way. The location of the defects on a given board determine the size of the defect free rectangles visualized by the grader. These dimensions, in turn, determine the grade of a board based on the grading specifications mentioned before. The following is a list of the commonly used hardwood grades:

First - 91% clear.

Second - 83-1/3% clear.

Select - Can be cut into two-feet lengths that are 91% clear.

Number 1 Common - 66-2/3% clear.

Number 2 Common - 50% clear.

It may be noted that First and Second are usually combined in one grade called FAS. The percentage of First required in the combined grade varies from 20 to 40 percent depending on the species of wood. Also, the percentages mentioned above refer to the percentage clear area of the board that must be in clear cuttings of specified minimum dimensions. A more detailed discussion of lumber grades can be found in [3].

1.2.2 Wood Defects

These are imperfections that reduce the strength of parts in case of softwoods or affect the appearance in case of hardwoods. These imperfections may be due to:

a. Growth, such as, knots, resin pockets, inclusion of bark;
b. Fungus and insect attack;
c. Torn grain and checks (cracks developed during the drying process).

Some of the defects are more common than the others. Defects also vary according to the species of wood.

1.2.3 Unit of Measure

Boards and planks are sold by the board-foot. A board-foot is equivalent to the volume of a board one inch thick, one foot long and one foot wide.

1.3 Problem Background and Statement

The wood that comes to the shop floor is in the form of rectangular stock of uniform thickness. The length and width, however, vary randomly from one board to the other. Boards, depending on grade, have a number of defects of various shapes and sizes. Defects have random locations on the boards. The ends of a board are trimmed to remove checks and the width is ripped to remove rough edges and achieve parallelism. The defects, for reasons of strength and

appearance, are removed from the board by a cutting process. The remaining clear wood (later termed as "clear section") is further subjected to a cutting process to produce parts of specified dimensions and quantities. The part dimensions and the quantity required are specified in a production document called a ticket. The cutting process, together with the grade of wood used, determines the yield of the process. Yield here is taken as the ratio of the volume of parts cut to the volume of raw wood processed.

Two methods are mainly used for cutting defects out of boards:

(i) cut-off-first

(ii) rip-first

In the first method, the defects are cut out by saw cuts that are perpendicular to the length of the board. The portion of wood between two clear sections constitutes waste (see Fig. 1(a)). These clear sections may be either ripped to obtain parts of smaller width, or they may be glued-up to make parts of larger width.

In the second method, defects are ripped out by saw cuts parallel to the length of the board. The strips so produced are subjected to a cut-off operation as in (i) to obtain parts of required length. Further ripping or gluing may be necessary to get the finished dimensions of a part (see Fig. 1(b)).

In the above two methods, all defects on a board are either cut-off or ripped. In a proposed third method, called "cut-rip", both methods (i) and (ii) are considered in arriving at a decision to cut out a defect or leave it on the board for a later operation to remove it (see Fig. 1(c)). In this method, just as in the case of cut-off-first, the defects are

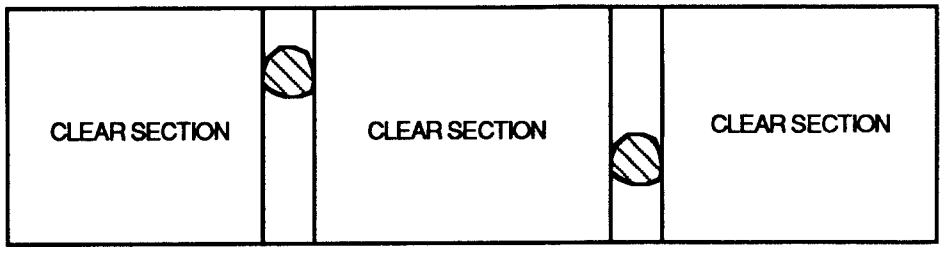


Fig. 1(a) Cut-Off

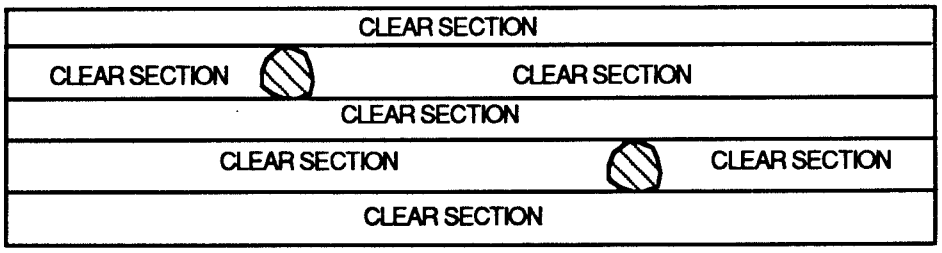


Fig. 1(b) Rip-First

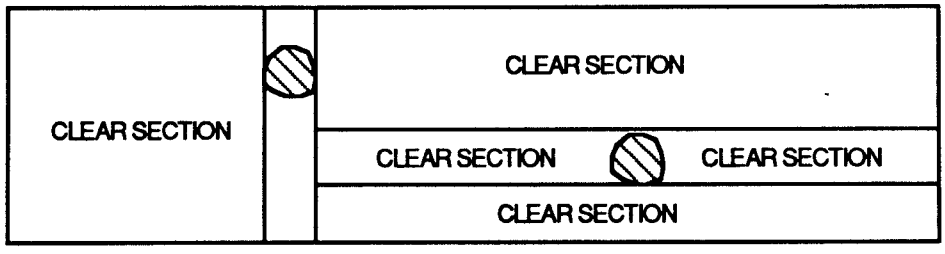


Fig. 1(c) Cut-Rip

considered one at a time. In order to make a cut-off or rip decision for a defect, only a part of the board is considered at a time as shown in Fig. 1(d). The part of board considered contains only the defect for which a cut-off or rip decision is to be made. The cut-off-first and rip-first strategies are applied one at a time to the part of board considered. The resulting yields are compared. Based on the higher yield, a decision is made to cut-off the defect or leave it on the board to be removed by a later ripping operation. For example, in Fig. 1(d), the part of board considered for defect number 2 will be as shown if defect number 1 is assumed to be cut off. The procedure is repeated for the remaining defects on the board.

In all three cases, the basic objective is to produce rectangular parts of specified length, width and thickness. Also, clear sections in all three cases are rectangular areas on the given board free of any defects.

The defect removal operation using an automated cut-off saw can be based on the strategies described above. An important factor to be considered here is the speed of deriving a solution, to avoid machine idle time. This was one of the main considerations for the methods developed in this thesis.

In this thesis only the formulation and solution of the first method, namely the cut-off-first, is considered. However, due to the way the clear sections are defined, this solution procedure is applicable to the cut-rip method and the rip-first methods as well. In case of cut-

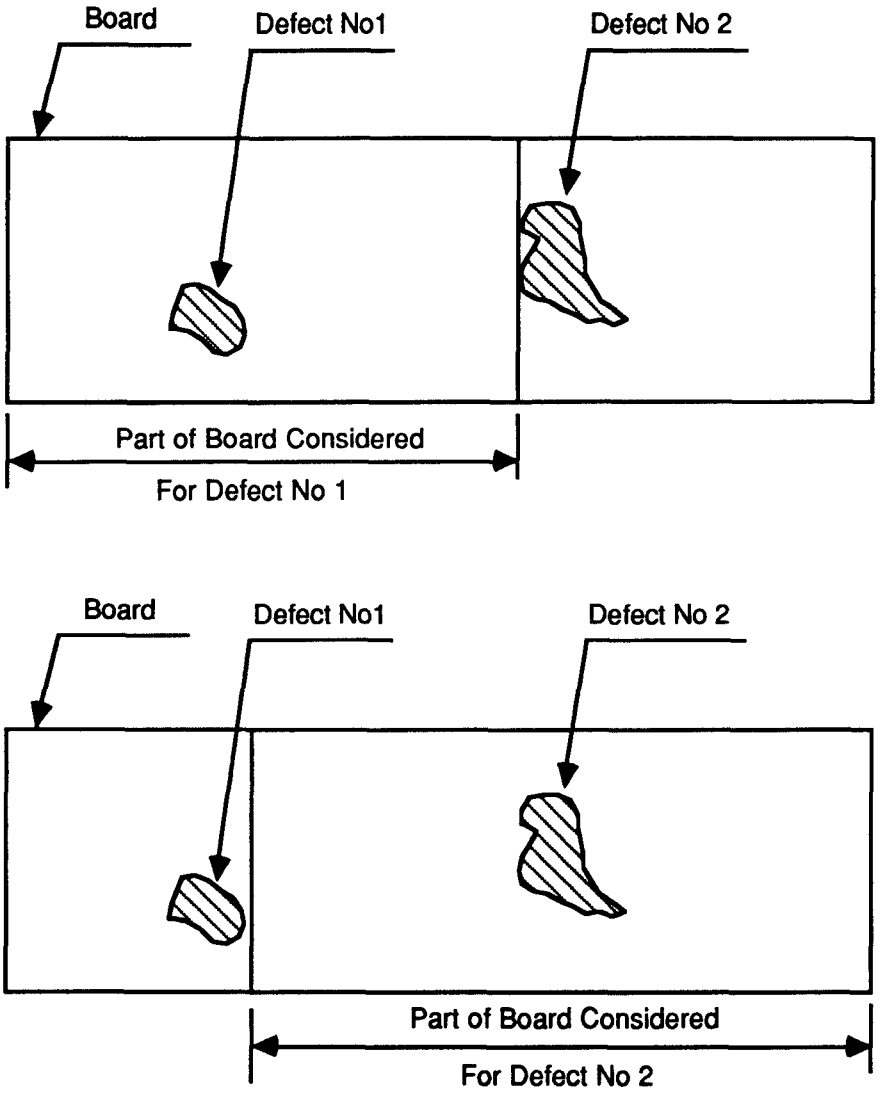


Fig. 1(d) Examples for Parts of Board Considered in the Cut-Rip Strategy

off-first method, saw cuts are assumed to be perpendicular to the board length, regardless of the shape of the defects. The problem then is to determine the positions of the saw cuts along the length of the board, taking into account the ticket lengths, so that yield is maximized.

The various methods of solution that have been reported in the literature to solve this and similar problems are examined in the next section.

1.4 Literature Survey

The one-dimensional problem of cutting a length of material into smaller lengths, so as to maximize utilization of material, gives rise to the well-known knapsack problem. An extension of the problem along the width of the stock results in the two-dimensional stock cutting problem. Both of these problems have been solved using linear programming [4], dynamic programming [5,6] or branch and bound method [6,7,8].

In [4], the cutting stock problem is formulated as a linear programming problem. Here it is assumed that a stock of standard length, in unlimited quantity, is available. The problem is to cut a specified number of pieces of specified lengths from the available stock. A cost is assigned to each piece of the standard length. The cost associated with filling an order is the total cost of the stock lengths cut to satisfy the order. The problem is to satisfy the requirements for the various lengths at the least raw material cost, subject to the constraint of demand for various lengths.

There are two disadvantages with this formulation. The first is the large number of variables, i.e., the number of times a specified length is to be cut on a given stock length. The second is the restriction that the solution variables have to be integers. In [4], the first disadvantage is handled by using dynamic programming and the second by relaxing the constraint on the variables to be integers.

The problem of cutting sheets with defective areas into given pieces, while minimizing waste is treated in [5,6]. The sheets, the pieces, and the defects are all rectangles. The problem of cutting irregular shapes from stock of fixed size is also treated in an approximate manner, by enclosing the irregular forms in rectangles. Here, each piece to be cut is assigned a value. Dynamic programming is used to generate the optimum patterns. The objective is to fit the given pieces into the clear portions of the sheet in such a way, that the total value of the cut pieces is a maximum.

The application of branch and bound method to the one-dimensional and two-dimensional problems is treated in [6,7,8]. A branch and bound algorithm proceeds by repeatedly partitioning the class of all feasible solutions into smaller sub-classes, called nodes, in such a way, that ultimately an optimal solution is obtained. The starting node for the algorithm contains all feasible solutions. The branching from a node is done so that each feasible solution belongs to exactly one sub-class. The maximum value of the objective function attained by the sub-class of feasible solutions at a node is termed as the upper bound at a node. The value of the upper bound is used to choose the node from which further branching will take place. If the solution

at a node is infeasible after branching, the node is pruned. The method is constructive in that it generates smaller and smaller sub-classes, some of which will contain only one feasible solution, one of which will be optimal. The procedure is time consuming. It can be effective for handling cutting problems of very small size.

Several techniques for this problem have been reported in the literature using heuristics [9,10,11,12,13,14]. Some of these methods have been based on an exhaustive search of feasible solutions [10,11,13,14]. In other cases, weighing factors are used to confine the search to a small subset of feasible solutions [9,12]. In case of [10,11,13,14], the one-dimensional cutting stock problem is solved using a combinatorial heuristic. The problem is to cut given rectangular pieces on a rectangular board to maximize the value or yield, depending on the objective function. In case of the OPTYLD program, the board is first cut into strips, using the best combination of piece widths that can be cut within the board width. These strips are further cut into the given lengths. In case of the CROMAX program [11], the objective is to cut the given lengths within the board length to optimize yield. In both cases, the solution is arrived at by examining all combinations of the piece widths or lengths. The method has the serious disadvantage that even with problems of moderate size the number of combinations to be examined to reach the optimal solution may be very large. This factor limits the practical utility of the method. In case of [9], the one-dimensional stock cutting problem is solved by the use of weighting factors. Rectangular areas on the board free of defects are identified in the first stage. The area to be cut out of the

board is selected based on the weighting factor assigned to the length of the clear area.

In case of [12], the lengths to be cut are selected from the list of piece lengths to be cut, ordered by the magnitude of the piece lengths. The first feasible length in the list is cut and the length of the board is reduced by the length of the piece just cut. The procedure is repeated for the remaining lengths.

The major drawback of these methods has not been in the quality of the solutions produced, but in the applicability of these methods. In particular, these methods are not well suited for real time implementation either because of the excessive solution time or because of the nature of the solution obtained. For example, the CROMAX program [11] requires 5 minutes or more of solution time to process an eight feet board (on UNIVAC 1100/80). The actual processing time, however, for a similar size board on an automated cutting saw is desired to be in the order of a few seconds. This difference in the computational versus the physical processing time imposes considerable machine idle time.

In the case of the YIELD program [9], the program specifies the size and location of rectangular clear areas on the board with respect to the lower left corner of the board, in a way that maximizes yield. In order to cut these clear areas from the board, the cut-off or the rip-first method (explained in section 1.3) may have to be used, depending on the location of the clear areas. The program, however, does not consider the cutting of tickets on these clear areas. The yield in [9] is computed as the ratio of the total clear area to the total area of the

board. Hence such a program can only be used to determine the maximum cutting yield for a particular grade. The yield after the tickets are cut on these clear areas will be lower than the maximum yield so determined and will depend on the tickets to be cut and the cutting strategy.

1.5 Thesis Organization

The remainder of this thesis is organized as follows. First, the technique for generation of simulated boards to facilitate the evaluation of alternate solution procedures is discussed in Chapter 2. In Chapter 3, the one-dimensional stock cutting problem is formulated, followed by the solution procedure using Kolesar's algorithm. A time-efficient heuristic is next introduced for the solution of this one-dimensional problem. This heuristic is further modified to include cutting priorities in the solution procedure. Next, the experimental test results of a set of experiments that were used to evaluate the solution procedure, are presented in Chapter 4. This is followed by a presentation and discussion of results in Chapter 5. Finally the conclusions and recommendations for further work are presented in Chapter 6.

CHAPTER 2
SIMULATION

In this chapter, the use of simulation as an important tool to evaluate different solution strategies is presented. A discussion of the Monte Carlo technique and a method to validate the simulation model are is also included.

2.1 Motivation

The evaluation of different solution strategies involves considerable expense in terms of material, labor and machine time costs. In order to carry out this evaluation in an inexpensive and efficient manner, a simulation approach for generating boards of a given grade had to be adopted. This was done using the well-known Monte-Carlo method [15]. The simulation of boards has certain distinct advantages. It can be used to evaluate the effect of different process variables such as, ticket length, lumber grade and solution strategy on yield. It can also be used to generate the same set of boards repeatedly by using the same set of random numbers during different simulations. Hence comparisons of different solution strategies can be made quantitatively. The effect of mixing two or more grades in pre-specified proportions can also be evaluated. The savings in time and material required for such evaluations are considerable.

2.2 Variables Used in Simulation

The simulation of boards, of a specified grade, was carried out using seven variables (Fig.2):

1. Board length (L)
2. Board width (W)
3. Number of defects (N)
4. Relative position of the center of the defect along the board length (X'=x/L)
5. Relative position of the center of the defect along the board width (Y'=y/L)
6. Defect length (a)
7. Defect width (b)

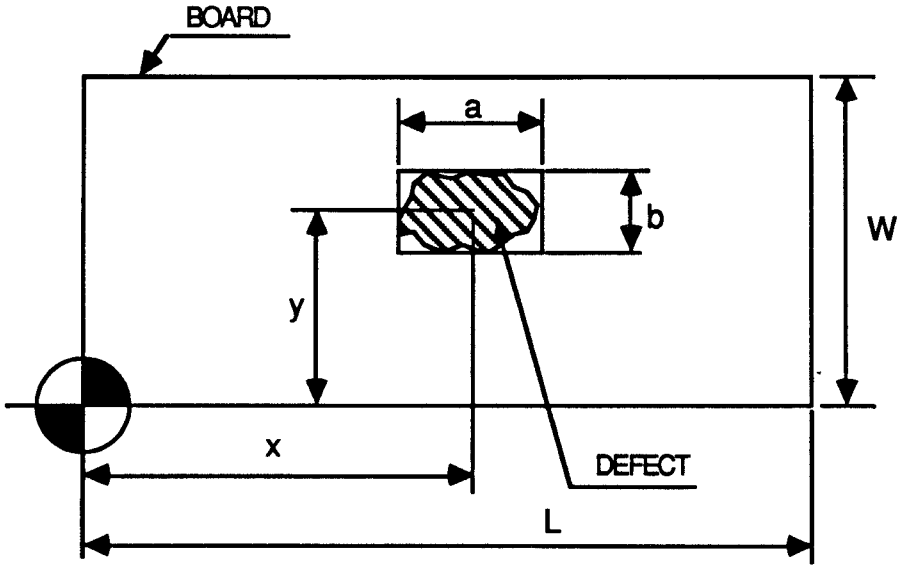


FIG. 2 Variables Used in Monte-Carlo Simulation

The lower corner of the board was chosen as the origin with the +x axis along the board length and the +y axis along the board width. The defects, regardless of their shape, were treated as rectangular. This was done by enclosing the contour of the defects in a rectangle with edges parallel to the edges of the board. This does not affect the yield of a board, since the saw cuts are always parallel to the edges of the board. Also, since boards have defects on both sides, the defect coordinates were measured with respect to one corner of the board. This, in effect, transfers all the defects to one face of the board. As defined above, the relative position of a defect (X', Y') is defined as the ratio of the actual coordinate (x or y) to the board length or width, respectively. The choice of the relative position as a variable ensures that a simulated defect stays inside the boundary of the simulated board.

2.3 Monte Carlo Simulation

Actual data collected for 160 boards of No.1 common and 200 boards of No.2 common grades for the seven variables were used to generate simulated boards. The procedure for simulating one of the variables, board length, is described below:

The frequency distribution for the board length was obtained from the data collected for the actual boards. A set of consecutive integers was assigned to each class interval in the variable range (see Table 1). The number of integers assigned to each interval was equal to

Mid Point of Class Interval For Actual Board Lengths (Inches)	Frequency	Random Number(s) Assigned to the Class Interval
33	3	0 - 2
39	1	3
41	2	4 - 5
43	1	6
51	1	7
55	1	8
57	1	9
95	1	10
99	1	11
107	1	12
119	5	13 - 17
121	39	18 - 56
123	9	57 - 65
125	1	66
129	1	67
139	1	68
141	1	69
143	10	70 - 79
145	22	80 - 101
147	3	102 - 104

Table 1 Monte Carlo Simulation For Board Length L (Number 1 Common)

the frequency of the variable . For example,the first class interval of Table 1 with a mid point value of 33 inches for the board length and having a frequency of 3 would be assigned integers 0,1 and 2. The procedure is repeated for all the intervals in the range of the variable.

The simulation of a variable is done by generating a random integer value within the range of the set of integers assigned to that variable. The value of the variable for the class interval that contains the generated random integer in the assigned integer set is taken to be the simulated value. As in the previous example, if the random integer is 2, then the simulated value for the variable length is 33 inches.
This procedure is used for the seven variables to generate the simulated values for a given board. It is important to note that use of the same set of random digits yields the same set of simulated values. This fact can be used in generating the same set of simulated boards repeatedly, for the purpose of comparisons of alternate solution procedures.

This simulation procedure was used to generate No.1 common and No.2 common grades, given the actual frequency distributions for the seven variables. Graphs showing the comparison of actual and simulated frequency distributions for the seven variables used to define a simulated board are shown in Figs. 3 (a)-(g). In these graphs, the values of the variable under consideration is plotted on the x-axis and the frequency of the variable values on the y-axis. The values plotted on the x-axis correspond to the values of the class intervals. The measured frequency of the variable values corresponding to the actual

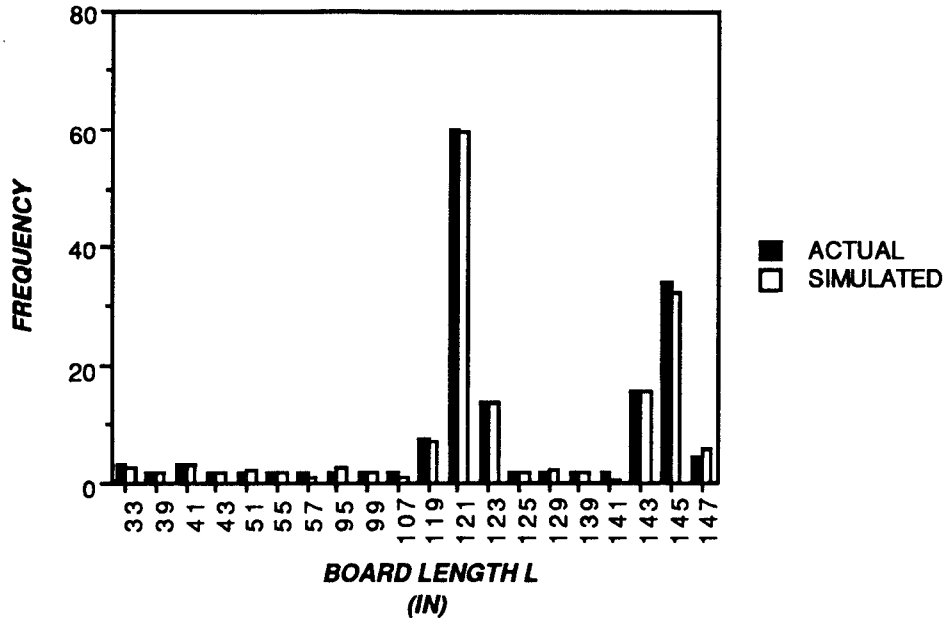


FIG. 3 (a) Comparison of Actual and Simulated Frequency Disributions for Variable L

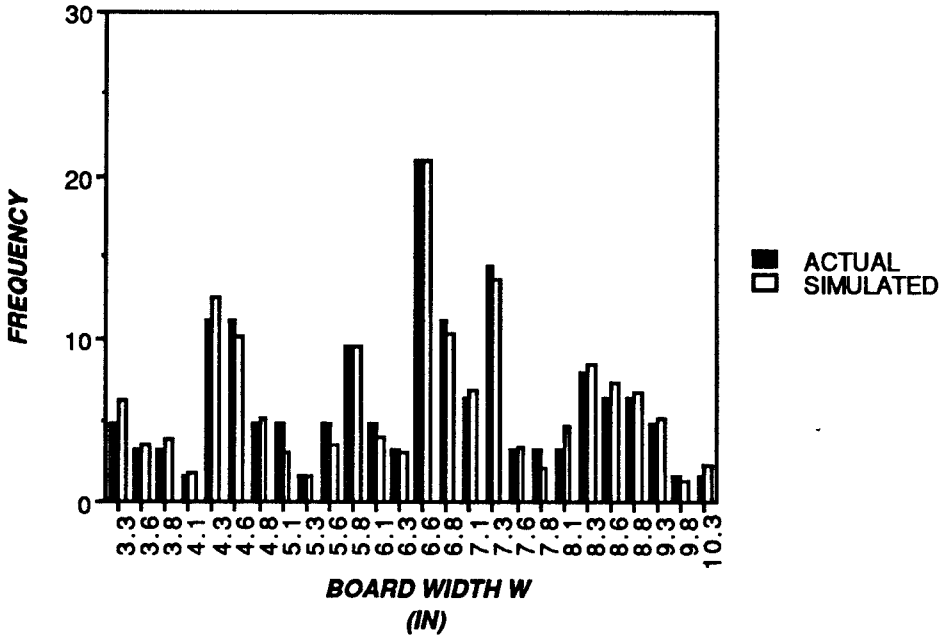


FIG. 3 (b) Comparison of Actual and Simulated Frequency Disributions for Variable W

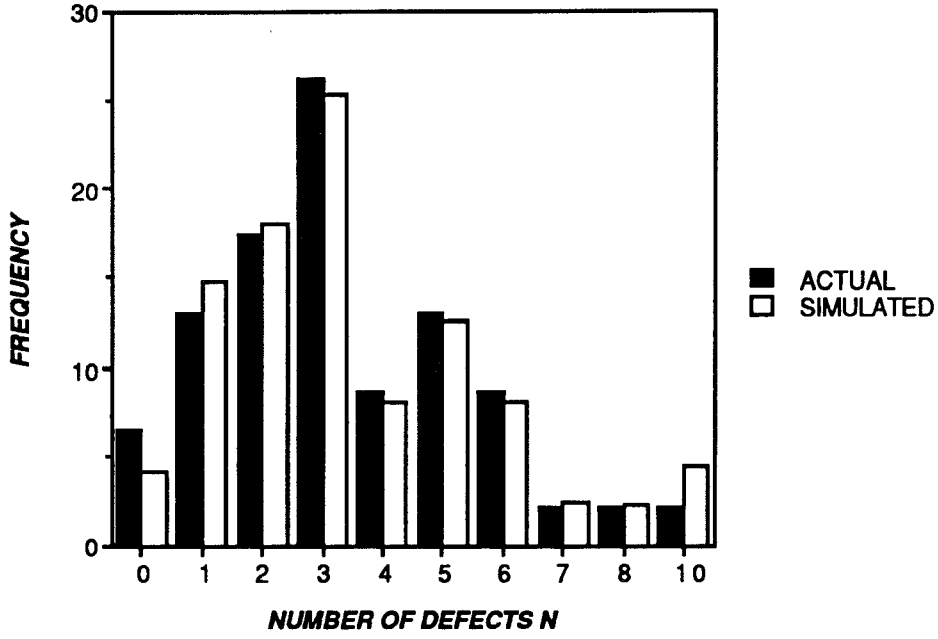


FIG. 3 (c) Comparison of Actual and Simulated Frequency Disributions for Variable N

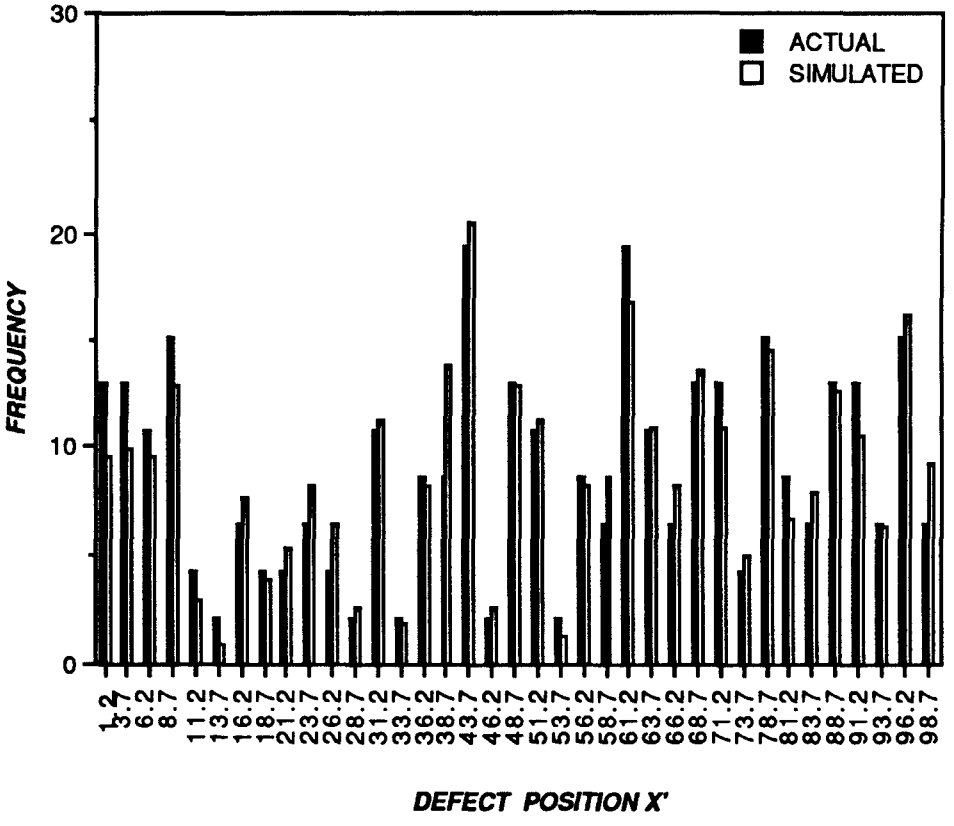


FIG. 3 (d) Comparison of Actual and Simulated Frequency Disributions for Variable X'

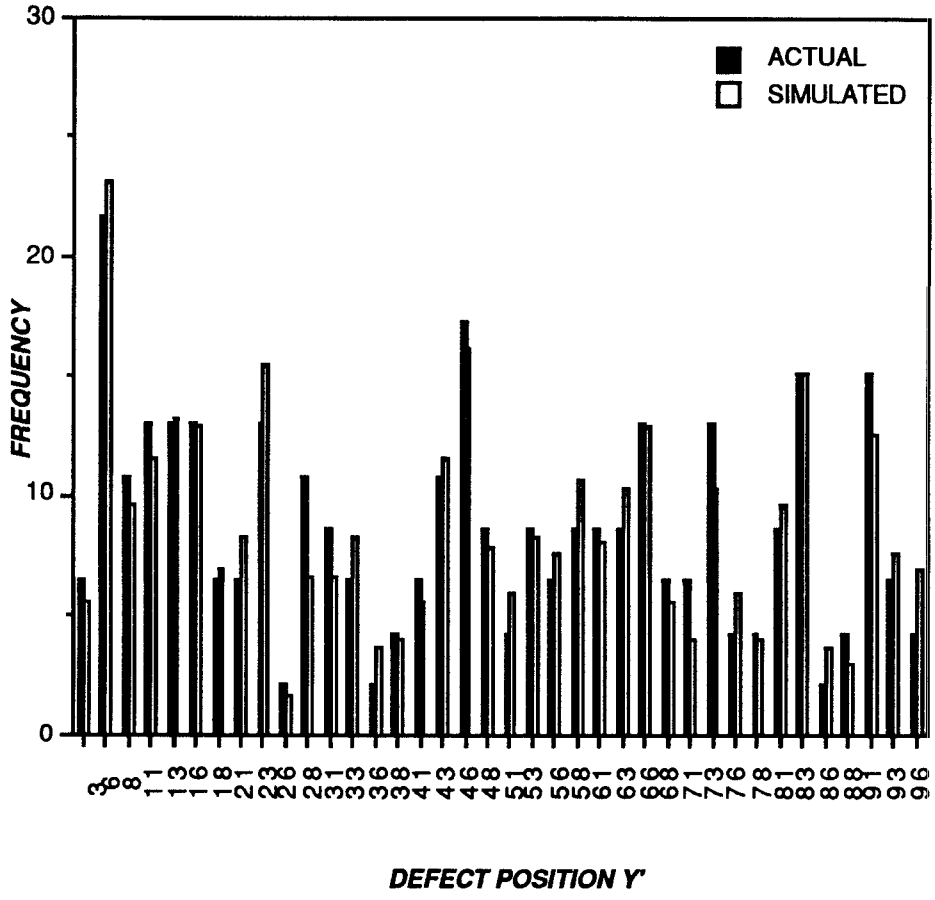


FIG. 3 (e) Comparison of Actual and Simulated Frequency Disributions for Variable Y'

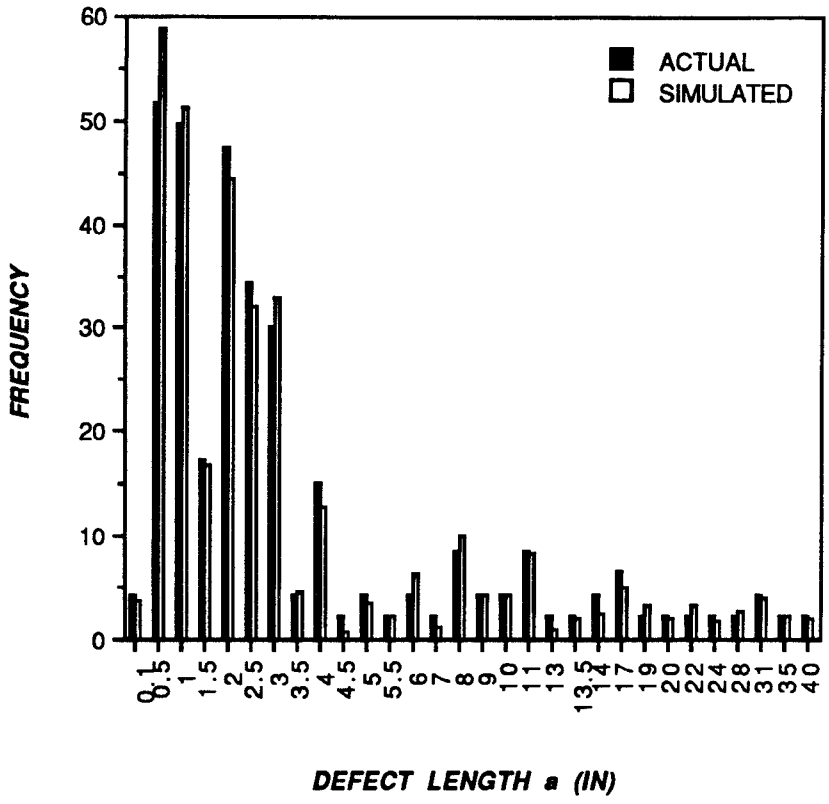


FIG. 3 (f) Comparison of Actual and Simulated Frequency Disributions for Variable a

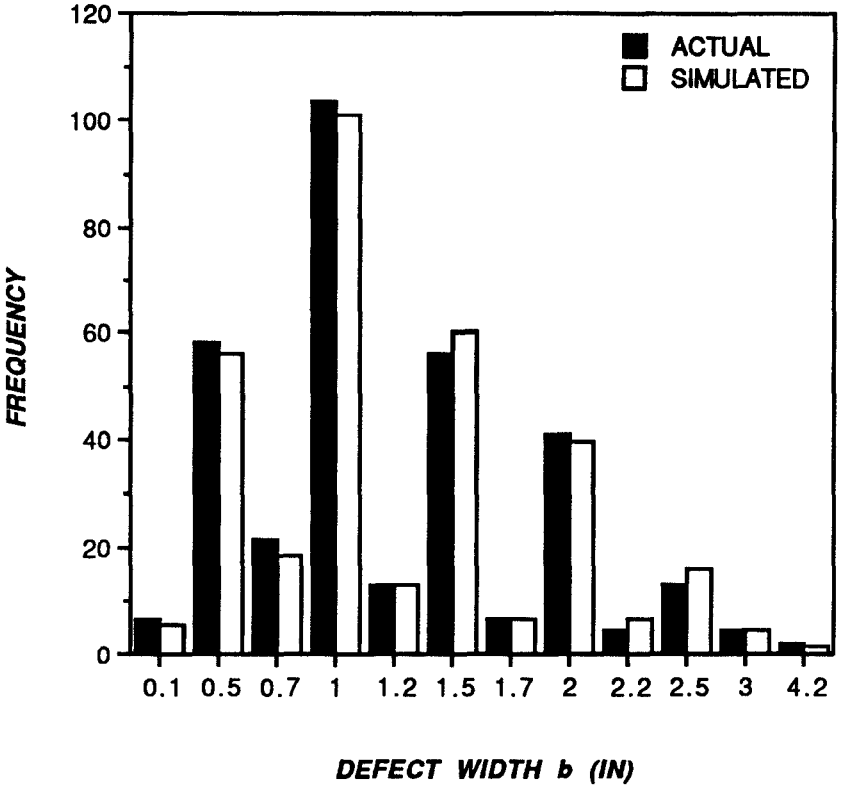


FIG. 3 (g) Comparison of Actual and Simulated Frequency Disributions for Variable b

sample of the boards is shown by the black bars. The corresponding frequency for the simulated variable value is shown by the white bars. The closer the two are, the better the "fit" of the simulated boards to the actual boards.

2.4 Validation of the Simulated Boards

A visual evaluation of the graphs in Figs. 3 (a)-(g) can be confirmed by the χ^2 values (explained below) for the variables in Table 2. It may be seen that better the "fit", lower will be the χ^2 value for that variable. It may be seen that in case of variables L,N,b the distributions have a better match and correspondingly a lower χ^2 value.The chi-square value (χ^2) for a given distribution is given by,

$$\chi^2 = \sum_{j=1}^{J} [(F_{aj} - F_{sj})^2 / F_{sj}] \qquad (1)$$

Here, F_{aj} is the actual frequency of a given variable value in the class interval j and F_{sj} is the simulated frequency for the same variable value in the same interval j. The summation is over the total number of intervals J into which the variable range has been subdivided. The chi-squared values for the actual and simulated distributions agreed well for a 5% significance level [Table 2]. The first χ^2 column of the table lists the chi-squared values for the simulated distributions. The second column lists the maximum chi-squared values taken from a standard table [16] for a 95% probability that the two distributions agree.

VARIABLE	χ^2	$\chi^2_{(MAX)}$
LENGTH	1.673	26.296
WIDTH	3.388	33.924
NUMBER OF DEFECTS	1.252	18.307
X' (x/L)	2.880	53.358
Y' (y/L)	3.225	49.758
a	2.867	42.557
b	0.976	15.507

TABLE 2 Comparison of Chi-Square Values for the Simulation Variables

The chi-squared values for the simulated distributions can be seen to be substantially less than the recommended values in the second column. Hence, the "goodness of fit" of the two distributions is considered to be satisfactory.

Also, the clear section lengths on the simulated boards and their yield (for a simulated cutting of a set of tickets) closely agreed with the figures for actual boards of the simulated grade, as seen in Chapter 4.

A plot of simulated boards made using AUTOCAD [17], is shown in Fig.4. As mentioned before, the simulated boards bear all defects on the top surface.

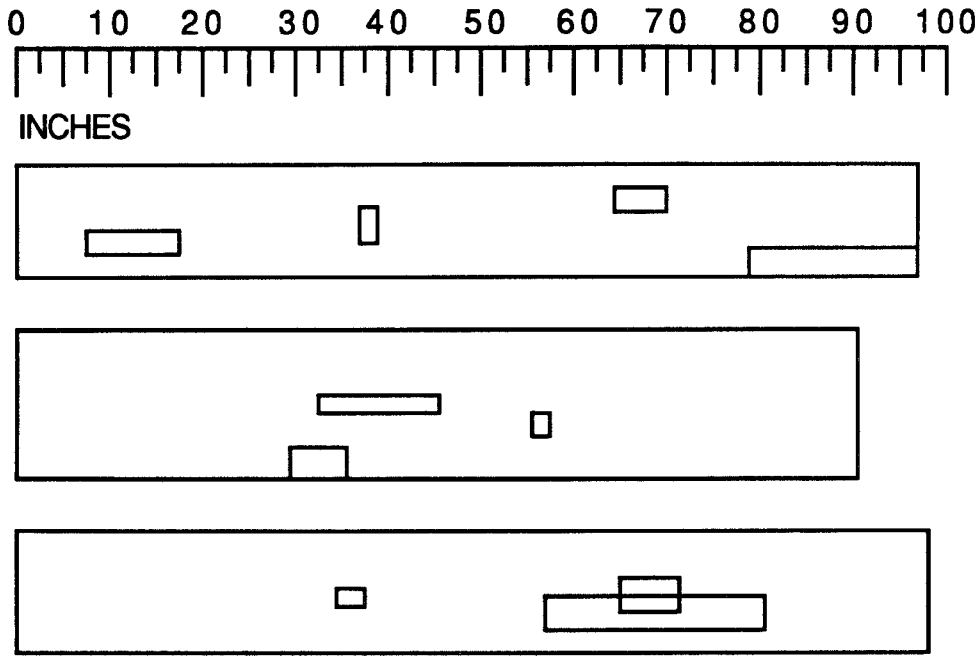


FIG. 4 Sample Plots of Simulated Boards

CHAPTER 3
FORMULATION AND SOLUTION PROCEDURE

In this chapter, the formulation and solution of the one-dimensional cutting stock problem, using the knapsack formulation is presented. Next, a time-efficient heuristic is introduced to solve the above problem. Finally, the need for cutting length priorities and their inclusion in the solution procedure is discussed.

3.1 Problem Formulation

The one-dimensional stock cutting problem is to determine the ticket lengths and quantities to be cut on a clear section, to maximize material utilization or yield. However, practical issues, such as ticket priority or value of the pieces cut, give rise to other objectives as well. These are:

(a) maximization of total value of pieces cut on a clear section

(b) maximization of volume of wood cut for tickets with priority.

In the formulation to follow, the objective is to maximize the total length cut on a given clear section, subject to the conditions that the total length cut cannot exceed the clear section length, and the volume requirements for the tickets are satisfied.

To formulate the problem, let:

BFC_k = Cumulative volume of wood cut for ticket including the section under consideration

BF_k = Volume requirement of ticket k

Int = Integer function to transform a number to an integer by truncating the fractional part

I_k = Maximum number of integer pieces of ticket k that can be cut on the clear section of length SL, i.e., I_k=Int (SL/TL(k))

k = Ticket number (k = 1,...,K)

L_{ki} = Length of material (wood) required to accommodate i pieces of ticket k (i= 1,...,I_k)

SL = Length of clear section

TL(k) = Ticket length for ticket k

W = Width of clear section

X_{ki} = Binary variable: 1 if L_{ki} is cut; 0 otherwise

The objective f is to maximize the total length cut on a clear section :

$$\text{maximize } f = \sum_{k=1}^{K} \sum_{i=1}^{I_k} L_{ki} X_{ki} \qquad (2)$$

subject to

$$\sum_{k=1}^{K} \sum_{i=1}^{I_k} L_{ki} X_{ki} \leq SL \qquad (3)$$

i.e., the sum of lengths cut for each of the tickets on a given clear section can not exceed the length of the clear section.

$$BFC_k \leq BF_k \qquad (k=1,...,K) \tag{4}$$

i.e., the cumulative volume of wood cut for a ticket cannot exceed the volume requirement for that ticket.

$$X_{ki} = 0 \text{ or } 1, \qquad (i=1,...,K) \quad (k=1,...,K) \tag{5}$$

i.e., $X_{ki} = 1$ if L_{ki} is cut; 0 otherwise.

3.2 Algorithmic Solution

Equations (2)-(5) constitute a (0-1) integer linear programming problem that can be solved by a number of algorithms. The number of binary variables involved depends on the number of tickets and the clear section length. In general, the number of binary variables n is given by the summation of the maximum number of integer pieces that can be cut for the K tickets.

$$n = \sum_{k=1}^{K} \text{Int}\,(SL/TL(k)) \tag{6}$$

Hence, the number of binary variables is a function of section length and ticket lengths. For No.1 common grade boards, the number of variables involved with ten tickets is usually between 20 and 40. The

clear section length range is from 16" to 143". The ticket lengths are assumed to be in the range of 16" to 80". The number of variables increases rapidly with increasing number of tickets or section length. This adversely affects the solution time. However, this problem can be formulated as a knapsack problem. A typical knapsack problem involves the loading of a knapsack with a set of items or objects. Each item has a weight and a value associated with it. The objective is to load the knapsack with minimum weight while maximizing the value of the items loaded.The present problem can be formulated as a knapsack problem in which the weight and value are synonymous. This fact can be taken advantage of to solve this problem with an algorithm that is faster than general integer programming algorithms. One such algorithm was proposed by Kolesar [7].

3.2.1 Knapsack Formulation

In the above formulation, an integer number of cuts of a ticket length is treated as representing the items of the knapsack problem. The basic assumption is that every clear section is used to produce only one of the k tickets. In other words, tickets having the same length but different widths are not cut on the same clear section. This is a somewhat limiting assumption in that, when a ticket requirement is about to be finished, part of a clear section may remain unutilized. The weight and value of the items are synonymous with the length of cut. For example, if it is possible to cut three tickets of length 16" each on a clear section, the weight and value of the three items corresponding to

this ticket would be 16, 32 and 48. The main advantage of this formulation is that the number of binary variables in the objective function and the constraints are kept to a minimum, thereby improving the solution time. In using this formulation with the simulated boards, the clear sections on a board were solved one at a time for the tickets to be cut. Whenever the section width exceeds the part width, the parts are assumed to be ripped from the clear section. In contrast, if the part width exceeds the section width, the parts are assumed to be produced by gluing-up sections of same length. The cumulative total width cut for each ticket was used to keep track of the cumulative volume of wood cut for each ticket. The constraint of ticket requirements was thus included in the solution process. Whenever a ticket requirement was satisfied, it was replaced by a new ticket.

3.2.2 Kolesar's Algorithm

In the following paragraphs, the steps of the algorithm as it applies to the above formulation are presented. A general description of this algorithm, as it applies to a standard knapsack problem, may be found in [7].

The Kolesar's algorithm uses the branch-and-bound technique. The algorithm proceeds by repeatedly partitioning the class of all feasible solutions into smaller subclasses in such a way that ultimately an optimal solution is reached. This partitioning or branching is done so that each feasible solution belongs to exactly one subclass. For each subclass or node j, an upper bound B(j) to the maximum value of the

objective function is computed. Based on this upper bound, further branching is carried out. Next, upper bounds are calculated for the new branches. The process is repeated until a feasible solution is obtained that gives a value to the objective function greater than the greatest value of all previous solutions.

The following provides the definition of certain terms required for the understanding of the algorithm :

Feasible Solution: A collection of X_{ki} satisfying equations (3)-(5).

Item: Length of cut equal to an integer multiple of the ticket length. A length of cut is considered as an item only when the length of cut is less than the clear section length.

ϕ: The symbol for null or empty set.

Included Items: Set of items explicitly included in the solutions at a node j, denoted by I(j).

Excluded Items: Set of items explicitly excluded from solutions contained in node j, denoted by E(j).

Free Items: Items that do not belong to I(j) or E(j) and have not yet been specifically assigned. This set is denoted by F(j). When this set is empty at a node, the node contains only one solution and further branching from that node is impossible.

Terminal Node: A node that has no branches emanating from it.

The items are first ordered in descending order of magnitude. The feasibility of the solution at the node is checked. If constraint (3) is violated, no further computations are made for this node. Otherwise,

the upper bound at a node is calculated by relaxing the constraint (3) for the free items. The items included at this node are cut first. Then all free items in the ranked list are included one at a time until (3) is exactly satisfied or until there are no more free items. The objective function value at this stage is the upper bound for this node.

The branching operation is carried out based on two decisions. The first decision is to select the node from which the branching is to be done. In the branch-and-bound method, the node with the highest bound is selected for branching. The second decision is to select the item that will be included at the first and excluded at the second of the two descendent nodes. This selection is arbitrary and the free item that is highest in order in the ranked list is selected.

The following steps outline the operations and decisions involved in the algorithm:

Stage 1:

(a) Test the nontrivial feasibility of the problem by verifying for at least one index ki (k=1,..,K ; i=1,..,I_k)

$$L_{ki} \leq SL \qquad (7)$$

If the problem is nontrivially feasible, proceed to (b). If not, stop.

(b) If $\sum_{k=1}^{K} \sum_{i=1}^{I_k} L_{ki} \leq SL$, then all the items may be cut and the problem is trivial.

If not, proceed to (c).

(c) Rank the items in decreasing magnitude.

(d) For node #1 set B(1)= ϕ, I(1)= ϕ, E(1)= ϕ. Proceed to Stage 2.

Stage 2:

(a) Find the terminal node with the largest value of B(j). This is the node from which further branching is done.

(b) Test if at the current node j, F(j)= ϕ. If so, an optimal solution is given by the items contained in I(j). If not, select a new free item i*, highest in order from the ranked list for further branching.

Stage 3:

(a) Set j=j+1, I(j)=I(j-1), E(j)=E(j-1) $\cup$ (i*). This means that for even nodes, the items included in the parent node are carried over, and the item i* is added to the list of excluded items. Proceed to (c).

(b) Set j=j+1, I(j)=I(j-1) $\cup$ (i*), E(j)=E(j-1). This means that for odd nodes, the item i* is added to the list of included items, and the items excluded from the parent node are carried over. Proceed to (c).

(c) Test the feasibility of the solutions contained in node j by verifying if the sum of the items included exceeds the section length. If infeasible, set B(j)= - 1000 or any large negative number. This will ensure that this node is eliminated from further branching decisions. Otherwise, compute the upper bound B(j) by first including the items in I(j). Then proceed in sequence to include the free items, one at a time, until the total length of the items is exactly equal to the section length or all free items are exhausted. The summation of the item lengths is B(j). If the node is even, go to 3(b). Otherwise, go to Stage 2.

Example: The operation of the algorithm is illustrated with a simple example. We consider a clear section of length, SL=60", and two tickets of lengths, TL(1)=28" and, TL(2)=25". The maximum number of pieces that can be cut for tickets 1 and 2 is I_k =2 (k=1,2). The item lengths for ticket 1 are 28" and 56". Item lengths for ticket 2 are 25" and 50".

Stage 1:

(a) Since at least one item length is less than the section length, the problem is nontrivially feasible.

(b) Sum of the item lengths 56+50+28+25 > 60. Proceed to (c).

(c) The list of items ordered by magnitude is 56, 50, 28 and 25.

(d) Set B(1)= ϕ , I(1)= ϕ , E(1)= ϕ . Proceed to stage 2.

Stage 2:

(a) Since only node No.1 exists, branching will be carried out from this node.

(b) None of the four items has yet been assigned. Hence, the set of free items is F(1) = {1,2,3,4}. Select item No.1 of length 56 as i* for further branching.

Stage 3:

(a) j=j+1=1+1=2, I(2)= ϕ , E(2)={1}, F(2)={2,3,4}. Proceed to (c).

(c) Since there are no included items at node No. 2, the upper bound is found by relaxing the length constraint for the free items.

B(1)=50+10=60.

Since j is even, proceed to 3(b).

(b) j=j+1=2+1=3, I(3)={1}, E(3)= ϕ , F(3)={2,3,4}. Proceed to Stage 2.

A flowchart for the simulation of the cut-off- first method using Kolesar's algorithm is shown in Fig.5.

The results of the algorithm are shown in the solution tree of Fig.6. In the program runs used to evaluate the formulation, six tickets were considered with 250 boards. The average solution time per board was about one minute on an IBM compatible PC with an 8088-1 processor. The solution time mentioned above includes simulated board generation time.

The above formulation cannot be used on a real-time basis since the board processing time on an automated cut-off saw is of the order of a few seconds. Also, current industrial practice is to consider ten tickets for cutting instead of six. Hence, in such a case the solution time would be much longer, since it increases exponentially with the number of nodes. In order to drive the saw using the results derived on a real-time basis, it would be desirable to have a solution that compares favorably with the algorithmic solution and is considerably faster. This was achieved by developing two simple heuristics, explained in the next section, to solve the one-dimensional stock cutting problem.

3.3 Heuristic Solution

In the heuristic solution procedure to be discussed, the solution is derived by a stage-by-stage optimization of the original problem. The problem is successively reduced until there is no feasible solution.The solution procedure proposed is as follows:

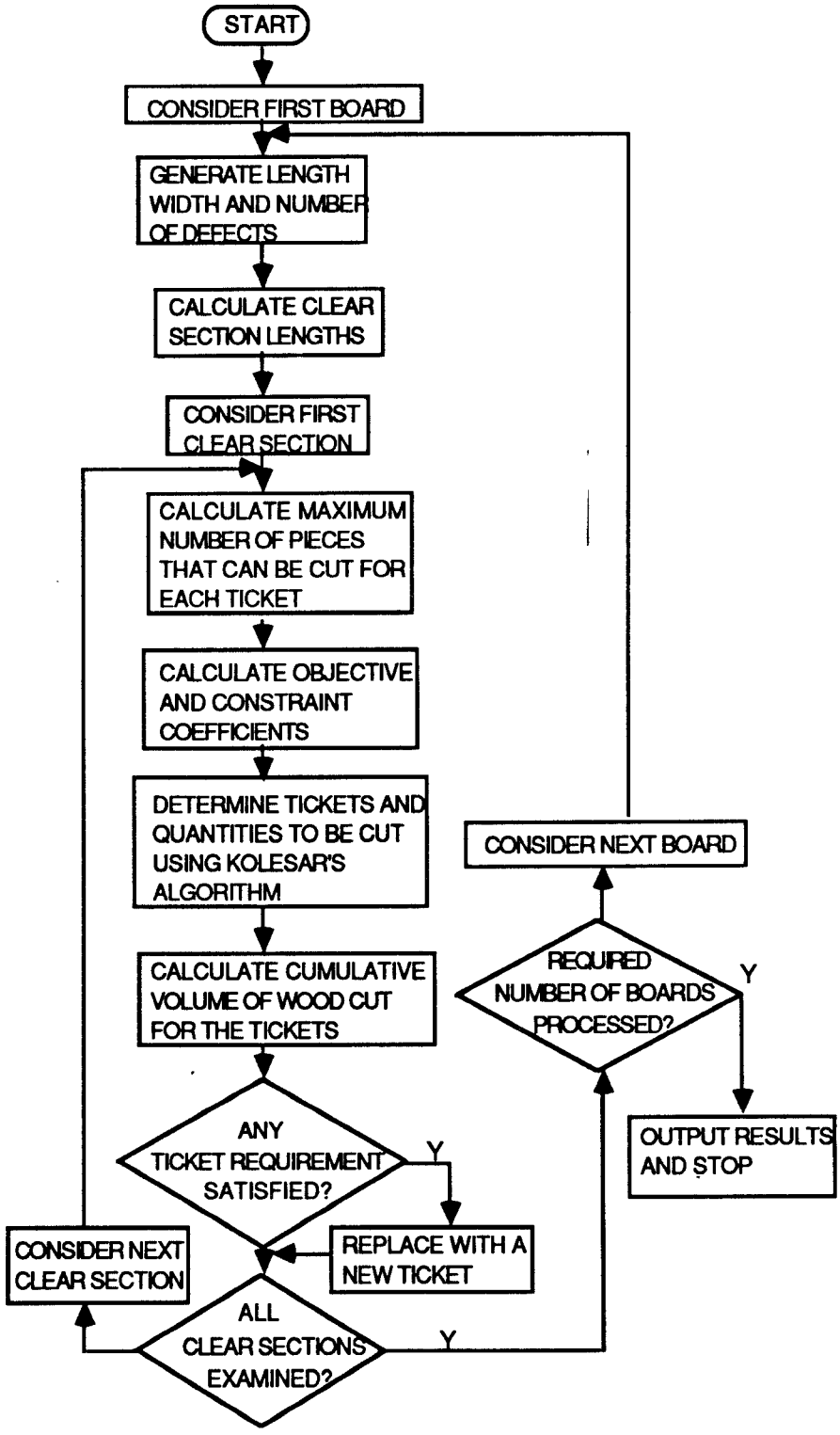


FIG. 5 Algorithmic Solution Procedure Using Kolesar's Method

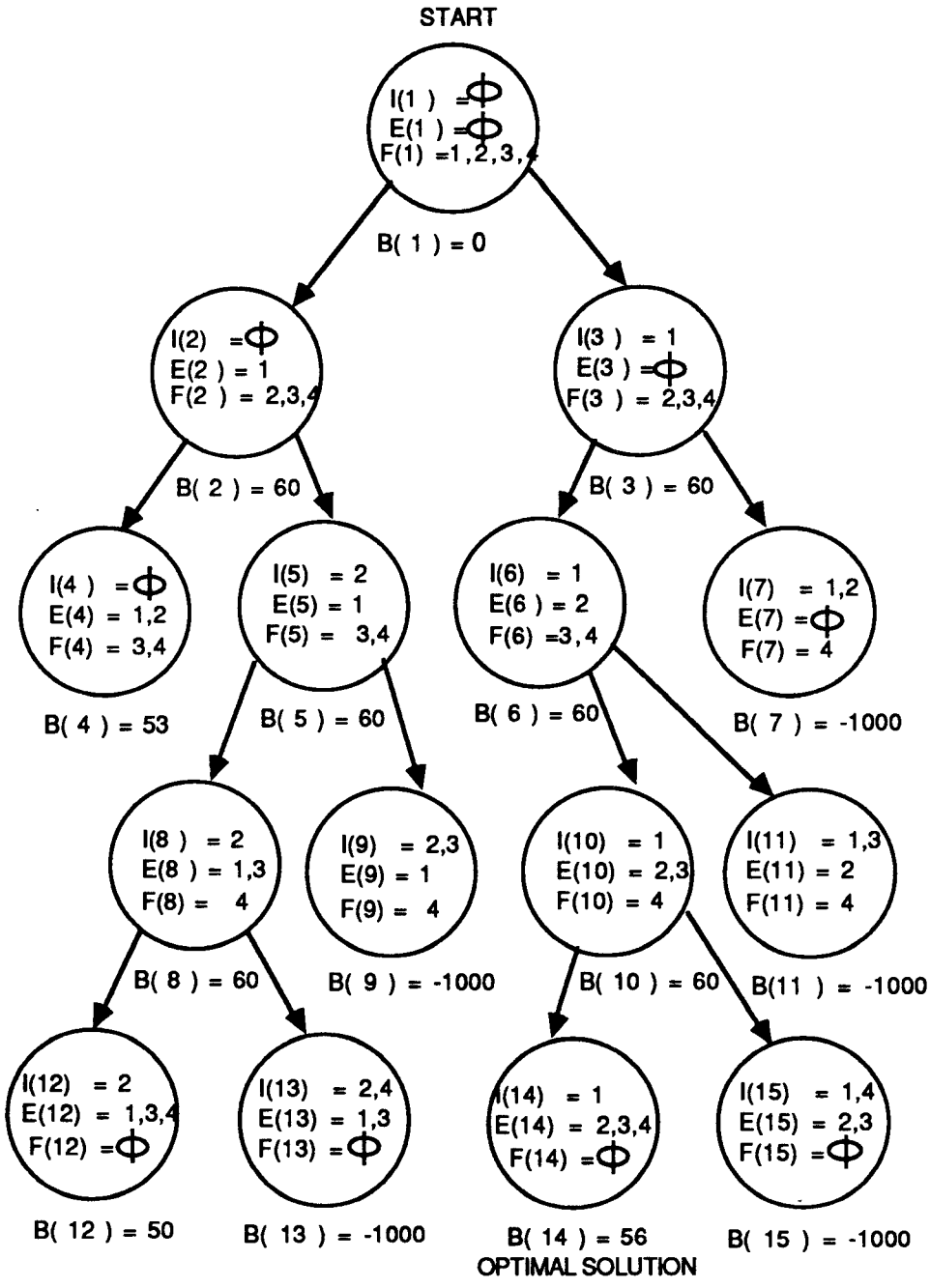


FIG. 6 Solution Tree for the Example Using Kolesar's Algorithm

Step 1: The tickets are considered one at a time, and the maximum number of pieces that can be cut on a given clear section is computed for feasible tickets only. A ticket is infeasible if the section length is shorter than the ticket length.

Step 2: Using the maximum number of pieces, the yield for each ticket when cut on the clear section is computed.

Step 3: The ticket with maximum yield is chosen as the solution. One piece of this ticket is cut on the clear section.

Step 4: The section length is reduced by the length of the ticket cut in step 3. If all the ticket lengths are longer than the revised section length, the solution process is complete. Otherwise, go to step 1.

The flowchart for the heuristic is shown in Fig.7.

As it is observed, using this heuristic the section length is optimized in several stages. Since the ticket that gives best yield is selected at each stage, it can be intuitively seen that the yield cannot decrease from one stage to another. Its main advantage is that as the section length decreases at each stage, the number of feasible tickets that can be cut also decreases. This speeds up the solution process at each stage making this method very fast and efficient.

3.3.1 Need for Ticket Priorities

The demand for clear cuts of different lengths is usually not uniform. In general, longer cuts are preferred over shorter cuts. This, to some extent, depends on the products to be fabricated. The table-top of a large conference-room table, for example, would require longer

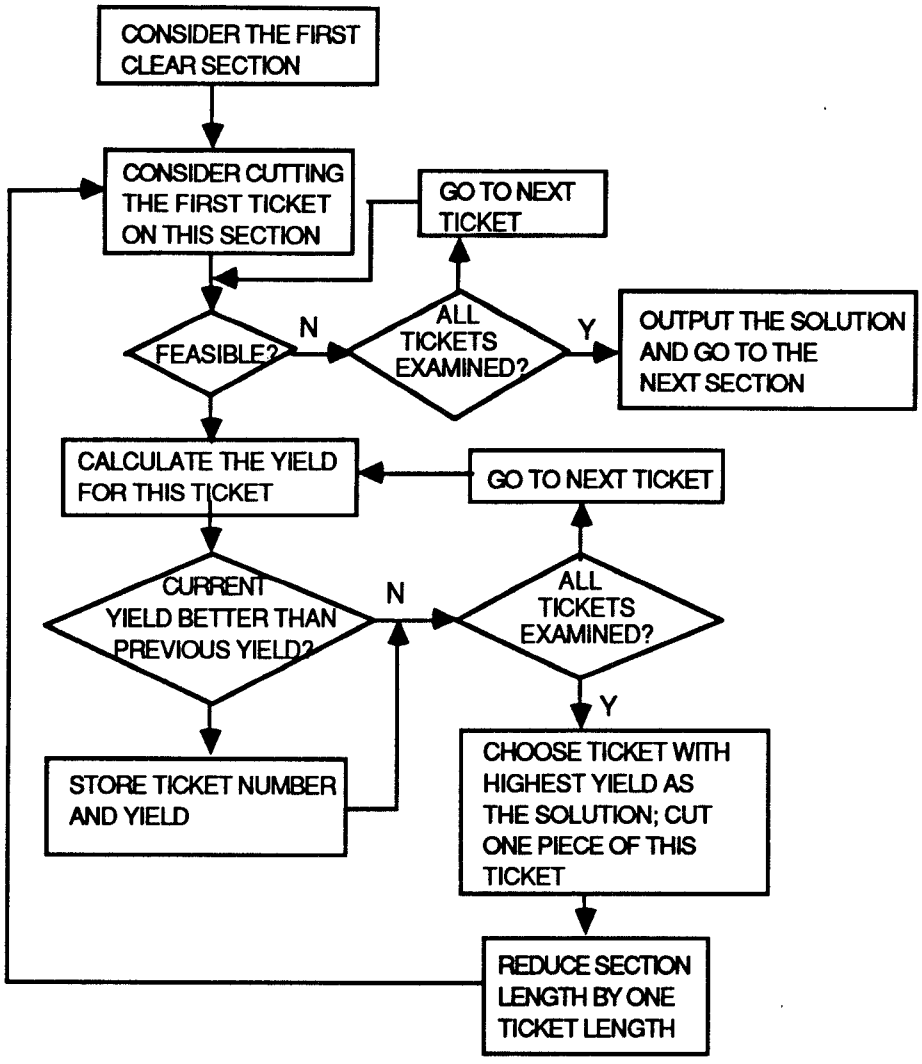


FIG. 7 Solution Procedure Using Heuristic Method

cuts than a small chair. Longer cuts are usually obtained by using better grades which are comparatively costlier.

In addition, the variation of volume of wood cut as a function of ticket length is shown in Fig.8. The output of a simulation run using the heuristic for 1000 boards of No. 1 common grade, with ticket

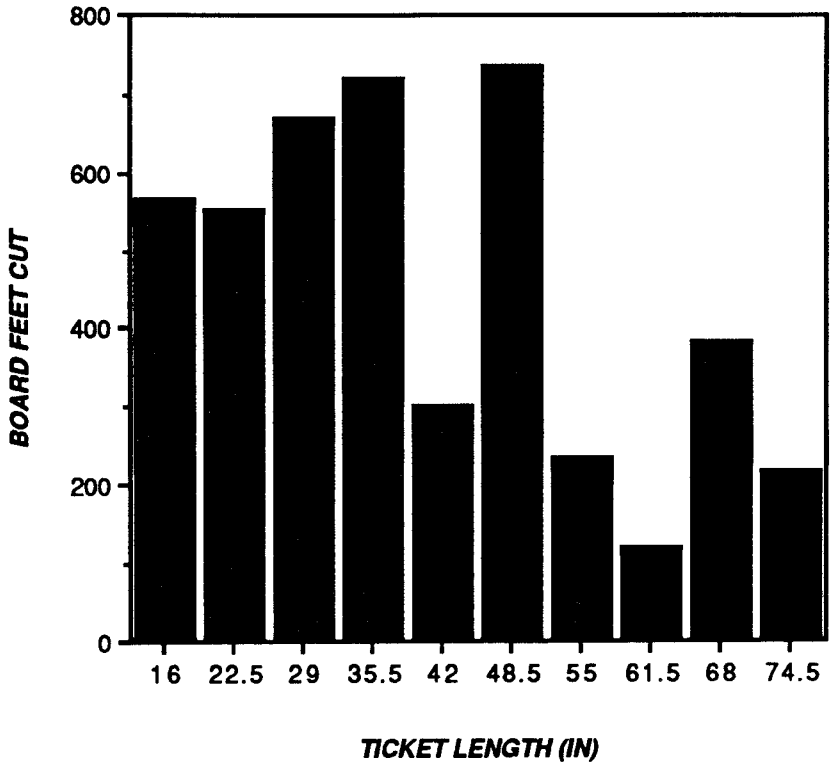


FIG. 8 Volume of Wood Cut as a Function of Ticket Length

lengths ranging from 16" to 74.5", was used to plot this figure. Short tickets can be cut on short as well as long clear sections. In contrast, long tickets can not be cut on sections shorter than the ticket length. As a consequence, short tickets have a better chance of being cut compared to longer tickets. This appears to be the reason for the general trend of decreasing volume cut with increasing ticket length. In general, the cutting process will be biased toward shorter ticket lengths. This is true whenever the difference of length between short and long tickets is considerable, as in Fig.8.

The above discussion brings out the need for including priorities in the solution process. Priorities, to a limited extent, help in controlling the volume of wood cut for different ticket lengths. Hence, it may be desirable to obtain a certain proportion of long cuts from a lower lumber grade at the same time with shorter cuts. One way of achieving this is by using priorities. The proposed method of allocating priorities is discussed below.

First, a scale of priorities was developed. The scale ranges from 0 to 11, in which 0 indicates normal or no priority and 11 indicates topmost priority. Priority 11 tickets are cut on the first available clear section, that is longer than the ticket length, regardless of yield. In using the priorities, the yield of individual tickets is artificially boosted by a factor F. This factor is related to the priority P by the relation,

$$F = e^{CP} \qquad (8)$$

where e is the base of natural logarithms, P = 0,...,10 and C is a constant (arbitrarily, C=0.35 in this study). The boosted yield (i.e., normal yield multiplied by F) is used as the criterion to select the ticket to be cut. Thus, a ticket with a low yield may still be cut if its priority and hence the associated factor F are comparatively large. This method of ticket selection results in a loss of yield. This is because, the ticket with the highest actual yield may not have the highest boosted yield. Thus high priorities are causing increasing levels of yield sacrifices. Here, we associate priority with yield sacrifice because yield is generally used as a performance measure for the rough-mill. Some other measure such as the value of pieces cut could also be used in

allocating priorities. Using this framework, the general solution procedure using priorities is described in the following section.

3.3.2 Heuristic Solution with Priorities

In the solution procedure discussed under section 3.3, the ticket giving the best yield was always selected as the solution in a clear section. If the solution procedure is to include priorities, it follows that the ticket that gives best yield will not, in general, be the ticket with the highest priority. In this case the yield of a given ticket must be artificially modified, using the priority for that ticket. The proposed procedure for including priorities is discussed below:

Step 1: The tickets are considered one at a time and the yield for each feasible ticket is calculated as before.

Step 2: The factor F for each feasible ticket is calculated using equation (8). The actual yield for each ticket is multiplied by the corresponding factor to get the boosted yield.

Step 3: The boosted yields of Step 2 are sorted in order of decreasing magnitude.

Step 4: From the above list the ticket highest in order is selected as the solution. If none of the tickets meets the above criteria, there is no feasible solution.

The flow chart of Fig.9 illustrates the above solution procedure. The results of a run made using ten tickets is shown in Fig.10. In this run, the shortest ticket 16" had a priority of 1 and the longest 80" had a priority of 10. The intermediate tickets had priorities 2 to 10 in the order

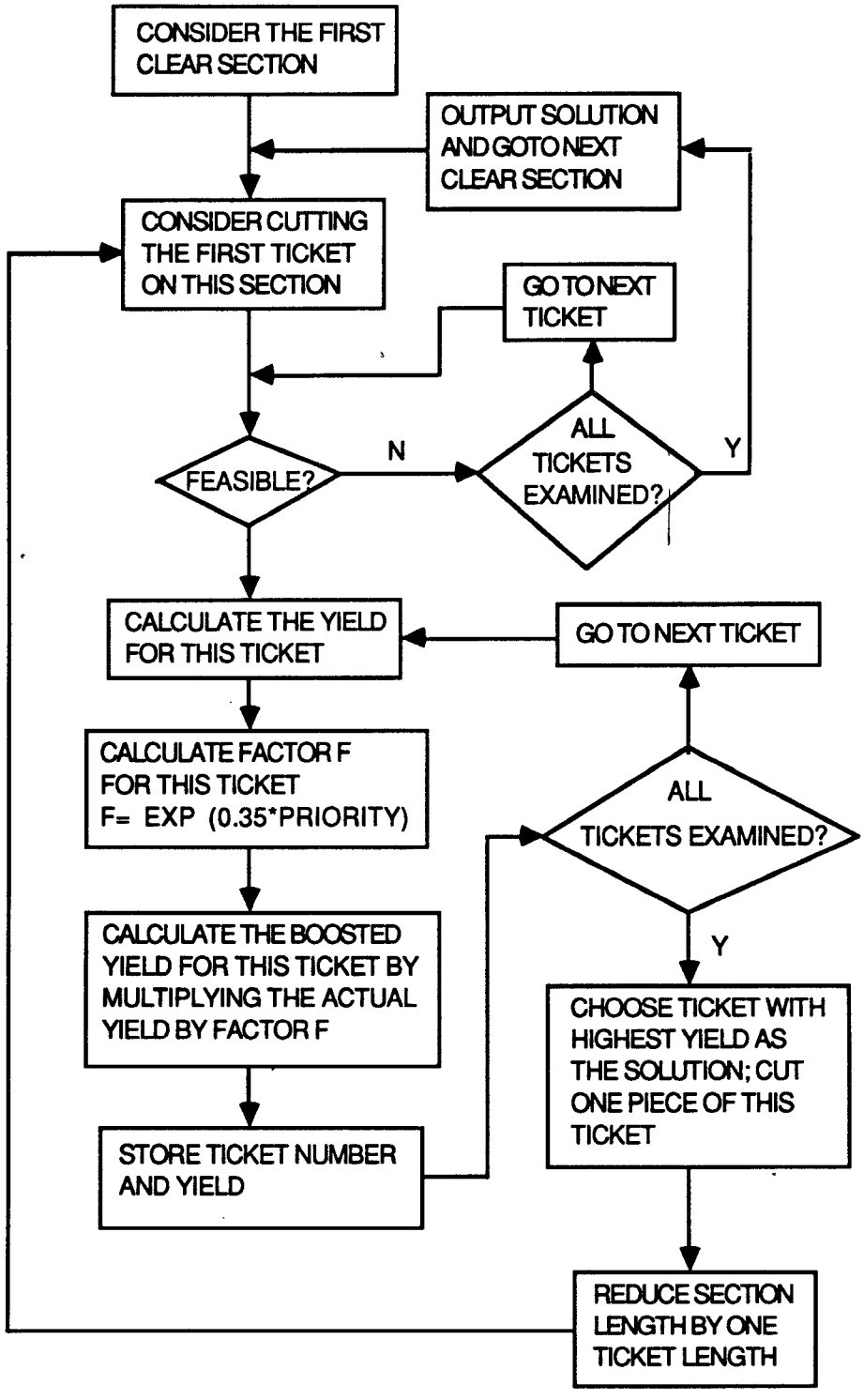


FIG. 9 Flowchart of Heuristic with Priorities

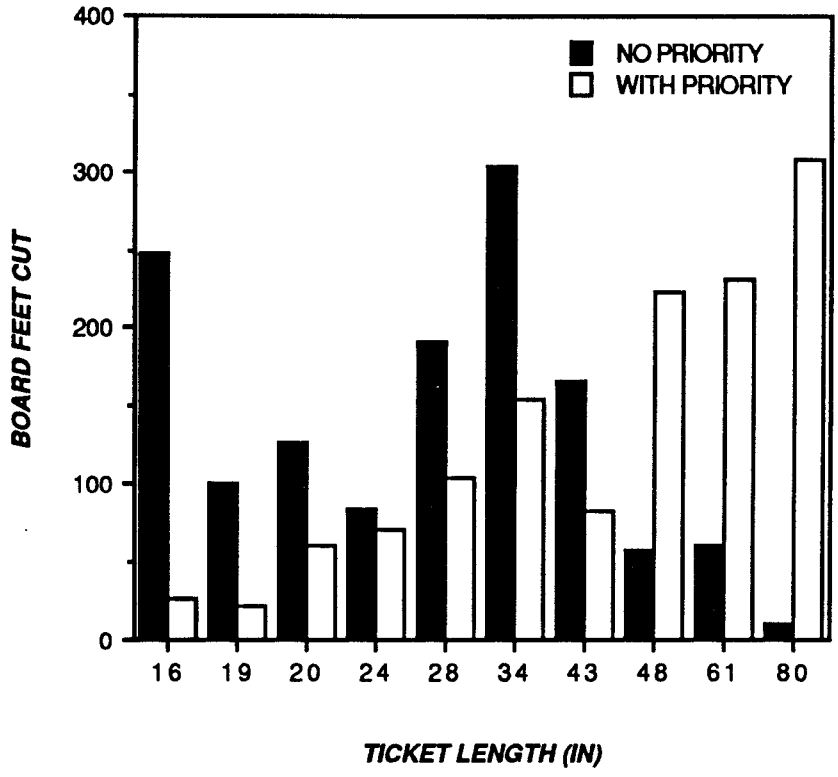


FIG. 10 Effect of Priority on Volume of Wood Cut

of increasing ticket length. A set of 300 simulated boards was used for the two runs. The graph illustrates the significant increase in volume cut for long tickets when they are given high priority.

3.3.3 Dynamic Allocation of Priorities

Assignment of ticket priorities can be either performed by the user or automatically by the algorithm. The former is rather inefficient and time consuming. Therefore, to implement this concept in a practical way the difference between ticket demand and supply

was used as the measure to automatically allocate priorities. The priorities for the tickets were reallocated after solving each clear section. The tickets were ordered in decreasing order of magnitude of the difference between demand and supply. Then the priorities were reallocated, and the highest priority was given to the ticket with the largest difference. This dynamic priority allocation ensures an automatic close match between production and requirement. The output of a simulation run for 400 boards using the above priority allocation is shown in Fig.11. The match between production and requirement is indicated by the match between the slopes of corresponding sections of the two curves. If the simulation were continued for a very large volume of boards and tickets, the slopes of corresponding sections of the two curves would become equal. This means that all tickets will be completed at about the same time. This may be desirable from the viewpoint of obtaining a required mix of ticket lengths on the saw most of the time. However, it is important to note that the match between requirement and production is always achieved at the cost of yield.

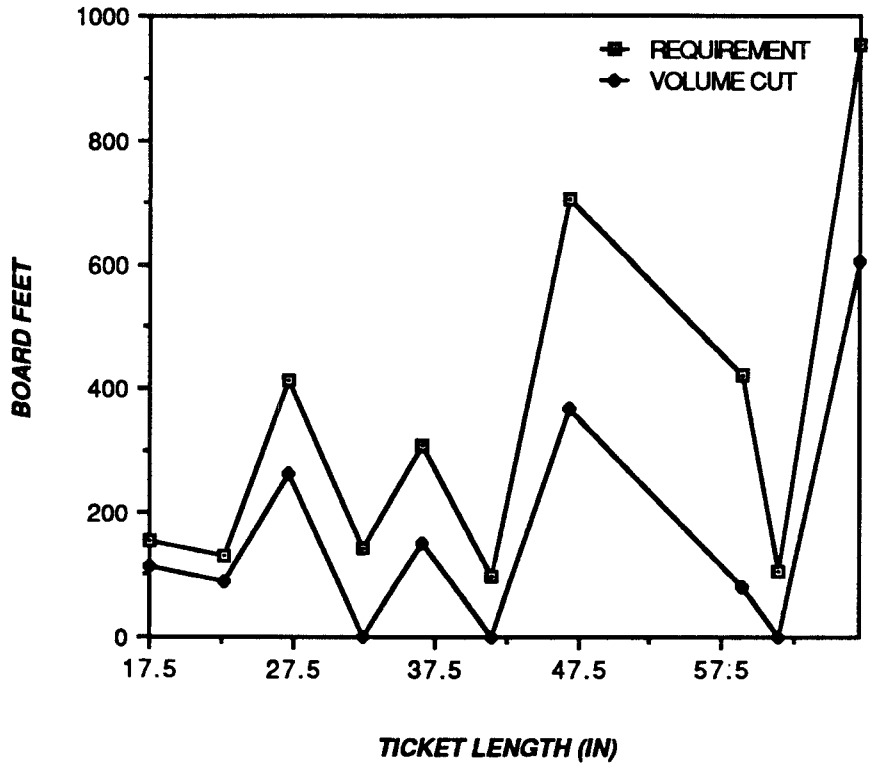


FIG. 11 Controlling Saw Output Using Priorities

CHAPTER 4
TESTS AND RESULTS

One of the primary objectives of this thesis was to develop a practical, yet, time-efficient method of cutting defects out of boards to improve yield. This necessitates the evaluation of the heuristic solution procedure for both, the cut-off and the cut-rip strategy with regard to improvement in yield over the existing manual operation at the cut-off saw. To carry out this evaluation, a series of five experiments were designed and conducted. The objectives of these experiments were to:

(a) validate the solution obtained by using the heuristic procedures on simulated boards with the one obtained by physically cutting a set of boards following the cuts suggested by the heuristic, and

(b) verify that the yield obtained by using the heuristic is better than the yield of the manual operation.

4.1 Experimental Procedure

Out of the five experiments conducted, the first set of four experiments were aimed at validating the heuristic as applied to the cut-off and the cut-rip solution strategies. The aim of the fifth experiment was to determine the improvement in yield over the manual operation. The first set of experiments was conducted using the cut-off and cut-rip strategies. Each strategy was tried for two different lumber grades (No. 1 Common and No. 2 Common).

The fifth experiment was tried using a mixture of grades that was selected to resemble the mix used for normal production at the rough-mill plant. In all these experiments, priorities were not used. This was because of the fact that the operator at the cut-off saw has no capability of using any priorities in the actual operation.

The following steps were involved in carrying out the first set of four experiments:

1. 100 boards of the grade under consideration were selected. Each board was measured for the board length, width, number of defects, the x-y coordinates of the defects with respect to a corner of the board and the size of defects. These are the same parameters used in the Monte Carlo technique of board simulation discussed in Chapter 2. Each measured board was numbered and the defects highlighted for the convenience of the operator.
2. The board and defect data obtained in the previous step were input to the computer program that was written to implement the simulation technique, the heuristic solution procedure, with and without priorities, and the different cutting strategies. This program is discussed in detail in Appendix B.
3. The board data and the program output were used to make drawings of the boards using AUTOCAD [17]. These drawings indicated the tickets to be cut and the locations of the cuts on the board.
4. The operator at the cut-off saw was asked to physically cut the boards according to these drawings.

Steps 1 to 3 were also carried out for experiment 5. But for this experiment, to compare the yields of the operator and that of the program, the operator was not provided with the computer solution. Instead, the operator was asked to cut the boards at his discretion to maximize yield, considering only the highlighted defects. This was done to ensure that the inputs to the program and operator were exactly the same.

4.2 Results

The yields obtained in the case of the first set of four experiments are shown in Table 3. Apparently, the operator was able to obtain all the cuts specified on each of the board drawings. This confirmed the validity of the heuristic solution procedure as applied to the cut-off and the cut-rip methodologies. This explains why the actual and the program yields are equal in Table 3.

The results obtained for the fifth experiment are shown in Table 4. Here, in order to compare the yield of the operator and that of the program, the cut-rip strategy was used. This was done because the operator at the cut-off saw would, while cutting a board, arbitrarily decide whether to cut-off a defect or to leave it uncut for a later ripping operation to remove it. Hence, the pieces cut by the operator at the cut-off saw were subsequently subjected to a ripping operation at the rip-saw. This ensured a one-to-one comparison of the yields in the two cases. It is important to note that the yield achieved by the operator at the cut-off saw can not be directly compared with the yield of the program. This is because, the program uses the cut-rip strategy in

Grade of Wood	Method Of Removing Defects: Cut-Off	Method Of Removing Defects: Cut-Rip
1 Com	Expt # 1 Program: 79.5% Actual : 79.5%	Expt # 3 Program: 81.75% Actual : 81.75%
2 Com	Expt # 2 Program: 72.5% Actual : 72.5%	Expt # 4 Program: 74.1% Actual : 74.1%

Table 3 Results for the First Set of Four Experiments

which the cut pieces are completely clear of defects, and at this time the program decides which specific tickets have to be cut on each clear section. This of course will happen later at the rip saw. In contrast,the pieces cut at the cut-off saw by the operator still have defects in them that are removed at the rip saw and the operator does not plan the tickets to be cut at the cut-off saw. This decision is made by the operator at the rip saw. It can be seen from Table 4 that the yield of the program using the cut-rip strategy was 82.58% against 50.13% for the operator- a difference of 32.45%. But in practice, the pieces produced by the program undergo an edge trim operation to achieve parallel edges, bringing down the program yield by about 10%. Also, in case of the operator, a part of the waste produced at the rip saw is recovered in a

Results:

	OPERATOR		PROGRAM
Ticket Length Ins	Volume Cut at Cut-Off Saw Bd Ft	Volume Cut at Rip Saw Bd Ft	Volume Cut by Program Bd Ft
17.50	24.06	17.21	55.48
22.75	13.98	9.48	27.34
27.25	20.17	11.21	34.59
32.50	24.74	17.60	21.02
36.50	18.31	9.00	47.15
41.38	20.03	10.91	37.71
46.88	18.47	10.74	63.24
58.88	19.10	12.20	6.96
61.50	29.41	19.60	18.96
67.00	128.65	76.82	8.36
Total:	316.92	194.77	320.81
Yield:	**81.58%**	**50.13%**	**82.58%**

Total Volume of Raw Boards= 388.5 Bd ft

Table 4 Results of Fifth Experiment

later salvage operation, increasing the operator yield by about 7%. Thus, the expected yield improvement can be in the range of 15% (72%-57%) even without considering the salvage operation in case of the program. This shows that the implementation of the cut-rip procedure can result in considerable material savings. The low yield in case of the operator is in part due to the fact that the rip-saw operator tries to produce rips of fixed width on a given board, thus wasting considerable material. In contrast, the cut-rip strategy produces rips of variable width, thus providing a better utilization of material. Another important advantage of the computerized solution procedure is that priorities can be included according to demand requirements. This gives the user a better control over the course of the cutting operation.The practical implementation of the cut-rip strategy is discussed in the next chapter.

CHAPTER 5
COMPARISON AND DISCUSSION OF RESULTS

As mentioned in Chapter 3, the technique of simulation and the use of priorities in the heuristic solution procedure were implemented in a computer program. This program, OPTYIELD, is explained in detail in Appendix B. This program was used to make a comparison of the cutting strategies and the effect of parameters such as priorities and grade on the yield of the process. The results of these program runs are discussed below. The implementation of the heuristic as applied to cut-off and cut-rip strategies is also disussed in this chapter.

5.1 Comparison of Heuristic and Algorithmic Solution Procedures

The same 250 simulated boards used in the knapsack formulation were regenerated and applied to the program runs for the evaluation of the heuristic. This was done by utilizing the same set of random numbers in both simulations. The tickets were also identical in the two cases. The yield obtained by using the heuristic compared very favorably with that of the algorithmic solution which gave a maximum yield of approximately 79% for No. 1 common grade. The difference in yield level for the two strategies was about 2%. The results of a run using six tickets and 250 identical boards is presented in Table

5 for the purpose of comparison. A drop of as much as 10% in yield may be expected if dynamic priority allocation is used in the solution process. This is because some of the yield is sacrificed as noted in section 3.3.2.

	BOARD FEET CUT	
TICKET LENGTH	ALGORITHM	HEURISTIC
16.75	277.52	240.32
24.75	265.48	160.11
34.75	234.31	441.63
45.125	149.61	116.66
62.25	76.61	41.98
82.125	19.74	14.75
RAW VOLUME OF WOOD	1406.24	1406.24
TOTAL VOLUME CUT (BOARD FEET)	1023.29	1015.45
YIELD (%)	72.76	72.21
SOLUTION TIME (SEC/BOARD)	55	2

TABLE 5 Comparison of Algorithmic and Heuristic Solutions

The average solution time for the heuristic (without priorities) was approximately 2 seconds per board, including the board generation time, against 55 seconds per board for the algorithmic solution on an IBM-AT compatible with an 8088-1 processor. The solution time, during a practical implementation, will further be reduced since the board generation time is eliminated. In such a case, the board dimensions and the defect locations and sizes would be directly available from an input device such as a scanner instead of being generated by simulation. Such solution times make a real-time application of the heuristic method feasible.

In the case of the algorithmic solution the size of the problem depends on the number of binary variables as mentioned in section 3.2. The solution time for this method is dependent on the number of nodes to be examined. Thus, only problems of moderate size can be solved.

5.2 Effect of Ticket Length on Yield

The simulation and formulation were also used to evaluate the sensitivity of the yield with varying ticket lengths. This is an important issue in woodworking industry, since the requirements for various ticket lengths is determined by the product mix being manufactured. The dependence of yield on ticket lengths cut is shown in Fig.12. To evaluate this variation, ten runs were made with the ticket length held constant for each run. Since ten tickets were used for each run, this was done by confining the ticket lengths to a narrow range. Each of the ten runs shown was made using ten tickets with ticket lengths $\pm$ 3"

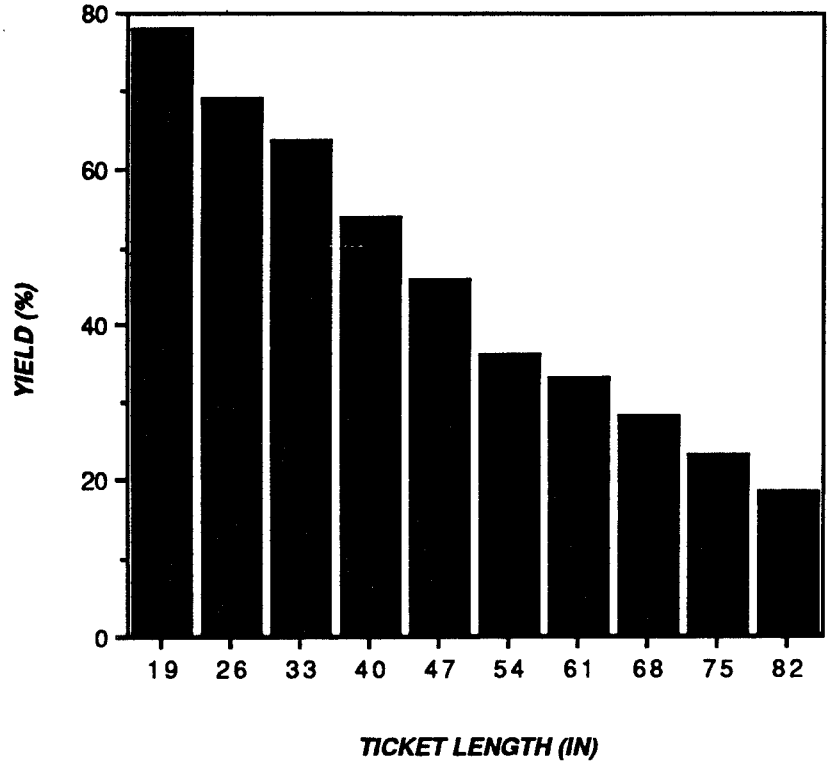


FIG. 12 Variation of Yield with Ticket Length

about a mean value. A set of 250 identical boards was used for each run generated by using an identical set of random numbers. As indicated in the graph, the yield decreases with increasing ticket length. This is a natural consequence of the fact that as the ticket length increases, more clear sections are left unutilized, because of the section length constraint. In fact, a good level of yield for Number 1 Common grade (about 79%) could be obtained only when the tickets used covered the whole range, from the shortest to the longest (16" to 80"). This can be intuitively seen to be due to the fact that a wide range of ticket lengths will yield more number of combinations that meet the

section length constraint. These combinations are also more likely to have a combined length close to the given clear section length, ensuring good yield.

Fig.12 also brings out another important fact of practical significance. The yield obtains its highest value when the shortest tickets are used. In other words, the solutions obtained using the algorithm or the heuristic are biased toward the shortest tickets. This results in more volume of wood being cut for the shortest tickets. Since longer cuts may be occasionally preferred over shorter cuts,this can be a serious disadvantage. As mentioned before, the product mix being manufactured determines the demand for the various ticket lengths. This bias also has an adverse effect on value optimization, since longer cuts usually have a higher monetary value. Hence, inclusion of a priority or a weighting factor for the tickets is mandatory before these algorithms can be implemented in a production environment.

As seen above the product mix has an important bearing on the yield of the cutting process. In order to get a reasonably good yield, it may be desirable to ensure that the product mix results in a demand for a good mix of ticket lengths from the shortest to the longest.

5.3 Comparison of Cut-Off and Cut-Rip Strategies

A comparison of cut-off and cut-rip strategies was made using a series of ten simulation runs. Again, in order to make a one-to-one comparison of yields possible, the same set of boards were used in both strategies. The difference in yield was generally about 7% in favor of the cut-rip strategy (see Table 6). In order to improve the volume of

PROGRAM RUN NUMBER	YIELD (%)	
	CUT-OFF	CUT-RIP
1	72.61	79.17
2	71.60	78.68
3	71.38	78.46
4	70.54	77.94
5	69.80	77.47

Table 6 Comparison of Yields for cut-off and cut-rip Strategies

wood cut for longer tickets, a defect combining technique was used. The effect of combining two close-by defects into one is shown in Fig.13. Arbitrarily, the defects were combined whenever the difference in the x-coordinates of the defects was less than the shortest ticket length and the difference in the y-coordinates was less than 50% of the board width. It is obvious, from the figure, that the clear section lengths are increased by this technique. This fact is of importance, since in most of the woodworking industries longer lengths are required in greater quantities compared to shorter lengths on the average. Thus the cut-rip strategy, in addition to giving higher yield, can also increase the output of longer lengths. This makes it an attractive alternative to the cut-off strategy.

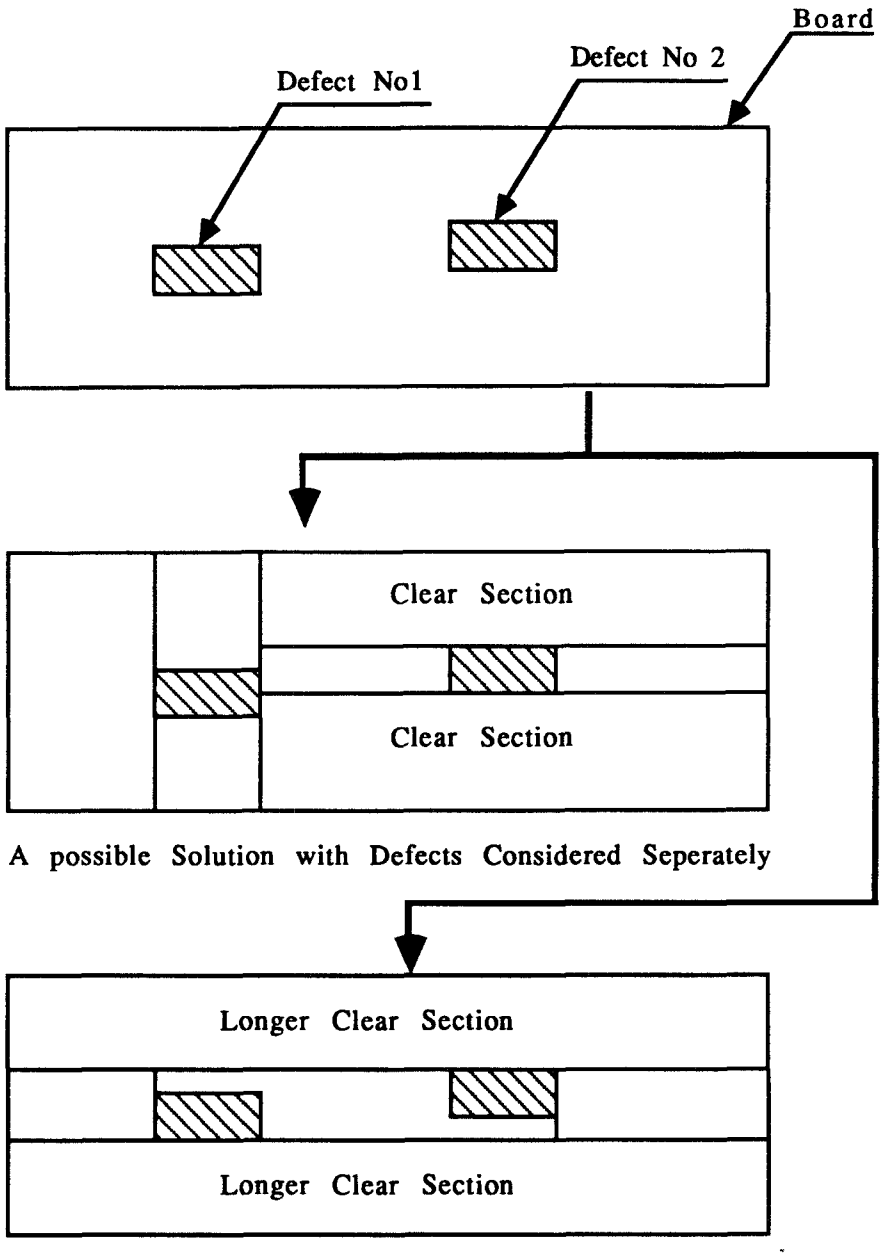


FIG. 13 Effect of Combining Defects in the Cut-Rip Strategy

5.4 Implementation of the Cut-Rip Strategy

The implementation of the cut-rip strategy involves consideration of certain issues. Firstly, several setup changes may be required to cut a board as specified by the cut-rip solution procedure. The pieces cut at the cut-off saw have to be processed on the rip saw. The strips produced at the rip saw may undergo another cut-off operation depending on the solution for a given board. This involves set up changes. Secondly, whenever a defect is left on a clear section to be removed by a later ripping operation, the problem of physically identifying the clear section with its solution arises in later operations. The second issue can be handled by affixing a computer generated illustration of a clear section on a given board, with the ticket(s) to be cut.

The implementation of the program solution for the cut-rip strategy involves four steps:

(i) The program solution for different clear sections of a board is illustrated on stickers that are affixed to the board on the clear sections. The stickers indicate the ticket length to be cut and the number of pieces to be cut on a given clear section.

(ii) The cut-off saw operator cuts the board into proper lengths depending on the position of the stickers.

(iii) These pieces are ripped into proper widths again depending on the sticker positions. The rip width is not critical and the only requirement here is that the strip width should be sufficient to contain the defect within the strip. The strips from the rip saw

will still have the stickers on them indicating the tickets to be cut lengthwise on the strips.

(iv) These strips are brought back and cut off at the cut-off saw to ticket lengths depending on the solution printed on the stickers.

These additional operations involved in the practical implementation of the cut-rip procedure are offset by the advantages of the method: increased yield and more volume cut for longer lengths.

5.5 Hardware Requirements

The practical implementation of these algorithms, typically, involves executing the optimization program on a computer-driven saw or on a Personal Computer external to the saw. A flow chart of the OPTYIELD program, developed during the course of this thesis, is shown in Fig. 14. The program is explained in detail in Appendix B.

A typical set up may consist of a marking station, an optical scanner, a saw unit, a computer control unit and a sorting station. The boards come to the marking station on a conveyor where the defects are marked using a fluorescent marker. The board then passes through a scanner that generates the coordinates of the marks with respect to one end of the board. This data is used by the computer to determine the lengths to be cut on the board to maximize yield. The board then enters the saw unit where the saw, actuated by the control unit, makes the specified cuts on the board. The cut pieces coming out of the saw unit are sorted by length in the sorting unit. A typical implementation procedure for the system described is discussed below.

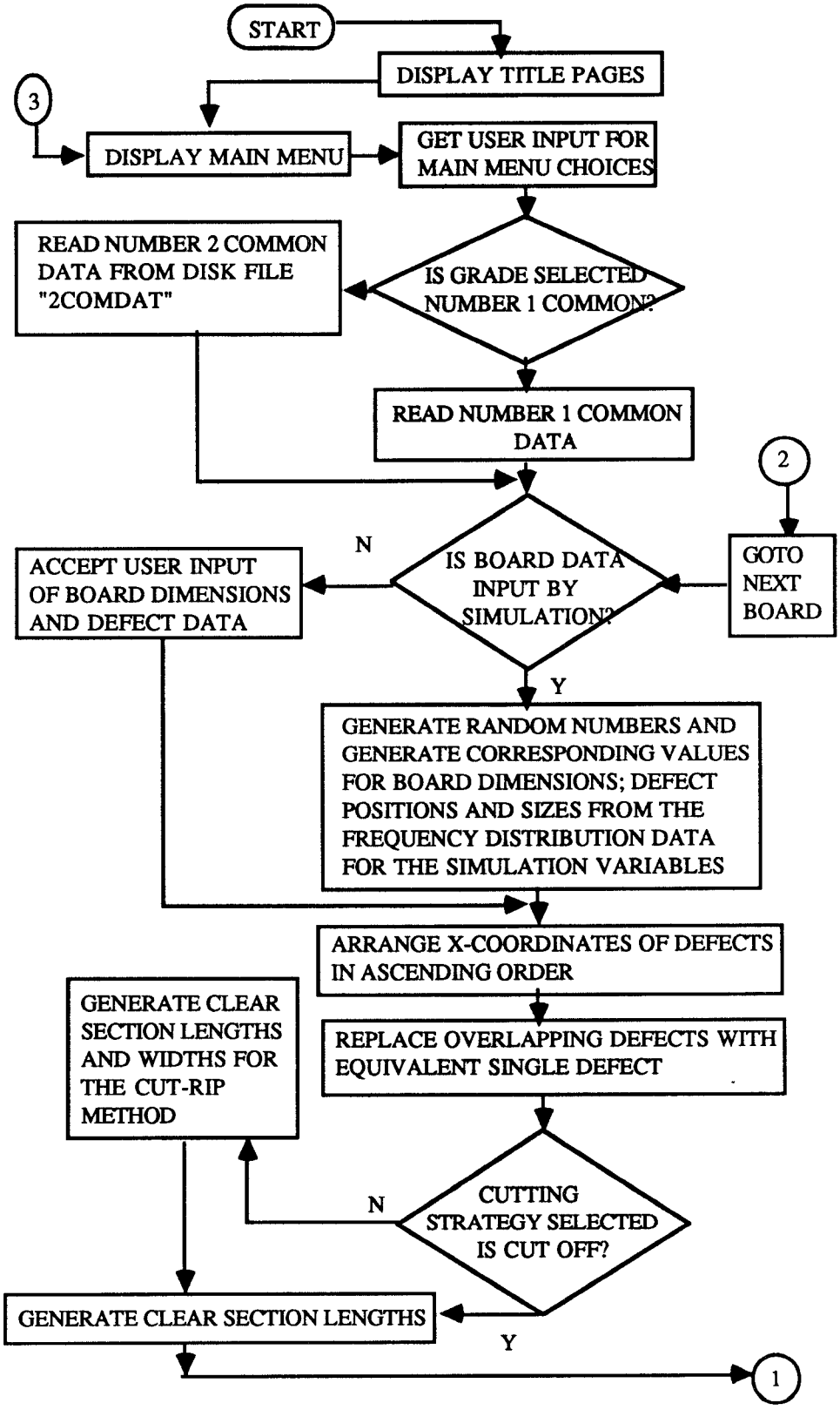


FIG. 14 Flowchart of the OPTYIELD Program

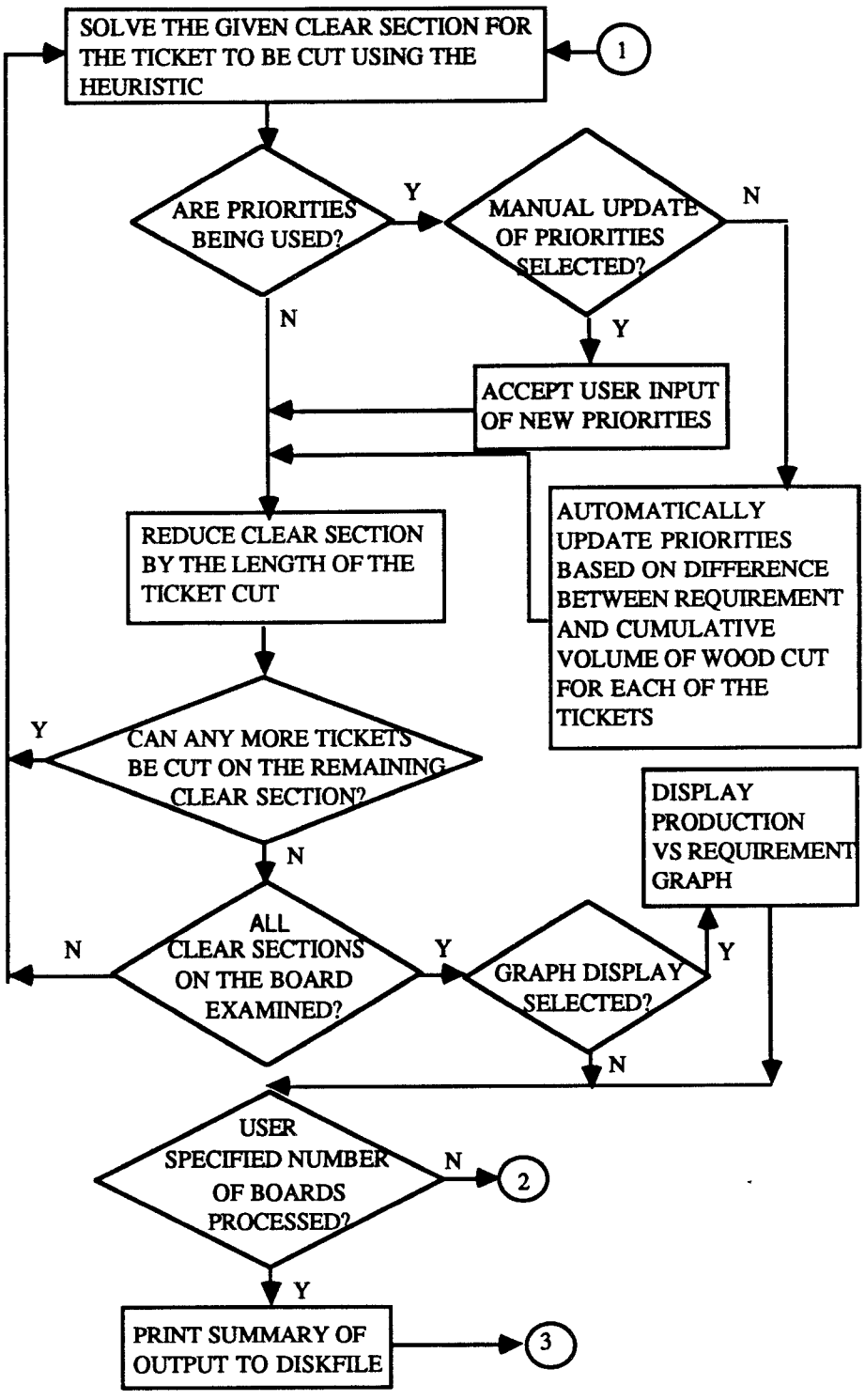


FIG. 14 Flowchart of the OPTYIELD Program (continued)

The input to the program includes board dimensions and x-coordinates of the defects. This input is normally provided by a 'computer vision' system or an equivalent device. Such a device may be in the form of an optical scanner that reads the position of the defects on the board by using the reflection from fluorescent chalk marks on either side of a defect. A quality code sent from the scanner along with the coordinate for a mark, identifies the mark as the beginning of a defective area or a clear section.These coordinates and the board dimensions can be read into the optimization program through a communication port. This data is used by the program to determine the clear section locations and the tickets to be cut. The output of the program is in the form of x- coordinates where the saw cuts are to be made on the board. In the case of automated saws, this output can be used directly to actuate the saw. In the case of manually operated saws, this output can be used to display a drawing of the board on a computer moniter with the position of saw cuts indicated. This drawing can be used by the operator to make the required cuts. An important advantage in case of automating the cutting process is that operator fatigue is not involved and the output of the cutting process will be consistent with regard to yield.

5.6 Some Practical Considerations

The methods discussed so far, even though have many advantages, also have certain limitations. One drawback of the approach presented here is that it does not take into account some of the "hard constraints" encountered in a practical application. These

include constraints that may not be overcome at any cost. As applied to the lumber industry, these may include:

(a) parts of fixed width to be produced by a single rip. This applies to parts that may not be made by gluing-up smaller widths (eg., drawer fronts).

(b) parts that may have defects in them that do not affect the strength of the part. This usually applies to parts that are used in the interior of a assembly and not visible on the outside.

The approach, as presented, does not consider the rips available on a board. Strips of small width, produced during the ripping operation are salvaged by a secondary sawing operation. In some cases, it may be possible to achieve a better yield over the cut-off-first strategy by first ripping out the defect and then using the cut-off operation. This becomes clear when we consider the extreme case when a narrow defect has a length comparable to the board length. In case of rip-first strategy, once the defect is ripped, the proposed approach can be used without any change. In the two strategies above, all the defects are either cut-off or ripped. In contrast, the cut-rip strategy provides an intermediate solution. Here, some of the defects are cut-off and some are ripped. Once the clear sections for a board are identified, the proposed solution procedure remains valid.

CHAPTER 6
CONCLUSIONS

In this thesis the formulation and solution of the cut-off first strategy of removing defects from lumber, as a one-dimensional stock cutting problem, has been discussed. The objective was to develop a real-time computerised solution procedure for the one-dimensional stock cutting problem to maximize yield. This requirement originated from the fact that the solution procedures would be implemented on an automated saw. The following are the conclusions based on the work described in this thesis.

6.1 Conclusion

During the course of this project, a technique to simulate the generation of boards of a specified grade was developed using the Monte Carlo method. This simulation greatly aided the comparison of alternate solution procedures. Furthermore, the use of very expensive raw materials and machine time required for making such comparisons was eliminated.

A fast heuristic with a solution time comparatively less than the physical processing time for a board on an automated saw was developed. A procedure to control the output of the saw, in terms of length requirements, using priorities for ticket lengths was also developed. The heuristic gave yields better than that of the manual

cutting operation. The control of the saw output is an advantage in a production environment with varying demands.

A new cutting strategy called cut-rip was developed to further improve yield over the cut-off-first strategy and increase the volume of wood cut for longer tickets.

A menu driven software OPTYIELD was developed during this project to implement the solution strategies discussed in this thesis on an automated saw. In addition, the program developed can generate and simulate the cutting of boards of a specified grade or grade mix with or without priorities. This greatly simplifies the comparison of solution alternatives and answers "what if" questions.

The cut-off-first strategy of removing defects results in a reasonably good yield in most cases. However, in cases where the defect length is very large, excessive wastages can result. Hence, by considering the rips available on a board it may be possible to obtain better yields. This issue, in part, was addressed by the cut-rip strategy since it considers a limited area of the board and makes a cut-off or rip decision.

Finally, the cut-rip strategy, efficient at producing longer cutting lengths, has a potential for practical applications with the use of the defect combining technique. This strategy can be used with advantages over the cut-off strategy, if a product mix results in greater demands for longer cutting lengths. The overall expected improvement in raw wood yield can be in the range of 12 to 15%, which, in turn, yields monetary savings of about $ 141750 per year, assuming $ 1050000 as annual raw wood cost for a production volume of 1750000 board feet.

6.2 Suggestions for Further Work

The implementation of the work described in this thesis involves capital investment with significant savings in raw material and processing time. In contrast, the manual operation involves wastage in material and increased processing time. The choice of cutting strategy, however, will have to be based on an economic analysis of process alternatives.

Developing an efficient heuristic for this one-dimensional problem is a first step toward solving the two-dimensional stock cutting problem on a real-time basis.These are important aspects to be considered in future work in this area.

REFERENCES

[1] Groneman, C.H., "General Woodworking," McGraw-Hill, 1976, pp. 2-9.

[2] Wolansky, W.D., "Woodworking Made Easy," Charles Scribner's Sons, New York, 1972, pp. 1-9.

[3] "Wood Handbook 72," Forest Products Laboratory, U.S Department of Agriculture, Washington D.C.

[4] Gilmore, P.C., and Gomory, R.E.,"A Linear Programming Approach to the Cutting Stock Problem," *Opns.Res.*, Vol. 11, 1963, pp. 863-887.

[5] Hahn, S.G.,"On the Optimal Cutting of Defective Sheets," I.B.M. New York Scientific Center Report No. 320-2916, 1967.

[6] Sarin, S.C.,"Two-dimensional Stock Cutting Problems and Solution Methodologies," *Trans. of ASME., Journal of Engineering for Industry.*, Vol. 105, No. 3, Aug. 83, pp. 155-160.

[7] Kolesar, P.J.,"A Branch and Bound Algorithm for the Knapsack Problem," *Manage.Sci.*, vol. 13, 1967, pp. 723- 735.

[8] Christofides, N.,and Whitlock, C.,"An Algorithm for Two-dimensional Cutting Problem," *Opns.Res.*, Vol. 25, No. 1, 1977, pp. 30-44.

[9] Wodzinski, C., and Hahm, E.,"A Computer Program to Determine Yields of Lumber," USDA Forest Products Laboratory unnumbered Research Paper, 1966.

[10] Giese, P.J., and McDonald, K.A.,"OPTYLD - A Multiple Rip-First Computer Program to Maximize Cutting Yields," Forest Products Laboratory Research Paper FPL 412, 1982.

[11] Giese, P.J., and Danielson, J.D.,"CROMAX - A Crosscut-First Computer Simulation Program to Determine Cutting Yield," Forest Products Laboratory General Technical Report FPL-38, 1983.

[12] Hallock, H., and Giese, P.,"Does Gang Ripping Hold the Potential for Higher Clear Cutting Yields," Forest Products Laboratory Research Paper FPL 369, 1980.

[13] McDonald, K.A., Giease, P.J., and Woodfin, R.O.,"Maximum Cutting Yields for 6/4 Ponderosa Pine Shop Lumber," Forest Products Laboratory Research Paper FPL 437, 1983.

[14] Hallock, H., and Giese, P.,"Cutting Yields from Standard Hardwood Lumber Grades when Gang Ripping," USDA Forest Service Research Paper FPL 370.

[15] Schroeder, R., "Operations Management," Second Edition, McGraw Hill Book Company, New York, 1985, pp. 24-25.

[16] Kreyszig, E., Chapter. 18,"Advanced Engineering Mathematics," Second Edition, John Wiley and Sons, New York, 1967, pp. 817-819.

[17] Berghauser, T.W., and Schlieve, P.L., "The Illustrated AutoCAD Book," Wordware Publishing, Inc., Texas, 1987.

[18] "Selected ASTM Engineering Materials Standards, American Society for Testing Materials, Philadelphia, Pa, 1956, pp. 341-343.

APPENDIX A

Glossary of Commonly Used Terms in Woodworking

Checks: A separation along the grain, the greater part of which occurs across the rings of annular growth.

Grade: The designation of the quality of wood.

Grain: The direction, size, arrangement, appearance or quality of the fibers in wood.

Hardwoods: Generally, one of the botanical groups of trees that have broad leaves in contrast to the conifers. The term has no reference to the actual hardness of wood.

Knot: That portion of a branch which has become incorporated in the body of a tree.

Lumber: Sawed wood.

Rip: To saw or split lumber with the grain.

Rough mill: The area of fabrication that involves cutting boards into rough size lengths and gluing-up if necessary. Finishing the parts to proper dimensions and contouring is done in a different area.

Softwoods: Generally, one of the botanical groups of trees that in most cases have needle or scalelike leaves; the conifers; also the wood produced by such trees. The term has no reference to the actual hardness of wood.

Ticket: A shop-floor document that specifies the rough, finished dimensions of a part and the quantity required. It is also accompanied by a set of shop orders required for the fabrication of the part at the various work centers.

APPENDIX B

OPTYIELD Program : User's Guide

This section describes briefly the yield optimization program OPT-YIELD developed during the course of this project work. This program is mainly intended to demonstrate the simulation technique, the heuristic solution procedure, and the use of priorities discussed in this thesis. All the three defect removal strategies discussed before have been implemented in this program. The program is written in BASICA and can be run on IBM PC XT/AT or compatibles. The program is menu driven. The user is provided with an explanation of the menu choices available at the time of start-up. When the menu choices are made, the user is prompted for the required input. A complete listing of the program may be found in Appendix C.

List of Principal Variables Used in the Program

Variable	Stands for	Value	Remarks
A	Defect length	---	Array
AAO	Total raw volume of wood processed	---	---
A1	Raw volume of a board	---	---
B	Defect width	---	Array
BF	Board feet required	---	Array
BFC	Board feet cut	---	Array
BFSEC	Board feet cut on a section	---	---
BFT	Cumulative volume of wood cut	---	---
BFT1	Cumulative volume of wood cut for GR=1	---	---

Variable	Stands for	Value	Remarks
BFT2	Cumulative volume of wood cut for GR=2	---	---
CUTWA	Waste if a defect were cut-off	---	---
DEC	Decision code for a defect in the cut-rip strategy	0	Defect is cut-off
		1	Defect is ripped
F1	Value of a class interval for variable L	---	Array
F2	Value of a class interval for variable W	---	Array
F3	Value of a class interval for variable N	---	Array
F4	Value of a class interval for variable X	---	Array
F5	Value of a class interval for variable Y	---	Array
F6	Value of a class interval for variable A	---	Array
F7	Value of a class interval for variable B	---	Array
GR	Grade being cut	1	No. 1 common
		2	No. 2 common
J	Board number	---	---
L	Length of a clear section	---	Array
LO	Overall board length	---	---
M	Number of defects	---	---
NSEC	Number of clear sections	---	---
Q$	Variable to store menu choice made	---	String

Variable	Stands for	Value	Remarks variable
PR	Priority value for a ticket	---	Array
RA	Rip allowance on width	---	%
RET	Replacement ticket number	---	---
RIPWA	Waste if a defect were ripped	---	---
STAM	Variable to keep track of menu choices made by user	---	Array
SW	Clear section width	---	Array
TL	Ticket length	---	Array
TOL	Total length of tickets cut on a clear section	---	---
V1 to V7	Frequencies in a class interval corresponding to values F1 to F7	---	Array
WD	Decreased board width after applying rip allowance	---	---
WO	Overall board width	---	---
X	X-coordinate of defect	---	Array
Y	Y-coordinate of defect	---	Array
YB	Yield for a board	---	%
YC	Cumulative yield	---	%

User Input

The input to the program consists of :

(a) the ticket data in the form of ticket length, width, and number of pieces / total volume required for each ticket

(b) priorities to be used for each ticket (if user-defined)

(c) grade(s) to be used

(d) cutting strategy to be used i.e; cut-off or cut-rip

(e) board and defect dimensions either from actual boards or from simulated boards.

Program Execution

The program execution is started by choosing the RUN option from the main menu. Error messages are displayed if all data required for a run are not available. The input data are retained after a run is made so that the program may be rerun after making modifications to the input data.

Program Output

The output of the program program is mainly in the form of clear section dimensions, the tickets cut on each clear section, volume cut for each ticket, yields - individual board and cumulative. A summary of the output for a run may be displayed by selecting from the DISPLAY SUMMARY OF OUTPUT option. The output of a run is also written to a disk file for later reference. However, this file is overwritten every time a new run is made.

An Example to Illustrate the Operation of the Program

The operation of the program is briefly illustrated below with a simple example. A flowchart of the program is shown in Fig. 13. The following data are assumed as user input to the program:

Kerf : 0.2 inches

Ticket Data:

Ticket number 1

Length : 16.5 inches

Width : 24.0 inches

Requirement : 100 Board Feet

Ticket number 2

Length : 45.0 inches

Width : 12.0 inches

Requirement : 200 Board Feet

The following are assumed as options selected by the user for the main menu choices of the program:

Priority option selected	: Automatic update of priorities
Grade option selected	: Single grade; Number 1 common
Cutting strategy selected	: Cut-off
Board data input	: By simulation with end trim = 1 inch; Edge rip allowance = 3% of board width; Number of boards to be cut = 100
Production vs requirement graph option	: Not selected

The sequence of activities performed by the program, including the user input of options for menu choices, is listed below with corresponding program line numbers:

Line Numbers

From	To	Activity performed
9100	9615	Display a brief explanation of menu choices available to the user
5110	6380	Display main menu and accept user input of menu choices

Line Numbers		
From	To	Activity performed
6390	6540	Check availability of all data required to execute the program; print error messages if necessary for corrective action by the user
120	620	Read frequency distribution data for the simulation variables of the selected grade from the DATA statements in line numbers 1780 to 1940
650	-	Initialize variables
670	710	Generate a value for the simulated board length using a random number generated in subroutine 2160
720	760	Generate a value for simulated board width
770	-	Calculate volume of the board generated and cumulative volume of wood processed
790	830	Generate a value for simulated number of defects
1200	1230	Arrange defect x-coordinates in ascending order and replace overlapping defects with a single equivalent defect to reduce solution time. Calculate clear section lengths on the given board
1470	1585	Solve the clear section lengths one at a time for tickets and number of pieces to be cut using heuristic in subprogram 7030; print solution to screen
7390	7490	Replace finished tickets if any
4000	5100	Update ticket priorities based on difference between requirement and cumulative volume cut for each ticket
1590	1610	Calculate individual board and cumulative yields
1620	1770	Display yield for board processed and return control to line number 650 until user specified

Line Numbers		
From	To	Activity performed
		number of boards have been processed
1950	2110	Summary of program output written to diskfile
60	-	Control returned to main menu at line number 5110

To illustrate the solution procedure for a clear section, a clear section length of 100 inches is assumed.

Stage 1

Clear section length = 100.00 inches

Ticket Length (with kerf)	Maximum number of pieces that can be cut	Yield(%)	Modified yield based on priority
16.70	5	82.5	166.13
45.20	2	90.0	1042.95

Since 1042.95 is greater than 166.13, one piece of ticket length 45.2 is cut on the given section. Section length is reduced to 100-45.2 = 54.8 inches. The difference between requirement and volume cut is 100 for ticket 1 and 196.25 for ticket 2. The new priorities are 9 for ticket 1 and 10 for ticket 2 since the scale of priorities is from 1 to 10 and there are only two tickets.

Stage 2

Clear section length = 54.8 inches

Ticket Length (with kerf)	Maximum number of pieces that can be cut	Yield(%)	Modified yield based on priority

16.70	3	90.32	2107.7
45.20	1	82.11	2719.1

Again one piece of length 45.2 is cut on the given clear section. Remaining section length is 54.8 - 45.2 = 9.6 inches. Since this is less than both ticket lengths, no more tickets can be cut. Thus the final solution for the original clear section of 100 inches length is to cut two pieces of ticket 2.

The above sequence of activities is repeated until the user specified number of boards have been processed. Then the control is returned to the main menu. The user can then display the summary of the program run by selecting the "display summary of output" option.

APPENDIX C
Listing of OPTYIELD Program

A complete listing of the program OPT-YIELD is provided below for the user's reference. The listing of the main program is followed by a listing of the program "2comdat.bas". This file is required to be on the A drive if Number 2 common grade is selected for any run.

Listing of main program OPTYLD.BAS

```
5 ON ERROR GOTO 9700
10 REM MENU DRIVEN OPTIMIZATION PROGRAM
15 KEY OFF:CLS:GOSUB 9100
20 CLEAR ,,2000:DEFINT I,J,H,K:COLOR 14,9:CLS:DIM PR(30)
40 RANDOMIZE TIMER:DIM STAM(18)
50 OPEN "O",#1,"c:output"
60 GOSUB 5110
70 IF STAM(13)=1 THEN DIM
I(1),F1(1),F2(1),F3(1),F4(1),F5(1),F6(1),F7(1),V1(1),V2(1),V3(1
),V4(1),V5(1),V6(1),V7(1):GOTO 630
80 IF STAM(6)=1 AND STAM(7)=2 THEN CHAIN "A:2comdat",9000,ALL
90 IF STAM(6)=2 THEN GOSUB 6940
100 IF STAM(6)=2 AND GR=2 THEN CHAIN "A:2COMDAT",6000,ALL
110 IF RIND>0 THEN RIND=0:ERASE
I,F1,F2,F3,F4,F5,F6,F7,V1,V2,V3,V4,V5,V6,V7
120 DIM I(7),F1(20),F2(27),F3(10),F4(37),F6(30)
130 DIM V1(20),V2(27),V3(10),V4(37),V6(30)
140 DIM F5(38),F7(12),V5(38),V7(12)
150 FOR F=11 TO 30
160 BFC(F)=0:BF(F)=0
170 NEXT F
180 FOR C= 1 TO 7
190 READ I(C)
200 NEXT C
210 FOR C= 1 TO 20
220 READ F1(C)
230 NEXT C
240 FOR C= 1 TO 20
250 READ V1(C)
260 NEXT C
270 FOR C= 1 TO 27
280 READ F2(C)
290 NEXT C
300 FOR C= 1 TO 27
310 READ V2(C)
320 NEXT C
330 FOR C= 1 TO 10
340 READ F3(C)
350 NEXT C
360 FOR C= 1 TO 10
370 READ V3(C)
380 NEXT C
390 FOR C= 1 TO 37
400 READ F4(C)
410 NEXT C
```

```
420 FOR C= 1 TO 37
430 READ V4(C)
440 NEXT C
450 FOR C=1 TO 38
460 READ F5(C)
470 NEXT C
480 FOR C=1 TO 38
490 READ V5(C)
500 NEXT C
510 FOR C= 1 TO 30
520 READ F6(C)
530 NEXT C
540 FOR C= 1 TO 30
550 READ V6(C)
560 NEXT C
570 FOR C=1 TO 12
580 READ F7(C)
590 NEXT C
600 FOR C=1 TO 12
610 READ V7(C)
620 NEXT C
630 DEFINT I,J,H,K:AAO=0
640 ST$=TIME$
650 K=0:J=J+1:BFSEC=0:NSEC=0:YB=0:A1=0
660 IF STAM(13)=1 THEN GOSUB 2840:GOTO 1170
670 Z=1:GOSUB 2160
680 K=K+1
690 IF N<=V1(K) THEN GOTO 710
700 GOTO 680
710 LO=F1(K)
720 K=0:Z=2:GOSUB 2160
730 K=K+1
740 IF N<=V2(K) THEN GOTO 760
750 GOTO 730
760 WO=F2(K)
770 A1=LO*WO/144:AAO=AAO+A1
780 WD=WO-(WO*RA)
790 K=0:Z=3:GOSUB 2160
800 K=K+1
810 IF N<=V3(K) THEN GOTO 830
820 GOTO 800
830 M=F3(K)
840 IF STAM(11)=2 THEN GOTO 860
850 DIM X(M),A(M),L(M+1):GOTO 870
860 DIM X(M),A(M),Y(M),B(M),L(20),SW(20),DEC(20)
870 IF M=0 AND STAM(11)=1 THEN L(1)=LO-(2*E):NSEC=1:GOTO 1360
880 IF M=0 AND STAM(11)=2 THEN L(1)=LO-
(2*E):NSEC=1:SW(1)=WD:GOTO 1360
890 IF M=1 THEN D=1:GOTO 910
900 FOR D= 1 TO M
910 K=0:Z=4:GOSUB 2160
920 K=K+1
930 IF N<=V4(K) THEN GOTO 950
940 GOTO 920
950 X(D)=(LO*F4(K)/100)
960 IF STAM(11)<>2 THEN GOTO 1020
970 K=0:Z=5:GOSUB 2160
```

```
980 K=K+1
990 IF N<=V5(K) THEN GOTO 1010
1000 GOTO 980
1010 Y(D)=(WO*F5(K)/100)-(WO*RA/2)
1020 K=0:Z=6:GOSUB 2160
1030 K=K+1
1040 IF N<=V6(K) THEN GOTO 1060
1050 GOTO 1030
1060 A(D)=F6(K)
1070 IF STAM(11)<>2 THEN GOTO 1140
1080 K=0:Z=7:GOSUB 2160
1090 K=K+1
1100 IF N<=V7(K) THEN GOTO 1120
1110 GOTO 1090
1120 B(D)=F7(K)
1130 GOSUB 2430
1140 IF M=1 AND STAM(11)=1 THEN GOTO 1270
1150 IF M=1 AND STAM(11)=2 THEN GOSUB 3510:GOTO 1280
1160 NEXT D
1170 IF M=2 AND STAM(11)=1 THEN GOTO 1250
1180 IF M=2 AND STAM(11)=2 THEN GOTO 1240
1190 IF STAM(11)=2 THEN GOTO 2340
1200 GOSUB 2240
1210 GOSUB 2640
1220 GOSUB 2500
1230 GOTO 1280
1240 IF STAM(11)=2 THEN GOTO 2390
1250 IF X(1)>X(2) THEN
R=X(1):X(1)=X(2):X(2)=R:AS=A(1):A(1)=A(2):A(2)=AS
1260 GOSUB 2640
1270 GOSUB 2500
1280 IF NSEC=0 THEN J=J-1:AAO=AAO-A1:GOTO 1760
1290 CY=0
1300 FOR Y=1 TO NSEC
1310 CY=CY+(WD*L(Y)/144)
1320 NEXT Y
1330 CY=CY/A1
1340 IF CY<.66 THEN J=J-1:AAO=AAO-A1:GOTO 1760
1350 IF J>PJ THEN GOTO 1470
1360 PRINT#1," ":PRINT#1,"SOLUTION FOR BOARD  ";J
1370 CLS:PRINT"SOLUTION FOR BOARD  ";J
1380 PRINT#1,"BOARD LENGTH: ";:PRINT#1,USING "###.##";LO;:PRINT
#1," INS":PRINT "BOARD LENGTH: ";:PRINT USING
"###.##";LO;:PRINT " Ins"
1390 PRINT #1,"BOARD WIDTH: ";:PRINT#1,USING "##.##";WO;:PRINT
#1," INS":PRINT "BOARD WIDTH: ";:PRINT USING "##.##";WO;:PRINT
" Ins"
1395 IF STAM(11)=1 THEN GOTO 1460
1400 PRINT#1,"REV NO OF DEFS:";M
1410 FOR D=1 TO M
1420 PRINT#1,"X";D;" =";X(D);"  Y";D;" =";Y(D)+(WO*RA/2);"
A";D;" =";A(D);"  B";D;" =";B(D)
1430 IF DEC(D)=0 THEN PRINT#1,"Defect ";D;" cutoff":GOTO 1450
1440 PRINT#1,"Defect ";D;" Ripped"
1450 NEXT D
1460 PRINT#1,"NUMBER OF SECTIONS: ";NSEC:PRINT #1,"
":PRINT"NUMBER OF SECTIONS: ";NSEC
```

```
1470 FOR P=1 TO NSEC
1475 DIM QTY(10)
1480 IF J<=PJ AND STAM(11)=1 THEN PRINT#1,"SECTION NUMBER:
";P;"  SECTION LENGTH: ";:PRINT#1, USING "###.##";L(P):PRINT
#1," ":PRINT:PRINT"SECTION NUMBER: ";P;"  SECTION LENGTH:
";:PRINT USING "###.##";L(P):PRINT
1490 IF J<=PJ AND STAM(11)=2 AND SW(P)<>0 THEN PRINT#1,"SECTION
NUMBER: ";P;"  SECTION LENGTH: ";:PRINT#1, USING
"###.##";L(P);SW(P):PRINT #1," ":PRINT:PRINT"SECTION NUMBER:
";P;"  SECTION SIZE: ";:PRINT USING "###.##";L(P);SW(P):PRINT
1500 FOR C=1 TO 10
1510 IF TL(C)>L(P) THEN GOTO 1530
1520 GOTO 1550
1530 NEXT C
1540 GOTO 1580
1550 IF STAM(4)=2 THEN GOSUB 4920
1560 IF STAM(11)=2 THEN GOSUB 4000:GOTO 1580
1570 GOSUB 7030
1580 ERASE QTY
1585 NEXT P
1590 BFT=BFT+BFSEC
1600 YB=BFSEC/A1
1610 YC=BFT/AAO
1620 PRINT"BOARD NUMBER ";J;:PRINT "     YIELD= ";:PRINT USING
"##.##";(YB*100);:PRINT "%";:PRINT "     CUM YIELD= ";:PRINT
USING "##.##";(YC*100);:PRINT "%"
1630 IF J<=PJ THEN PRINT#1,"BOARD NUMBER ";J;:PRINT #1,"
YIELD= ";:PRINT#1,USING "##.##";(YB*100);:PRINT #1,"%";:PRINT
#1,"    CUM YIELD= ";:PRINT #1,USING "##.##";(YC*100);:PRINT
#1,"%"
1640 IF STAM(4)=1 THEN GOSUB 7280
1645 IF STAM(15)=1 THEN PRINT"          < Space > to
continue....":GOTO 1647
1646 GOTO 1650
1647 Q$=INKEY$
1648 IF Q$=" " THEN GOTO 1650
1649 GOTO 1647
1650 IF STAM(15)=1 THEN GOSUB 7780
1660 IF STAM(13)=1 THEN PRINT"<ENTER> TO PROCESS ANOTHER BOARD;
<1> TO EXIT":INPUT ME:GOTO 1680
1670 GOTO 1700
1680 IF ME=0 THEN GOTO 1760
1690 GOTO 1950
1700 IF STAM(6)=1 AND J<=MJ THEN GOTO 1760
1710 IF SKIP=1 THEN GOTO 1730
1720 IF STAM(6)=2 AND J>=J1 AND GR=1 THEN
CC1=BFT:AC1=AAO:SKIP=1:GR=2:GOTO 100
1730 IF STAM(6)=2 AND GR=2 AND J>MJ THEN CC2=BFT-CC1:AC2=AAO-
AC1:GOTO 1950
1735 IF STAM(6)=1 AND J>MJ THEN GOTO 1950
1740 GOTO 1760
1760 DEFINT I,J,H,K:IF STAM(11)=1 AND STAM(13)=2 THEN ERASE
X,A,L:GOTO 650
1770 IF STAM(11)=2 OR STAM(13)=1 THEN ERASE
X,Y,A,B,L,SW,DEC:GOTO 650
1780 DATA 104,101,100,330,330,330,330
```

```
1790 DATA
33,39,41,43,51,55,57,95,99,107,119,121,123,125,129,139,141,143,
145,147
1800 DATA
2,3,5,6,7,8,9,10,11,12,17,56,65,66,67,68,69,79,101,104
1810 DATA
1.5,3.3,3.6,3.8,4.1,4.3,4.6,4.8,5.1,5.3,5.6,5.8,6.1,6.3,6.6,6.8
,7.1
1820 DATA 7.3,7.6,7.8,8.1,8.3,8.6,8.8,9.3,9.8,10.3
1830 DATA
1,4,6,8,9,16,23,26,29,30,33,39,42,44,57,64,68,77,79,81,83,88,92
,96,99,100,101
1840 DATA 0,1,2,3,4,5,6,7,8,10
1850 DATA 7,20,37,63,72,85,94,96,98,100
1860 DATA
1.2,3.7,6.2,8.7,11.2,13.7,16.2,18.7,21.2,23.7,26.2,28.7,31.2,33
.7,36.2,38.7,43.7,46.2,48.7,51.2,53.7,56.2,58.7,61.2,63.7,66.2,
68.7,71.2,73.7
1870 DATA 78.7,81.2,83.7,88.7,91.2,93.7,96.2,98.7
1880 DATA
13,26,36,51,55,57,64,68,72,79,83,85,95,97,106,115,134,136,149,1
59,161,170,177,197,207,214,227,240,244,259,268,275,288,301,308,
323,330
1890 DATA
3,6,8,11,13,16,18,21,23,26,28,31,33,36,38,41,43,46,48,51,53,56,
58,61,63,66,68,71,73,76,78,81,83,86,88,91,93,96
1900 DATA
7,29,39,52,65,78,85,92,105,107,117,126,133,135,139,146,156,172,
181,185,194,201,210,219,228,241,248,255,268,272,276,285,299,301
,305,319,326,330
1910 DATA
0.1,0.5,1,1.5,2,2.5,3,3.5,4,4.5,5,5.5,6,7,8,9,10,11,13,13.5,14,
17,19,20,22,24,28,31,35,40
1920 DATA
4,57,108,125,173,208,239,243,258,260,264,266,270,271,280,284,28
8,297,299,301,305,312,314,316,318,320,322,326,328,330
1930 DATA 0.1,0.5,0.7,1,1.2,1.5,1.7,2,2.2,2.5,3,4.2
1940 DATA 7,67,89,194,207,264,271,307,311,324,328,330
1950 PRINT:PRINT:PRINT:PRINT#1," ":PRINT#1," ":PRINT#1," "
1960 PRINT #1,"SUMMARY OF TICKETS STATUS FOR THIS RUN:"
1970 PRINT #1,"---------------------------------------
":PRINT#1," ":PRINT#1," "
1980 PRINT#1,"TICKET NO";"   LENGTH    ";"   BD FT CUT";"
REQUIREMENT";"    NO OF PCS";"   PRIORITY"
1990 PRINT#1,"---------";"   ------    ";"   ---------";"    --
---------";"    ---------";"   --------":PRINT#1," "
2000 FOR U=1 TO 30
2010 IF U=12 AND BFC(12)<>0 THEN PRINT#1,:PRINT#1,"LIST OF
COMPLETED TICKETS FOLLOWS:":PRINT#1,"-------------------------
--------":PRINT#1," "
2020 IF U>10 AND BFC(U)=0 THEN GOTO 2070
2030 PRINT#1,"  ";:PRINT #1,USING "##";U;:PRINT #1,"    ";"
";:PRINT #1,USING "##.##";TL(U)-.2;:PRINT #1,"      ";:PRINT
#1, USING "####.##";BFC(U);:PRINT #1,"       ";:PRINT #1,USING
"####.##";BF(U);:PRINT#1,"          ";
2040 IF WP(U)<>0 THEN PRINT#1,USING "###";INT(CUW(U)/WP(U));
```

```
2050 IF PR(U)<>14 THEN PRINT#1,"        ";PR(U):PRINT#1,"
":GOTO 2070
2060 PRINT#1,"        ";11:PRINT#1," "
2070 NEXT U
2080 PRINT#1,"Kerf used:";RKF;" Ins";"   Rip
allowance:";RA*100;" %"
2090 PRINT#1,"RUN#:";IRUN;"  TOTAL VOL OF RAW WOOD:
";:PRINT#1,USING "###.##";AAO;:PRINT #1," BD FT":PRINT #1,"
TOTAL USABLE VOL: ";
2100 PRINT#1,USING "###.##";BFT;:PRINT#1," BD FT":PRINT#1,"
CUM YIELD: ";:PRINT#1,USING "###.##";(YC*100);:PRINT#1,"%"
2110 CLOSE:EN$=TIME$:SOUND 1000,1:SOUND 1500,2
2120 CLS:PRINT" RUN COMPLETED. PLEASE WAIT...."
2130 FOR I=1 TO 2000
2140 NEXT I
2144 IF STAM(11)=1 AND STAM(13)=2 THEN ERASE X,A,L,SW:GOTO 2150
2146 IF STAM(11)=2 OR STAM(13)=1 THEN ERASE X,Y,A,B,L,SW,DEC
2150 STAM(16)=1:RIND=1:GOTO 60
2160 N=RND(1)*(I(Z)-1)+1
2170 RETURN
2240 FOR Z=1 TO (M-1)
2250 IF X(Z)>X(Z+1) THEN
R=X(Z):S=A(Z):X(Z)=X(Z+1):A(Z)=A(Z+1):X(Z+1)=R:A(Z+1)=S
2260 NEXT Z
2270 P=1
2280 IF X(P)<=X(P+1) THEN GOTO 2300
2290 GOTO 2240
2300 P=P+1
2310 IF P=M THEN GOTO 2330
2320 GOTO 2280
2330 RETURN
2340 REM CALCULATION OF CLEAR SECTION DIMENSIONS FOR CUTRIP
2350 GOSUB 3100
2360 GOSUB 3730
2370 GOSUB 3510
2380 GOTO 1230
2390 IF X(1)>X(2) THEN
R=X(1):X(1)=X(2):X(2)=R:AS=A(1):A(1)=A(2):A(2)=AS:YS=Y(1):Y(1)=
Y(2):Y(2)=YS:BS=B(1):B(1)=B(2):B(2)=BS
2400 GOSUB 3730
2410 GOSUB 3510
2420 GOTO 1280
2430 REM SUB PROGRAM FOR CUTRIP TO CLIP DEFECTS EXTENDING
OUTSIDE THE BOARD    BOUNDARIES
2440 IF B(D)<1 THEN B(D)=1
2450 IF X(D)-(A(D)/2)<0 THEN X(D)=A(D)/2
2460 IF X(D)+(A(D)/2)>LO THEN X(D)=LO-(A(D)/2)
2470 IF Y(D)-(B(D)/2)<(1+RKF) THEN
B(D)=(B(D)/2)+Y(D):Y(D)=B(D)/2
2480 IF WD-(Y(D)+(B(D)/2))<(1+RKF) THEN B(D)=WD-(Y(D)-
(B(D)/2)):Y(D)=WD-(B(D)/2)
2490 RETURN
2500 REM GENERATION OF CLEAR SECTIONS FOR M>=1
2510 I=0:D=1
2520 DIFF=(X(D)-(A(D)/2))-E
2530 IF DIFF>=16 THEN I=I+1:L(I)=DIFF:ST(I)=E
2540 IF D=M THEN GOTO 2600
```

```
2550 D=D+1
2560 DIFF=X(D)-X(D-1)
2570 IF DIFF>=16 THEN I=I+1:L(I)=DIFF-(A(D)/2)-(A(D-
1)/2):ST(I)=X(D-1)+(A(D-1)/2)
2580 IF D=M THEN GOTO 2600
2590 GOTO 2550
2600 DIFF=LO-E-X(D)
2610 IF DIFF>=16 THEN I=I+1:L(I)=DIFF-
(A(D)/2):ST(I)=X(D)+(A(D)/2)
2620 NSEC=I
2630 RETURN
2640 REM ELIMINATION OF DEFECT OVERLAPS/REVISED M,X,A
2650 D=1
2660 IF X(D+1)-X(D)>(A(D+1)/2)+(A(D)/2) THEN GOTO 2710
2670 AEQ=(A(D)/2)+(A(D+1)/2)+X(D+1)-X(D)
2680 XEQ=X(D)-(A(D)/2)+(AEQ/2)
2690 X(D+1)=XEQ:A(D+1)=AEQ
2700 X(D)=0:A(D)=0
2710 D=D+1
2720 IF D=M THEN GOTO 2740
2730 GOTO 2660
2740 I=1
2750 FOR B=1 TO M
2760 IF X(B)<>0 THEN TX(I)=X(B):TA(I)=A(B):I=I+1
2770 NEXT B
2780 M=I-1
2790 ERASE X,A,L:DIM X(M),A(M),L(M+1)
2800 FOR I=1 TO M
2810 X(I)=TX(I):A(I)=TA(I)
2820 NEXT I
2830 RETURN
2840 CLS
2850 PRINT "INPUT LENGTH(Ins) FOR BOARD ";J;"   *** VALID RANGE
35 TO 150 ***":INPUT LO
2860 IF LO<35 OR LO>150 THEN CLS:BEEP:PRINT"INPUT ERROR. TRY
AGAIN..":GOTO 2850
2870 PRINT "INPUT BOARD WIDTH(Ins) FOR BOARD ";J;"   *** VALID
RANGE 3 TO 15 ***":INPUT WO
2880 IF WO<3 OR WO>15 THEN CLS:BEEP:PRINT"INPUT ERROR. TRY
AGAIN..":GOTO 2870
2890 A1=LO*WO/144:AAO=AAO+A1
2900 WD=WO-(WO*RA)
2910 PRINT"INPUT NO OF DEFECTS FOR BOARD ";J;"   *** MAXIMUM 15
DEFECTS ***":INPUT M
2920 IF M<0 OR M>15 THEN CLS:BEEP:PRINT"INPUT ERROR. TRY
AGAIN..":GOTO 2910
2930 IF M=0 THEN DIM X(1),Y(1),A(1),B(1),SW(1),DEC(1):L(1)=LO-
(2*E):NSEC=1:GOTO 1280
2940 DIM X(M),A(M),Y(M),B(M),L(20),SW(20),DEC(20)
2950 IF M=1 THEN D=1:GOTO 2970
2960 FOR D=1 TO M
2970 PRINT"INPUT X COORDINATE OF DEFECT ";D:INPUT X(D)
2980 IF X(D)<=0 OR X(D)>=LO THEN CLS:BEEP:PRINT"INPUT ERROR.
TRY AGAIN..":GOTO 2970
2990 PRINT"INPUT Y COORDINATE OF DEFECT ";D:INPUT
Y(D):Y(D)=Y(D)-(WO*RA/2)
```

```
3000 IF Y(D)<=0 OR Y(D)>=WD THEN CLS:BEEP:PRINT"INPUT ERROR.
TRY AGAIN..":GOTO 2990
3010 PRINT"INPUT DEFECT LENGTH(Ins) FOR DEFECT ";D:INPUT A(D)
3020 IF A(D)<=0 OR A(D)>=LO THEN CLS:BEEP:PRINT"INPUT ERROR.
TRY AGAIN..":GOTO 3010
3025 IF X(D)-(A(D)/2)<0 OR X(D)+(A(D)/2)>LO THEN BEEP:PRINT"
PART OF DEFECT OUTSIDE BOARD BOUNDARY. <ENTER> TO CLIP DEFECT
AT BOUNDARY OR <1> TO CORRECT INPUT..":INPUT COR
3027 IF COR<>0 THEN CLS:GOTO 3010
3030 PRINT"INPUT DEFECT WIDTH(Ins) FOR DEFECT ";D:INPUT B(D)
3040 IF B(D)<=0 OR B(D)>WO THEN CLS:BEEP:PRINT"INPUT ERROR. TRY
AGAIN..":GOTO 3030
3045 IF Y(D)-(B(D)/2)<0 OR Y(D)+(B(D)/2)>WO THEN BEEP:PRINT"
PART OF DEFECT OUTSIDE BOARD BOUNDARY. <ENTER> TO CLIP DEFECT
AT BOUNDARY OR <1> TO CORRECT INPUT ..":INPUT COR
3047 IF COR<>0 THEN CLS:GOTO 3030
3050 IF STAM(11)=2 AND B(D)<1 THEN B(D)=1
3060 IF M=1 AND STAM(11)=1 THEN GOTO 1270
3070 IF M=1 AND STAM(11)=2 THEN GOTO 2410
3080 NEXT D
3090 RETURN
3100 FOR Z=1 TO (M-1)
3110 IF X(Z)>X(Z+1) THEN
R1=X(Z):R2=A(Z):R3=B(Z):R4=Y(Z):X(Z)=X(Z+1):A(Z)=A(Z+1):B(Z)=B(
Z+1):Y(Z)=Y(Z+1):X(Z+1)=R1:Y(Z+1)=R4:A(Z+1)=R2:B(Z+1)=R3
3120 NEXT Z
3130 P=1
3140 IF X(P)<=X(P+1) THEN GOTO 3160
3150 GOTO 3100
3160 P=P+1
3170 IF P=M THEN GOTO 3190
3180 GOTO 3140
3190 RETURN
3200 REM SUBPROGRAM TO MAKE CUT-RIP DECISION / INPUT TO THIS
SUBPROGRAM ARE STP ENP XT YT AT BT WD   /  CR=1 => RIP/ CR=0
=>CUT OFF
3210 CR=2:SL=ENP-STP
3220 SEL=SL:GOSUB 4360:WE=WASTE
3230 SEL=XT-(AT/2)-STP
3240 IF SEL<16 THEN WL=SEL:GOTO 3260
3250 GOSUB 4360:WL=WASTE
3260 SEL=ENP-XT-(AT/2)
3270 IF SEL<16 THEN WR=SEL:GOTO 3290
3280 GOSUB 4360:WR=WASTE
3290 IF WL<0 THEN WL=0
3300 IF WR<0 THEN WR=0
3310 CUTWA=WD*(AT+WL+WR)
3320 RIPWA=(BT*(AT+WL+WR))+((WD-BT)*WE)+(2*RKF*(SL-WE))
3330 IF CUTWA>RIPWA THEN CR=1:GOTO 3350
3340 CR=0
3350 DEC(D)=CR
3360 RETURN
3370 REM SUB PROGRAM TO CALCULATE SECTION LENGTHS FOR CUTOFF-
CR=0
3380 LEN1=XT-(AT/2)-STP:LEN2=ENP-XT-(AT/2)
3390 IF LEN1>=16 THEN CLSEC=CLSEC+1:L(CLSEC)=LEN1:SW(CLSEC)=WD
3400 IF D<>M THEN GOTO 3420
```

```
3410 IF LEN2>=16 THEN CLSEC=CLSEC+1:L(CLSEC)=LEN2:SW(CLSEC)=WD
3420 RETURN
3430 REM SUB PROGRAM TO CALCULATE SECTION LENGTHS FOR RIPPING-
CR=1
3440 LEN1=ENP-STP
3450 LEN2=XT-(AT/2)-STP
3460 LEN3=ENP-XT-(AT/2)
3470 IF LEN1>=16 AND WD-BT>0 THEN
CLSEC=CLSEC+1:L(CLSEC)=LEN1:SW(CLSEC)=WD-(BT+(2*RKF))
3480 IF LEN2>=16 AND BT>0 THEN
CLSEC=CLSEC+1:L(CLSEC)=LEN2:SW(CLSEC)=BT
3490 IF LEN3>=16 AND BT>0 THEN
CLSEC=CLSEC+1:L(CLSEC)=LEN3:SW(CLSEC)=BT
3500 RETURN
3510 REM GENERATION OF CLEAR SECTIONS FOR M>=1
3520 CLSEC=0:D=1
3530 IF M<>1 THEN GOTO 3580
3540 STP=E:ENP=LO-E:XT=X(1):YT=Y(1):AT=A(1):BT=B(1)
3550 GOSUB 3200
3560 IF DEC(D)=0 THEN GOSUB 3370:NSEC=CLSEC:RETURN
3570 GOSUB 3430:NSEC=CLSEC:RETURN
3580 IF D<>1 THEN GOTO 3630
3590 STP=E:ENP=X(D+1)-
(A(D+1)/2):XT=X(D):YT=Y(D):AT=A(D):BT=B(D)
3600 GOSUB 3200
3610 IF DEC(D)=0 THEN GOSUB 3370:D=D+1:GOTO 3580
3620 GOSUB 3430:D=D+1:GOTO 3580
3630 IF D=M THEN GOTO 3680
3640 IF DEC(D-1)=0 THEN STP=X(D-1)+(A(D-1)/2):ENP=X(D+1)-
(A(D+1)/2):AT=A(D):BT=B(D):XT=X(D):YT=Y(D):GOTO 3600
3650 DEC(D)=0
3660 IF A(D)>=16 AND WD-B(D)>0 THEN
CLSEC=CLSEC+1:L(CLSEC)=A(D):SW(CLSEC)=WD-B(D)
3670 D=D+1:GOTO 3580
3680 IF DEC(D-1)=0 THEN STP=X(D-1)+(A(D-1)/2):ENP=LO-
E:XT=X(D):YT=Y(D):AT=A(D):BT=B(D):GOTO 3550
3690 STP=X(D)-(A(D)/2):ENP=LO-E:XT=X(D):YT=Y(D):AT=A(D):BT=B(D)
3700 GOSUB 3200
3710 IF DEC(D)=0 THEN GOSUB 3370:NSEC=CLSEC:RETURN
3720 GOSUB 3430:NSEC=CLSEC:RETURN
3730 REM ELIMINATION OF DEFECT OVERLAPS / REVISED M,X,Y,A,B
3740 D=1
3750 IF X(D+1)-X(D)>(A(D+1)/2)+(A(D)/2) THEN GOTO 3860
3760 AEQ=(A(D+1)/2)+(A(D)/2)+X(D+1)-X(D)
3770 BEQ=ABS(Y(D+1)-Y(D))+(B(D+1)/2)+(B(D)/2)
3780 XEQ=X(D)-(A(D)/2)+(AEQ/2)
3790 IF Y(D)>Y(D+1) THEN YS=Y(D+1):BS=B(D+1):GOTO 3830
3800 IF Y(D)=Y(D+1) AND B(D)>B(D+1) THEN
YS=Y(D):BS=B(D):BEQ=B(D):GOTO 3830
3810 IF Y(D)=Y(D+1) AND B(D)<B(D+1) THEN
YS=Y(D):BS=B(D+1):BEQ=B(D+1):GOTO 3830
3820 YS=Y(D):BS=B(D)
3830 YEQ=YS-(BS/2)+(BEQ/2)
3840 X(D+1)=XEQ:Y(D+1)=YEQ:A(D+1)=AEQ:B(D+1)=BEQ
3850 X(D)=0:Y(D)=0:A(D)=0:B(D)=0:GOTO 3870
3860 IF (X(D+1)-X(D)-(A(D+1)/2)-(A(D)/2))<16 AND ABS(Y(D)-
Y(D+1))<(.5*WD) THEN GOTO 3760
```

```
3870 D=D+1
3880 IF D=M THEN GOTO 3900
3890 GOTO 3750
3900 I=1
3910 FOR B=1 TO M
3920 IF X(B)<>0 THEN
TX(I)=X(B):TA(I)=A(B):TB(I)=B(B):TY(I)=Y(B):I=I+1
3930 NEXT B
3940 M=I-1
3950 ERASE X,Y,A,B:DIM X(M),Y(M),A(M),B(M)
3960 FOR I=1 TO M
3970 X(I)=TX(I):Y(I)=TY(I):A(I)=TA(I):B(I)=TB(I)
3980 NEXT I
3990 RETURN
4000 REM SUBPROGRAM FOR HEURISTIC WITH PRIORITIES 1-11
4010 SI=0:SLEN=L(P):TOL=0:DIM SOL(10):OY=0
4020 FOR I=1 TO 10
4030 IF TL(I)>SLEN OR TL(I)<16 THEN GOTO 4090
4040 N1=INT(SLEN/TL(I)):WAS=SLEN-(N1*TL(I)):YCUR=(SLEN-
WAS)/SLEN
4050 IF PR(I)=14 THEN YCUR=YCUR*50:GOTO 4080
4060 IF PR(I)=0 THEN GOTO 4080
4070 YCUR=YCUR*(EXP(.35*PR(I)))
4080 IF WAS>=0 AND YCUR>OY THEN OY=YCUR:OI=I
4090 NEXT I
4100 IF OY<>0 THEN
SI=SI+1:SOL(SI)=OI:QTY(OI)=QTY(OI)+1:SLEN=SLEN-TL(OI):OY=0:GOTO
4020
4110 IF OY=0 AND SI=0 THEN GOTO 4230
4120 FOR I=1 TO SI
4130 TN=SOL(I)
4170 TOL=TOL+(TL(TN)-RKF):BFC(TN)=BFC(TN)+((TL(TN)-
RKF)*SW(P)/144):CUW(TN)=CUW(TN)+SW(P)
4180 NEXT I
4190 BFSEC=BFSEC+(TOL*SW(P)/144)
4200 FOR I=1 TO 10
4210 IF BFC(I)>BF(I) THEN TF=I:GOSUB 4250
4220 NEXT I
4230 ERASE SOL
4232 FOR I=1 TO 10
4234 IF QTY(I)=0 THEN GOTO 4238
4235 PRINT#1," TICKET NUMBER: ";I;"  TICKET LENGTH: ";:PRINT#1,
USING "##.##";TL(I)-RKF;:PRINT #1,"   NUMBER OF PIECES:
";QTY(I):PRINT#1," "
4236 PRINT" TICKET NUMBER: ";I;"  TICKET LENGTH: ";:PRINT USING
"##.##";TL(I)-RKF;:PRINT "   NUMBER OF PIECES: ";QTY(I):PRINT
4238 NEXT I
4240 RETURN
4250 REM REPLACE FINISHED TICKETS
4260 SOUND 200,2:SOUND 400,4
4270 TL(RET)=TL(TF)-
RKF:PR(RET)=PR(TF):PC(RET)=PC(TF):W(RET)=W(TF):CUW(RET)=CUW(TF)
:WP(RET)=WP(TF):BF(RET)=BF(TF):BFC(RET)=BFC(TF):BFC(TF)=0:RET=R
ET+1:PRINT"TICKET NUMBER";TF;"OF LENGTH ";:PRINT USING
"##.##";TL(TF)-RKF;
4280 PRINT" COMPLETED. REPLACE WITH NEW TICKET."
```

```
4290 PRINT"TICKET LENGTH (INS)?":INPUT TL(TF):PRINT"TICKET
WIDTH(INS)?":INPUT W(TF):WP(TF)=W(TF)+RKF:CUW(TF)=0:CLS
4300 IF STAM(2)=1 THEN PRINT" INPUT TICKET REQUIREMENT IN BD
FT":INPUT BF(TF):GOTO 4320
4310 PRINT" INPUT TICKET REQUIREMENT AS NUMBER OF PIECES":INPUT
PC(TF):BF(TF)=TL(TF)*W(TF)*PC(TF)/144
4320 IF STAM(4)=1 THEN PRINT" INPUT PRIORITY FOR THIS
TICKET":INPUT PR(TF)
4330 IF STAM(4)=1 AND PR(TF)=11 THEN PR(TF)=14
4340 TL(TF)=TL(TF)+RKF
4350 RETURN
4360 REM SUBPROGRAM TO FIND WASTE FOR AN ARBITRARY LENGTH SEL
4370 OW1=2000:OW2=2000
4380 FOR I=1 TO 10
4390 IF TL(I)-RKF>SEL THEN GOTO 4450
4400 IF TL(I)-RKF=SEL THEN WAS1=0:GOTO 4440
4410 N1=INT(SEL/TL(I))
4420 IF (((N1+1)*TL(I))-RKF)=SEL THEN WAS1=0:GOTO 4440
4430 WAS1=SEL-(N1*TL(I))
4440 IF WAS1>=0 AND WAS1<OW1 THEN OW1=WAS1
4450 NEXT I
4460 FOR I=1 TO 10
4470 FOR H=(I+1) TO 10
4480 IF (TL(I)+TL(H)-RKF)>SEL THEN GOTO 4540
4490 IF (TL(I)+TL(H)-RKF)=SEL THEN WAS2=0:GOTO 4530
4500 N2=INT(SEL/(TL(I)+TL(H)))
4510 IF (((N2+1)*(TL(I)+TL(H)))-RKF)=SEL THEN WAS2=0:GOTO 4530
4520 WAS2=SEL-((TL(I)+TL(H))*N2)
4530 IF WAS2>=0 AND WAS2<OW2 THEN OW2=WAS2
4540 NEXT H
4550 NEXT I
4560 IF OW1=2000 AND OW2=2000 THEN WASTE=SEL:GOTO 4600
4570 IF WAS1=0 OR WAS2=0 THEN WASTE=0:GOTO 4600
4580 IF WAS1<WAS2 THEN WASTE=WAS1:GOTO 4600
4590 WASTE=WAS2
4600 RETURN
4610 REM SUBPROGRAM TO SET TICKET DATA
4611 IF RKF<>0 THEN CLS:GOTO 4620
4612 CLS:PRINT"INPUT KERF IN INCHES      ** RANGE: 0-0.4 INCHES
**"
4614 INPUT RKF
4616 IF RKF<0 OR RKF>.4 THEN BEEP:CLS:PRINT"INPUT ERROR. TRY
AGAIN..":GOTO 4614
4620 IF STAM(1)=1 THEN STAM(1)=0:ERASE
TL,W,BF,BFC,PC,WP,CUW,DF,ID
4630 CLS:PRINT"CHOOSE ONE OF THE OPTIONS BELOW....":PRINT:PRINT
4640 PRINT"    <A>  TO ENTER TICKET REQUIREMENT IN BD FT":PRINT
4650 PRINT"    <B>  TO ENTER TICKET REQUIREMENT AS NUMBER OF
PIECES"
4655 PRINT:PRINT:PRINT"    CHOOSE ONE OF THE OPTIONS OR <SPACE
> TO RETURN TO MAIN MENU..."
4660 Q$=INKEY$
4670 IF Q$="A" OR Q$="a" THEN Q$="":STAM(2)=1:GOTO 4700
4680 IF Q$="B" OR Q$="b" THEN Q$="":STAM(2)=2:GOTO 4700
4685 IF Q$=" " THEN RETURN
4690 GOTO 4660
4700 DIM TL(30),W(30),BF(30),BFC(30),PC(30)
```

```
4710 DIM WP(30),CUW(30),DF(10),ID(10)
4720 FOR I=1 TO 10
4730 CLS:LOCATE 22,10:PRINT"< ENTER > TO EXIT...":LOCATE 1,1
4740 PRINT"Input Ticket Length (Ins) for Ticket #";I;" ***
VALID RANGE 16 TO 85 ***":INPUT TL(I):TL(I)=TL(I)+RKF
4742 IF TL(I)-RKF=0 AND I=1 THEN ERASE
TL,W,BF,BFC,PC,WP,CUW,DF,ID:RETURN
4745 IF TL(I)-RKF=0 THEN TI=I:GOTO 4842
4750 IF TL(I)-RKF<16 OR TL(I)-RKF>85 THEN CLS:BEEP:PRINT "INPUT
ERROR. TRY AGAIN..":GOTO 4740
4760 PRINT"Input Ticket Width (Ins) for Ticket #";I;"  ***
VALID RANGE 1 TO 48 ***":INPUT W(I):WP(I)=W(I)+RKF
4770 IF W(I)<=0 OR W(I)>48 THEN CLS:BEEP:PRINT "INPUT ERROR.
TRY AGAIN..":GOTO 4760
4780 IF STAM(2)=1 THEN GOTO 4820
4790 IF STAM(2)=2 THEN PRINT"INPUT NO OF PIECES
REQUIRED.":INPUT PC(I):BF(I)=(TL(I)-RKF)*W(I)*PC(I)/144
4795 IF BF(I)>900 THEN BEEP:CLS:PRINT"TICKET REQUIREMENT CAN'T
EXCEED 900 BD FT. REDUCE NUMBER OF PIECES REQUIRED..":GOTO 4790
4800 IF PC(I)<=0 THEN CLS:BEEP:PRINT"INPUT ERROR. TRY
AGAIN..":GOTO 4790
4810 GOTO 4840
4820 PRINT"Input Ticket Requirement (Bd Ft) for Ticket
#";I;"*** MAX 900 BD FT ***":INPUT BF(I)
4830 IF BF(I)<=0 OR BF(I)>900 THEN CLS:BEEP:PRINT"INPUT ERROR.
TRY AGAIN..":GOTO 4820
4840 NEXT I
4842 IF TI<>0 AND TI<>10 THEN GOTO 4844
4843 GOTO 4850
4844 FOR I=TI TO 10
4845 TL(I)=300
4848 NEXT I
4850 TI=10:STAM(1)=1:CLS
4860 TL(11)=0:RET=12
4870 BFO=0
4880 FOR I=1 TO 10
4890 BFO=BFO+BF(I)
4900 NEXT I
4910 RETURN
4920 REM SUB PROGRAM TO AUTOMATICALLY ASSIGN BEST PRIORITIES
4930 FOR I=1 TO 10
4940 DF(I)=BF(I)-BFC(I)
4950 ID(I)=I
4960 NEXT I
4970 CHK=0
4980 FOR I=1 TO 9
4990 IF DF(I)<DF(I+1) THEN
SDF=DF(I+1):DF(I+1)=DF(I):DF(I)=SDF:SID=ID(I+1):ID(I+1)=ID(I):I
D(I)=SID:CHK=2:GOTO 5020
5000 REM DUMMY STATEMENT
5010 IF DF(I)=DF(I+1) AND TL(ID(I))<TL(ID(I+1)) THEN
SDF=DF(I+1):DF(I+1)=DF(I):DF(I)=SDF:SID=ID(I+1):ID(I+1)=ID(I):I
D(I)=SID:CHK=2
5020 NEXT I
5030 IF CHK=2 THEN GOTO 4970
5040 MP=10
5050 FOR I=1 TO 10
```

```
5060 IF TL(ID(I))>=200 THEN PR(ID(I))=0:GOTO 5090
5070 PR(ID(I))=MP
5080 MP=MP-1
5090 NEXT I
5100 RETURN
5110 REM SUBPROGRAM TO CONTROL MAIN MENU
5120 CLS:Q$="":GOSUB 5240
5130 Q$=INKEY$
5140 IF Q$="A" OR Q$="a" THEN Q$="":GOSUB 4610:GOTO 5120
5150 IF Q$="B" OR Q$="b" THEN Q$="":GOSUB 5500:GOTO 5120
5160 IF Q$="C" OR Q$="c" THEN Q$="":GOSUB 5630:GOTO 5120
5170 IF Q$="D" OR Q$="d" THEN Q$="":GOSUB 6050:GOTO 5120
5180 IF Q$="E" OR Q$="e" THEN Q$="":GOSUB 6160:GOTO 5120
5190 IF Q$="F" OR Q$="f" THEN Q$="":GOSUB 6270:GOTO 5120
5200 IF Q$="G" OR Q$="g" THEN Q$="":GOSUB 6390:GOTO 5120
5210 IF Q$="H" OR Q$="h" THEN Q$="":GOSUB 6550:GOTO 5120
5220 IF Q$="I" OR Q$="i" THEN CLOSE:CLS:SYSTEM
5225 IF Q$="S" OR Q$="s" THEN Q$="":GOSUB 8500:GOTO 5120
5230 GOTO 5130
5240 REM SUBPROGRAM TO PRINT MAIN MENU
5250 CLS:PRINT:PRINT
5260 PRINT"
5555555555555555555555555555555555555555555555555555555555555555555
5555555"
5270 PRINT" |                              MAIN MENU
|"
5280 PRINT" |                              ~~~~~~~~~
|"
5290 PRINT" |
|"
5295 IF STAM(1)<>0 THEN COLOR 12,9:PRINT" |                    <A>
INPUT TICKET DATA                                  |":COLOR
14,9:GOTO 5310
5300 PRINT" |                  <A>  INPUT TICKET DATA
|"
5310 PRINT" |
|"
5315 IF STAM(3)<>0 THEN COLOR 12,9:PRINT" |                    <B>
INPUT PRIORITY                                     |":COLOR
14,9:GOTO 5330
5320 PRINT" |                  <B>  INPUT PRIORITY
|"
5330 PRINT" |
|"
5335 IF STAM(5)<>0 THEN COLOR 12,9:PRINT" |                    <C>
INPUT GRADE(S)                                     |":COLOR
14,9:GOTO 5350
5340 PRINT" |                  <C>  INPUT GRADE(S)
|"
5350 PRINT" |
|"
5355 IF STAM(10)<>0 THEN COLOR 12,9:PRINT" |
<D>  SELECT CUTTING STRATEGY                              |":COLOR
14,9:GOTO 5370
5360 PRINT" |                  <D>  SELECT CUTTING STRATEGY
|"
```

```
5370 PRINT" |
|"
5375 IF STAM(12)<>0 THEN COLOR 12,9:PRINT" |
<E>  BOARD DATA INPUT                                    |":COLOR
14,9:GOTO 5390
5380 PRINT" |                  <E>  BOARD DATA INPUT
|"
5390 PRINT" |
|"
5395 IF STAM(14)<>0 THEN COLOR 12,9:PRINT" |
<F>  PRODUCTION vs REQUIREMENT GRAPH OPTION          |":COLOR
14,9:GOTO 5410
5400 PRINT" |                  <F>  PRODUCTION vs REQUIREMENT
GRAPH OPTION        |"
5410 PRINT" |
|"
5415 IF RIND=1 THEN COLOR 12,9:PRINT" |                  <G>
RUN                                                |":COLOR
14,9:GOTO 5430
5420 PRINT" |                  <G>  RUN
|"
5430 PRINT" |
|"
5435 IF STAM(17)<>0 AND RIND=1 THEN COLOR 12,9:PRINT" |
<H>  DISPLAY SUMMARY OF OUTPUT                       |":COLOR
14,9:GOTO 5450
5440 PRINT" |                  <H>  DISPLAY SUMMARY OF OUTPUT
|"
5450 PRINT" |
|"
5460 PRINT" |                  <I>  QUIT (EXIT TO DOS)
|"
5470 PRINT" |      SELECT REQUIRED OPTION / <S> TO DISPLAY
CURRENT CHOICES...     |"
5480 PRINT"
|555555555555555555555555555555555555555555555555555555555555555555
555555|"
5490 RETURN
5500 REM SUBPROGRAM TO SET PRIORITY OPTIONS
5510 CLS:PRINT:PRINT:PRINT:PRINT
5520 PRINT"                        PRIORITY OPTIONS"
5530 PRINT"                        ~~~~~~~~ ~~~~~~~~":PRINT:PRINT
5540 PRINT"             <A>  MANUALLY UPDATE PRIORITIES":PRINT
5550 PRINT"             <B>  AUTOMATICALLY UPDATE
PRIORITIES":PRINT
5560 PRINT"             <C>  NO PRIORITIES":PRINT:PRINT
5570 PRINT"             SELECT ONE OF THE OPTIONS OR <SPACE >
TO RETURN TO MAIN MENU..."
5580 Q$=INKEY$
5590 IF Q$="A" OR Q$="a" THEN Q$="":GOSUB
6800:STAM(4)=1:STAM(3)=1:RETURN
5600 IF Q$="B" OR Q$="b" THEN Q$="":STAM(4)=2:STAM(3)=1:RETURN
5610 IF Q$="C" OR Q$="c" THEN Q$="":GOSUB
6890:STAM(4)=3:STAM(3)=1:RETURN
5615 IF Q$=" " THEN RETURN
5620 GOTO 5580
5630 REM SUBPROGRAM TO SET GRADES
```

```
5640 CLS:PRINT:PRINT
5650 PRINT"          GRADE OPTIONS"
5660 PRINT"          ~~~~~ ~~~~~~~":PRINT:PRINT
5670 PRINT"          <A>  PROCESS SINGLE GRADE":PRINT
5680 PRINT"          <B>  MIX GRADES":PRINT:PRINT
5690 PRINT"          SELECT ONE OF THE OPTIONS OR <SPACE > TO
RETURN TO MAIN MENU..."
5700 Q$=INKEY$
5710 IF Q$="A" OR Q$="a" THEN Q$="":STAM(6)=1:GOTO 5740
5720 IF Q$="B" OR Q$="b" THEN Q$="":STAM(6)=2:GOTO 5840
5725 IF Q$=" " THEN RETURN
5730 GOTO 5700
5740 CLS:PRINT:PRINT:PRINT:PRINT
5750 PRINT"          GRADES AVAILABLE"
5760 PRINT"          ~~~~~~ ~~~~~~~~~":PRINT:PRINT
5770 PRINT"          <A>  NUMBER 1 COMMON":PRINT
5780 PRINT"          <B>  NUMBER 2 COMMON":PRINT:PRINT
5790 PRINT"          SELECT ONE OF THE OPTIONS OR <SPACE > TO
RETURN TO MAIN MENU..."
5800 Q$=INKEY$
5810 IF Q$="A" OR Q$="a" THEN Q$="":STAM(7)=1:STAM(5)=1:RETURN
5820 IF Q$="B" OR Q$="b" THEN Q$="":STAM(7)=2:STAM(5)=1:RETURN
5825 IF Q$=" " THEN RETURN
5830 GOTO 5800
5840 CLS:PRINT:PRINT:PRINT:PRINT
5850 PRINT"          GRADES AVAILABLE"
5860 PRINT"          ~~~~~~ ~~~~~~~~~":PRINT:PRINT
5870 PRINT"          <A>  NUMBER 1 COMMON":PRINT
5880 PRINT"          <B>  NUMBER 2 COMMON":PRINT:PRINT
5890 PRINT"          SELECT REQUIRED
GRADE(S)":PRINT:PRINT:PRINT"                          < SPACE > TO
EXIT TO MAIN MENU"
5900 Q$=INKEY$
5910 IF Q$="B" OR Q$="b" THEN GOTO 5980
5920 IF Q$=" " THEN GOTO 6020
5930 IF Q$="A" OR Q$="a" THEN Q$="":STAM(8)=1:CLS:PRINT"INPUT
% IN MIX":INPUT C1:GOTO 5950
5940 GOTO 5900
5950 IF C1>1 THEN C1=C1/100:C2=1-C1:GOTO 5840
5960 IF C1=0 THEN C2=1:GOTO 5840
5970 C2=1-C1:GOTO 5840
5980 IF Q$="B" OR Q$="b" THEN Q$="":STAM(9)=1:CLS:PRINT"INPUT
% IN MIX":INPUT C2
5990 IF C2>1 THEN C2=C2/100:C1=1-C2:GOTO 5840
6000 IF C2=0 THEN C1=1:GOTO 5840
6010 C1=1-C2:GOTO 5840
6020 IF Q$=" " AND (STAM(8)=1 OR STAM(9)=1) THEN
Q$="":STAM(5)=1
6030 IF C2>0 THEN STAM(9)=1:RETURN
6040 GOTO 5840
6050 REM SUBPROGRAM TO SET CUTTING STRATEGY
6060 CLS:PRINT:PRINT:PRINT:PRINT
6070 PRINT"          CUTTING STRATEGY OPTIONS"
6080 PRINT"          ~~~~~~~ ~~~~~~~~ ~~~~~~~":PRINT:PRINT:PRINT
6090 PRINT"          <A>  CUT-OFF":PRINT
6100 PRINT"          <B>  CUT-RIP":PRINT:PRINT
```

```
6110 PRINT"          SELECT REQUIRED STRATEGY OR < SPACE > TO
RETURN TO MAIN MENU..."
6120 Q$=INKEY$
6130 IF Q$="A" OR Q$="a" THEN
Q$="":STAM(11)=1:STAM(10)=1:RETURN
6140 IF Q$="B" OR Q$="b" THEN
Q$="":STAM(11)=2:STAM(10)=1:RETURN
6145 IF Q$=" " THEN RETURN
6150 GOTO 6120
6160 REM SUBPROGRAM TO SET BOARD DATA OPTIONS
6162 CLS:PRINT"INPUT END TRIM (INS)          ** RANGE  0-5 INS
**":INPUT E
6164 IF E<0 OR E>5 THEN BEEP:PRINT"INPUT ERROR. TRY
AGAIN..":GOTO 6162
6165 CLS:PRINT"INPUT EDGE RIP ALLOWANCE AS % OF BOARD WIDTH
** RANGE 0-10 % **":INPUT RA
6166 IF RA>1 THEN RA=RA/100
6167 IF RA<0 OR RA>.1 THEN BEEP:PRINT"INPUT ERROR. TRY
AGAIN..:GOTO 6165
6168 IF STAM(6)=2 THEN GOTO 6178
6170 CLS:PRINT"INPUT NUMBER OF BOARDS TO BE
PROCESSED.":PRINT"(THE TOTAL BOARD FOOTAGE REQUIRED FOR THE
INPUT TICKETS IS";BFO;" BDFT)":INPUT MJ:PJ=MJ:MJ=MJ-1
6175 IF MJ<0 THEN BEEP:GOTO 6170
6178 CLS:PRINT:PRINT:PRINT:PRINT
6180 PRINT"          BOARD DATA INPUT OPTIONS"
6190 PRINT"          ~~~~~ ~~~~ ~~~~~ ~~~~~~~":PRINT:PRINT
6200 PRINT"          <A>  MANUALLY INPUT BOARD DATA ONE AT A
TIME":PRINT
6210 PRINT"          <B>  GENERATE SIMULATED BOARDS":PRINT:PRINT
6220 PRINT"          SELECT ONE OF THE OPTIONS OR <SPACE > TO
RETURN TO MAIN MENU..."
6230 Q$=INKEY$
6240 IF Q$="A" OR Q$="a" THEN
Q$="":STAM(13)=1:STAM(12)=1:RETURN
6250 IF Q$="B" OR Q$="b" THEN
Q$="":STAM(13)=2:STAM(12)=1:RETURN
6255 IF Q$=" " THEN RETURN
6260 GOTO 6230
6270 REM SUBPROGRAM TO SET GRAPH OPTIONS
6280 CLS:PRINT:PRINT:PRINT:PRINT
6290 PRINT"          GRAPH DISPLAY OPTIONS"
6300 PRINT"          ~~~~~ ~~~~~~~ ~~~~~~~":PRINT:PRINT
6310 PRINT"          <A> PAUSE AFTER EACH BOARD TO VIEW GRAPH
OF"
6320 PRINT"              PRODUCTION vs REQUIREMENT":PRINT
6330 PRINT"          <B> DO NOT DISPLAY GRAPH":PRINT:PRINT
6340 PRINT"          SELECT ONE OF THE OPTIONS OR <SPACE > TO
RETURN TO MAIN MENU..."
6350 Q$=INKEY$
6360 IF Q$="A" OR Q$="a" THEN
Q$="":STAM(15)=1:STAM(14)=1:RETURN
6370 IF Q$="B" OR Q$="b" THEN
Q$="":STAM(15)=2:STAM(14)=1:RETURN
6375 IF Q$=" " THEN RETURN
6380 GOTO 6350
6390 REM SUBPROGRAM TO CHECK DATA AND PRINT ERROR MESSAGES
```

```
6400 CLS:STAM(18)=0
6410 IF STAM(1)=0 THEN PRINT"    TICKET DATA NOT
AVAILABLE":PRINT:STAM(18)=1
6420 IF STAM(3)=0 THEN PRINT"    TICKET PRIORITIES NOT
SPECIFIED":PRINT:STAM(18)=1
6430 IF STAM(5)=0 THEN PRINT"    GRADES TO BE PROCESSED NOT
SPECIFIED - USE OPTION <B>":PRINT:STAM(18)=1
6440 IF STAM(10)=0 THEN PRINT"    CUTTING STRATEGY NOT
SPECIFIED - USE OPTION <D>":PRINT:STAM(18)=1
6450 IF STAM(12)=0 THEN PRINT"    USE OPTION <E> TO SPECIFY
BOARD DATA INPUT":PRINT:STAM(18)=1
6460 IF STAM(14)=0 THEN PRINT"    USE OPTION <F> TO SPECIFY
GRAPH DISPLAY OPTIONS":PRINT:STAM(18)=1
6470 IF RIND<>0 AND STAM(17)=0 THEN PRINT"    OPTION <H> MUST
BE USED BEFORE THE NEXT RUN CAN BE MADE":PRINT:STAM(18)=1
6480 IF STAM(6)=2 AND STAM(13)=1 THEN PRINT"     GRADE MIXING
NOT POSSIBLE WITH MANUAL INPUT OF BOARD DIMENSIONS":PRINT"
RESET ONE OF THE OPTIONS <C> OR <E>":STAM(18)=1
6490 IF STAM(18)=1 THEN BEEP:BEEP:GOTO 6510
6495 IF RIND=1 THEN GOSUB 9000
6500 GOTO 70
6510 PRINT:PRINT:PRINT:PRINT"                                  < SPACE
> TO EXIT TO MAIN MENU...."
6520 Q$=INKEY$
6530 IF Q$=" " THEN RETURN
6540 GOTO 6520
6550 REM SUBPROGRAM TO PRINT OUTPUT SUMMARY
6560 CLS:PRINT:PRINT
6570 PRINT"                              SUMMARY OF OUTPUT"
6580 PRINT"                              ~~~~~~~ ~~ ~~~~~~":PRINT
6590 PRINT"                    DATE : ";DATE$:PRINT
6600 PRINT"               RUN START : ";ST$
6610 PRINT"                 RUN END : ";EN$:PRINT
6620 PRINT"    GRADE(S) PROCESSED :";
6630 IF STAM(6)=1 AND STAM(7)=1 THEN PRINT" NO 1 COM":GOTO 6700
6640 IF STAM(6)=1 AND STAM(7)=2 THEN PRINT" NO 2 COM":GOTO 6700
6650 IF STAM(6)=2 AND STAM(8)=1 THEN PRINT" NO 1 COM";
6660 IF STAM(6)=2 AND STAM(9)=1 THEN PRINT"  NO 2 COM"
6670 IF STAM(6)=2 THEN PRINT:PRINT"              % IN MIX :";
6680 IF STAM(6)=2 AND C1<>0 THEN PRINT"   ";:PRINT USING
"##.##";C1*100;
6690 IF STAM(6)=2 AND C2<>0 THEN PRINT"     ";:PRINT USING
"##.##";C2*100
6700 PRINT
6710 PRINT" TOTAL VOL OF RAW WOOD : ";:PRINT USING
"####.##";AAO;:PRINT" BD FT"
6720 PRINT"      TOTAL USABLE VOL : ";:PRINT USING
"####.##";BFT;:PRINT" BD FT":PRINT
6730 PRINT"                  KERF : ";RKF;" INS"
6735 PRINT"              END TRIM : ";E;" INS"
6740 PRINT"         RIP ALLOWANCE :";RA*100;"% OF BOARD
WIDTH":PRINT
6750 PRINT"      CUMULATIVE YIELD : ";:PRINT USING
"##.##";YC*100;:PRINT" %":PRINT:PRINT
6760 PRINT"     < SPACE > TO EXIT TO MAIN MENU...."
6770 Q$=INKEY$
6780 IF Q$=" " THEN Q$="":STAM(17)=1:RETURN
```

```
6790 GOTO 6770
6800 REM SUBPROGRAM TO INPUT PRIORITIES
6810 FOR I=1 TO 10
6820 CLS:PRINT"INPUT PRIORITY FOR TICKET #";I;" -
LENGTH:";TL(I);" INS"
6830 PRINT"NOTE: PRIORITY 0 ==> NO PRIORITY"
6840 PRINT"      PRIORITY 11==> TOP MOST PRIORITY":INPUT PR(I)
6850 IF PR(I)>11 THEN CLS:BEEP:PRINT"INPUT ERROR. TRY
AGAIN..":GOTO 6820
6860 IF PR(I)=11 THEN PR(I)=14
6870 NEXT I
6880 RETURN
6890 REM SUBPROGRAM TO SET PR(I) TO 0
6900 FOR I=1 TO 30
6910 PR(I)=0
6920 NEXT I
6930 RETURN
6940 REM SUBPROGRAM TO CALCULATE GRADE MIX
6950 BFO=0
6960 FOR I=1 TO 10
6970 BFO=BFO+BF(I)
6980 NEXT I
6985 CLS:PRINT"INPUT NUMBER OF BOARDS TO BE
PROCESSED.":PRINT"(THE TOTAL BOARD FOOTAGE REQUIRED FOR THE
INPUT TICKETS IS";BFO;"BD FT)":INPUT MJ:PJ=MJ:MJ=MJ-1
6987 IF MJ<0 THEN BEEP:GOTO 6985
6990 J1=MJ*C1:J2=MJ:BFT1=BFO*C1:BFT2=BFO-BFT1
7000 IF J1=0 THEN GR=2:GOTO 7020
7010 GR=1
7020 RETURN
7030 REM SUBPROGRAM FOR HEURISTIC WITH PRIORITIES 1-11
7040 SI=0:SLEN=L(P):TOL=0:DIM SOL(10):OY=0
7050 FOR I=1 TO 10
7060 IF TL(I)>SLEN OR TL(I)<16 THEN GOTO 7120
7070 N1=INT(SLEN/TL(I)):WAS=SLEN-(N1*TL(I)):YCUR=(SLEN-
WAS)/SLEN
7080 IF PR(I)=14 THEN YCUR=YCUR*50:GOTO 7110
7090 IF PR(I)=0 THEN GOTO 7110
7100 YCUR=YCUR*(EXP(.35*PR(I)))
7110 IF WAS>=0 AND YCUR>OY THEN OY=YCUR:OI=I
7120 NEXT I
7130 IF OY<>0 THEN
SI=SI+1:SOL(SI)=OI:QTY(OI)=QTY(OI)+1:SLEN=SLEN-TL(OI):OY=0:GOTO
7050
7140 IF OY=0 AND SI=0 THEN GOTO 7260
7150 FOR I=1 TO SI
7160 TN=SOL(I)
7200 TOL=TOL+(TL(TN)-RKF):BFC(TN)=BFC(TN)+((TL(TN)-
RKF)*WD/144):CUW(TN)=CUW(TN)+WD
7210 NEXT I
7220 BFSEC=BFSEC+(TOL*WD/144)
7230 FOR I=1 TO 10
7240 IF BFC(I)>BF(I) THEN TF=I:GOSUB 7390
7250 NEXT I
7260 ERASE SOL
7262 FOR I=1 TO 10
7264 IF QTY(I)=0 THEN GOTO 7268
```

```
7265 PRINT#1," TICKET NUMBER: ";I;"  TICKET LENGTH: ";:PRINT#1,
USING "##.##";TL(I)-RKF;:PRINT #1,"   NUMBER OF PIECES:
";QTY(I):PRINT#1," "
7266 PRINT" TICKET NUMBER: ";I;"  TICKET LENGTH: ";:PRINT USING
"##.##";TL(I)-RKF;:PRINT "   NUMBER OF PIECES: ";QTY(I):PRINT
7268 NEXT I
7270 RETURN
7280 REM SUBPROGRAM TO MANUALLY UPDATE TICKET PRIORITIES
7290 PRINT "UPDATE PRIORITIES BY TICKET NUMBER AND <ENTER> TO
EXIT."
7300 PRINT "INPUT TICKET NUMBER":INPUT TN
7310 IF TN=0 THEN GOTO 7380
7320 IF TN>10 THEN CLS:BEEP:GOTO 7290
7330 PRINT "TICKET LENGTH IS: ";TL(TN);" INS"
7340 PRINT "CURRENT PRIORITY IS: ";PR(TN)
7350 PRINT "INPUT NEW PRIORITY":INPUT PR(TN)
7360 IF PR(TN)>11 THEN CLS:BEEP:PRINT"INPUT ERROR. TRY
AGAIN..":GOTO 7330
7370 CLS:GOTO 7290
7380 RETURN
7390 REM REPLACE FINISHED TICKETS
7400 SOUND 200,2:SOUND 400,4
7410 TL(RET)=TL(TF)-
RKF:W(RET)=W(TF):PC(RET)=PC(TF):CUW(RET)=CUW(TF):WP(RET)=WP(TF)
:BF(RET)=BF(TF):BFC(RET)=BFC(TF):BFC(TF)=0:RET=RET+1:PRINT"TICK
ET NUMBER";TF;"OF LENGTH ";:PRINT USING "##.##";TL(TF)-RKF;
7420 PRINT" COMPLETED. REPLACE WITH NEW TICKET."
7430 PRINT"TICKET LENGTH (INS)?":INPUT TL(TF):PRINT"TICKET
WIDTH(INS)?":INPUT W(TF):WP(TF)=W(TF)+RKF:CUW(TF)=0
7440 IF STAM(2)=1 THEN PRINT" INPUT TICKET REQUIREMENT IN BD
FT":INPUT BF(TF):GOTO 7460
7450 PRINT" INPUT TICKET REQUIREMENT AS NUMBER OF PIECES":INPUT
PC(TF):BF(TF)=TL(TF)*W(TF)*PC(TF)/144
7460 IF STAM(4)=1 THEN PRINT" INPUT PRIORITY FOR THIS
TICKET":INPUT PR(TF)
7470 IF STAM(4)=1 AND PR(TF)=11 THEN PR(TF)=14
7480 TL(TF)=TL(TF)+RKF
7490 RETURN
7500 PRINT" ":PRINT" "
7510 PRINT"-----------------------------------------------------
------------------"
7520 PRINT"l            YIELD OPTIMIZATION BASED ON TICKET
LENGTH                l"
7530 PRINT"l
==========================================                    l"
7540 PRINT"l     This program is designed to optimize yield for
a given grade    l"
7550 PRINT"l     based only on ticket lengths. The program
generates boards of   l"
7560 PRINT"l     no 1 common grade based on statistical data
and simulates the   l"
7570 PRINT"l     cutting of a set of ten tickets on these
boards.               l"
7580 PRINT"l
l"
7590 PRINT"l    The input to the program consists of:
l"
```

```
7600 PRINT"l       (i) Ticket length (Ins)
l"
7610 PRINT"l      (ii) Ticket width (Ins)
l"
7620 PRINT"l     (iii) Ticket requirement (Bd Ft)
l"
7630 PRINT"l
l"
7640 PRINT"l    The output of the program consists of:
l"
7650 PRINT"l       (i) Board and clear section dimensions
l"
7660 PRINT"l      (ii) Tickets cut on a clear section and the
quantity cut       l"
7670 PRINT"l     (iii) Yield : Individual board / Cumulative
l"
7680 PRINT"l      (iv) Volume of wood cut for each of the ten
tickets for        l"
7690 PRINT"l           the length of simulation
l"
7700 PRINT"l
l"
7710 PRINT"l    The output is also written to a disk file
A:DEMO1 for later      l"
7720 PRINT"l    printing.
l"
7730 PRINT"l                        <SPACE BAR> to continue
.........              l"
7740
PRINT"l_____________________________________________________________
_____________l"
7750 G$=INKEY$
7760 IF G$=" " THEN  CLS:RETURN
7770 GOTO 7750
7780 REM SUBPROGRAM TO PLOT TICKET STATUS
7790 MBF=0:BFCM=0:KEY OFF
7800 SCREEN 1:CLS:COLOR ,0
7810 FOR I=1 TO 10
7820 IF BF(I)>MBF THEN MBF=BF(I)
7830 IF BFC(I)>BFCM THEN BFCM=BFC(I)
7840 NEXT I
7850 IF BFCM>MBF THEN MBF=BFCM
7855 IF MBF<100 THEN MBF=INT((MBF+9.899999)/10)*10
7860 IF MBF>100 THEN MBF=INT((MBF+99.9)/100)*100
7865 H=MBF:V=(25*MBF/18):OL=V/25:YO=OL*4
7870 WINDOW (-3,0)-(13,V)
7880 LINE (0,YO)-(11,H+YO),3,B
7890 FOR I=1 TO 10
7900 LINE (I,YO)-(I,BF(I)+YO),1
7910 LINE (I,BF(I)+YO)-(I+.4,BF(I)+YO),1
7920 LINE (I+.4,BF(I)+YO)-(I+.4,YO),1
7930 NEXT I
7940 FOR I=1 TO 10
7950 LINE (I+.1,YO)-(I+.3,BFC(I)+YO),2,BF
7960 NEXT I
7970 FOR I=1 TO 9
7975 IF BF(I+1)=0 THEN GOTO 7990
```

```
7980 LINE (I+.2,BF(I)+YO)-(I+1.2,BF(I+1)+YO),1
7990 NEXT I
8000 FOR I=1 TO 9
8005 IF BFC(I+1)=0 THEN GOTO 8020
8010 LINE (I+.2,BFC(I)+YO)-(I+1.2,BFC(I+1)+YO),2
8020 NEXT I
8030 FOR I=1 TO 10
8040 LINE (I+.2,YO)-(I+.2,YO-(OL*.5)),3
8050 NEXT I
8060 FOR I=6 TO 21 STEP 5
8070 GP=(25-I)*OL
8090 LINE (-.3,GP)-(0,GP),3
8100 NEXT I
8110 LOCATE 23,10
8120 PRINT" ";
8130 FOR I=1 TO 10 STEP 2
8135 IF TL(I)>200 THEN GOTO 8150
8140 PRINT USING "##";TL(I);:PRINT"    ";
8150 NEXT I
8160 TV=0
8170 FOR I=21 TO 6 STEP -5
8180 LOCATE I,4
8200 PRINT USING "###";TV
8205 TV=TV+(MBF/4)
8210 NEXT I
8220 LOCATE 24,15:PRINT"TIC LENGTH (Ins)"
8225 LOCATE 25,2:PRINT"<SPACE> TO EXIT..."
8230 LOCATE 9,2:PRINT"B"
8240 LOCATE 10,2:PRINT"D"
8250 LOCATE 12,2:PRINT"F"
8260 LOCATE 13,2:PRINT"T"
8270 LINE (1,V-(OL/2))-(2,V-(OL/2)),1:LINE (7,V-(OL/2))-(8,V-
(OL/2)),2
8280 LOCATE 1,14:PRINT"REQT":LOCATE 1,29:PRINT"VOL CUT":LOCATE
1,1,0
8290 Q$=INKEY$
8300 IF Q$=" " THEN GOTO 8320
8310 GOTO 8290
8320 SCREEN 0:WIDTH 80:COLOR 14,9:CLS
8330 RETURN
8443 IF STAM(7)=2 THEN PRINT "    |  *   |"
8500 REM SUBPROGRAM TO DISPLAY CURRENT MENU STATUS
8502 CLS:CH=0
8503 IF STAM(1)=0 THEN GOTO 8548
8505 CH=1:PRINT:PRINT"  Current Values:"
8510   PRINT"  ~~~~~~~ ~~~~~~~":PRINT
8515 PRINT"  Ticket   Ticket   Ticket   Ticket"
8520 PRINT"  Number   Length   Width    Requirement -BD FT"
8525 PRINT"  ______   ______   ______
_________________":PRINT
8530 FOR I=1 TO 10
8533 IF TL(I)=300 THEN GOTO 8540
8534 IF I=10 THEN GOTO 8537
8535 PRINT"   ";I;"      ";:PRINT USING "##.##";TL(I)-
RKF;:PRINT"    ";:PRINT USING "##.##";W(I);:PRINT"
";:PRINT USING "###";BF(I)
```

```
8537 IF I=10 THEN PRINT"    ";I;"      ";:PRINT USING
"##.##";TL(I)-RKF;:PRINT"     ";:PRINT USING
"##.##";W(I);:PRINT"       ";:PRINT USING "###";BF(I)
8540 NEXT I
8541 PRINT:PRINT "                    <M> to modify ticket data"
8542 PRINT"_____________________________________________________"
8543 PRINT:PRINT:PRINT"               < Space > for next
page...."
8544 Q$=INKEY$
8545 IF Q$=" " THEN CLS:GOTO 8548
8546 IF Q$="M" OR Q$="m" THEN GOSUB 8800:GOTO 8500
8547 GOTO 8544
8548 IF STAM(3)=0 THEN GOTO 8577
8550 CH=1:PRINT:PRINT"   Priority Option"
8555 PRINT"   ~~~~~~~~ ~~~~~~"
8560 IF STAM(4)=1 THEN PRINT"   Manual Allocation of
Priorities"
8565 IF STAM(4)=2 THEN PRINT"   Automatic Allocation of
Priorities"
8570 IF STAM(4)=3 THEN PRINT"   No Priorities"
8575 PRINT"_____________________________________________________"
8577 IF STAM(5)=0 THEN GOTO 8607
8580 CH=1:PRINT:PRINT"   Grade(s) Selected"
8585 PRINT"   ~~~~~~~~ ~~~~~~~~"
8590 IF STAM(6)=1 AND STAM(7)=1 THEN PRINT"   Single Grade :
Number 1 Common"
8595 IF STAM(6)=1 AND STAM(7)=2 THEN PRINT"   Single Grade :
Number 2 Common"
8600 IF STAM(6)=2 THEN PRINT"   Mixed Grades : No 1 Com and No
2 Com"
8605 PRINT"_____________________________________________________"
8607 IF STAM(10)=0 THEN GOTO 8635
8610 CH=1:PRINT :PRINT"   Cutting Strategy :";
8615 IF STAM(11)=1 THEN PRINT" Cut-Off"
8620 IF STAM(11)=2 THEN PRINT" Cut-Rip"
8625   PRINT"   ~~~~~~~ ~~~~~~~~"
8630 PRINT"_____________________________________________________"
8635 IF STAM(12)=0 THEN GOTO 8660
8640 CH=1:PRINT:PRINT"   BOARD DATA :";
8645 IF STAM(13)=1 THEN PRINT"  Manual Input"
8646 IF STAM(13)=2 THEN PRINT"  By Simulation"
8650 PRINT"   ~~~~~ ~~~~"
8655 PRINT"_____________________________________________________"
8660 IF STAM(14)=0 THEN GOTO 8685
8665 CH=1:PRINT :PRINT"   Graph Display :";
8670 IF STAM(15)=1 THEN PRINT"  On"
8675 IF STAM(15)=2 THEN PRINT"  Off"
8677        PRINT"   ~~~~~ ~~~~~~~"
8680 PRINT"_____________________________________________________"
8685 IF CH=0 THEN BEEP:RETURN
8688 PRINT"     < space > to return to main menu..."
8690 Q$=INKEY$
8695 IF Q$=" " THEN RETURN
8700 GOTO 8690
8800 REM SUBPROGRAM TO MODIFY TICKET DATA
8805 CLS:PRINT"  Modify Ticket Data by Ticket Number; <ENTER>
to exit..."
```

```
8810 INPUT NT
8815 IF NT=0 THEN RETURN
8820 PRINT" Ticket #";NT
8825 PRINT" Length : ";TL(NT)-RKF;" Ins"
8830 PRINT"  Width : ";W(NT);" Ins"
8840 PRINT"   Reqt : ";BF(NT);" Bd Ft"
8845 PRINT"Input Ticket Length (Ins) for Ticket #";NT;" ***
VALID RANGE 16 TO 85 ***":INPUT TL(NT):TL(NT)=TL(NT)+RKF
8850 IF TL(NT)-RKF<16 OR TL(NT)-RKF>85 THEN CLS:BEEP:PRINT
"INPUT ERROR. TRY AGAIN..":GOTO 8845
8855 PRINT"Input Ticket Width (Ins) for Ticket #";NT;"*** VALID
RANGE 1 TO 48 ***":INPUT W(NT):WP(NT)=W(NT)+RKF
8860 IF W(NT)<=0 THEN CLS:BEEP:PRINT "INPUT ERROR. TRY
AGAIN..":GOTO 8855
8865 IF STAM(2)=1 THEN GOTO 8885
8870 IF STAM(2)=2 THEN PRINT"INPUT NO OF PIECES
REQUIRED.":INPUT PC(NT):BF(NT)=(TL(NT)-RKF)*W(NT)*PC(NT)/144
8875 IF PC(NT)<=0 THEN CLS:BEEP:PRINT"INPUT ERROR. TRY
AGAIN..":GOTO 8870
8880 GOTO 8805
8885 PRINT"Input Ticket Requirement (Bd Ft) for Ticket
#";NT;"*** MAX 900 BD FT ***":INPUT BF(NT)
8890 IF BF(NT)<=0 OR BF(NT)>900 THEN CLS:BEEP:PRINT"INPUT
ERROR. TRY AGAIN..":GOTO 8885
8895 GOTO 8805
9000 REM SUBPROGRAM TO RESET VARIABLES FOR RERUNS
9005 RIND=0:STAM(17)=0:AAO=0:J=0:BFSEC=0:NSEC=0:YB=0:YC=0:A1=0
9010 BFT=0:CC1=0:CC2=0:AC1=0:AC2=0:SKIP=0:RET=12
9015 FOR I=1 TO 30
9020 BFC(I)=0:CUW(I)=0
9025 NEXT I
9030 ERASE I,F1,F2,F3,F4,F5,F6,F7,V1,V2,V3,V4,V5,V6,V7
9045 CLOSE #1:OPEN "o",#1,"A:output"
9050 RESTORE
9055 RETURN
9100 REM SUBPROGRAM TO PRINT TITLE PAGES
9102 CLS
9105 LOCATE 6,1
9110 PRINT"
                                    "
 ___________________________________
9115 PRINT"                          |
|"
9120 PRINT"                          |            OPTYLD
|"                                                 .
9125 PRINT"                          |            ~~~~~~
|"
9130 PRINT"                          |
|"
9132 PRINT"                          |
|"
9135 PRINT"                          |          C I M  LAB
|"
9137 PRINT"                          |
|"
9140 PRINT"                          |
|"
```

```
9145 PRINT"                          |     UNIVERSITY OF MARYLAND
|"
9150 PRINT"                          |
|"
9155 PRINT"                          |    COLLEGE PARK   MD 20742
|"
9160 PRINT"
|_____________________________________|"
9165 FOR I=1 TO 8000
9170 NEXT I
9175 CLS
9180 PRINT
9185 PRINT"                       Y I E L D   O P T I M I Z A T
I O N"
9190 PRINT
9195 PRINT"   This program is designed to optimize yield at the
cut-off saw. The   "
9197 PRINT
9200 PRINT"   process variables such as ticket data, grade(s)
to be used, etc; are "
9202 PRINT
9205 PRINT"   specified by the user. All user input is made
through a main menu. At"
9207 PRINT
9210 PRINT"   each stage, the user is prompted for the required
information."
9212 PRINT
9215 PRINT"   The choices made by the user are retained after a
run; so that,      "
9217 PRINT
9220 PRINT"   reruns can be made without reentering data. The
program output is    "
9222 PRINT
9225 PRINT"   also written to a discfile a:output. This file is
OVERWRITTEN every  "
9227 PRINT
9230 PRINT"   time a rerun is made."
9235 GOSUB 9600
9260 PRINT:PRINT"                                M E N U   O P T I
O N S"
9265 PRINT
9270 PRINT"   T I C K E T   D A T A"
9275 PRINT"   The program can accept a maximum of ten tickets.
User input includes "
9280 PRINT"   ticket length, width, number of pieces or total
board feet required. "
9285 PRINT
9290 PRINT"   P R I O R I T I E S"
9295 PRINT"   The priority scale ranges from zero to eleven.
Zero indicates no     "
9300 PRINT"   priority while eleven indicates topmost priority.
Numbers one to ten "
9305 PRINT"   indicate increasing priority. If manual update of
priorities is      "
9310 PRINT"   chosen, the user is prompted to input revised
priorities after each  "
```

```
9315 PRINT"   board. Otherwise, with option B, priorities are
automatically updated"
9320 PRINT"   to ensure a good match between production and
requirement."
9325 PRINT
9330 PRINT"   B O A R D  D A T A  I N P U T"
9335 PRINT"   The options available are to manually input the
board dimensions one "
9340 PRINT"   at a time and to generate the board dimensions by
simulation. With the"
9345 PRINT"   first option, the user is prompted for the
required inputs. With the "
9350 PRINT"   second option, board dimensions are generated by
Monte Carlo simulation"
9355 PRINT"   technique using statistical data available for
the chosen grade."
9360 GOSUB 9600
9450 PRINT
9455 PRINT"   G R A P H  O P T I O N"
9460 PRINT"   When this option is selected, the program pauses
after each board to "
9465 PRINT"   display a graph of production vs requirement."
9600 PRINT :PRINT :PRINT "                                   <E> to
exit \ < SPACE > to continue..."
9602 Q$=""
9605 Q$=INKEY$
9610 IF Q$=" " THEN CLS:RETURN
9612 IF Q$="E" OR Q$="e" THEN Q$="":GOTO 20
9615 GOTO 9605
9700 CLS
9705 PRINT"Run Time Error, Run Aborted, Please Wait...."
9710 FOR I=1 TO 4000
9715 NEXT I
9720 CLS
9725 CLEAR
9730 GOSUB 9000
9735 RESUME 60
```

Listing of program 2comdat.bas:

```
6000 REM SUBPROGRAM TO SET 2COMMON GRADE DATA
6002 CLS:PRINT"READING 2 COMMON GRADE DATA..."
6010 DEFINT I,J,H,K
6020 IF RIND>0 THEN RIND=0:GOTO 6050
6030 IF STAM(6)=1 AND STAM(7)=2 THEN GOTO 6060
6040 IF STAM(6)=2 AND GR=2 AND BFT1=0 THEN GOTO 6060
6050 ERASE I,F1,F2,F3,F4,F5,F6,F7,V1,V2,V3,V4,V5,V6,V7
6060 DIM
I(7),F1(15),V1(15),F2(27),V2(27),F3(7),V3(7),F4(40),V4(40)
,F5(40),V5(40),F6(58),V6(58),F7(17),V7(17)
6070 RANDOMIZE TIMER
6080 FOR C=1 TO 7
6090 READ I(C)
6100 NEXT C
6110 FOR C=1 TO 15
```

```
6120 READ F1(C)
6130 NEXT C
6140 FOR C=1 TO 15
6150 READ V1(C)
6160 NEXT C
6170 FOR C=1 TO 27
6180 READ F2(C)
6190 NEXT C
6200 FOR C=1 TO 27
6210 READ V2(C)
6220 NEXT C
6230 FOR C=1 TO 7
6240 READ F3(C)
6250 NEXT C
6260 FOR C=1 TO 7
6270 READ V3(C)
6280 NEXT C
6290 FOR C=1 TO 40
6300 READ F4(C)
6310 NEXT C
6320 FOR C=1 TO 40
6330 READ V4(C)
6340 NEXT C
6350 FOR C=1 TO 40
6360 READ F5(C)
6370 NEXT C
6380 FOR C=1 TO 40
6390 READ V5(C)
6400 NEXT C
6410 FOR C=1 TO 58
6420 READ F6(C)
6430 NEXT C
6440 FOR C=1 TO 58
6450 READ V6(C)
6460 NEXT C
6470 FOR C=1 TO 17
6480 READ F7(C)
6490 NEXT C
6500 FOR C=1 TO 17
6510 READ V7(C)
6520 NEXT C
6530 DATA 104,104,104,426,427,428,427
6540 DATA
66.25,71.25,73.75,76.25,81.25,83.75,86.25,88.75,91.25,93.7
5,96.25,98.75,101.25,103.75,106.25
6550 DATA 1,6,8,11,14,20,26,30,33,39,50,72,91,98,104
6560 DATA
3.65,4.05,4.25,4.45,4.65,5.05,5.25,5.45,5.65,6.05,6.25,6.4
5,6.65,7.05,7.25,7.45,7.65,8.05,8.25,9.05,9.25,9.45,9.65,1
0.25,10.45,12.25,15.25
6570 DATA
1,5,8,9,12,29,37,42,44,50,58,70,73,76,80,81,84,89,90,92,93
,94,97,99,102,103,104
6580 DATA 1,2,3,4,5,6,7
6590 DATA 5,18,35,66,78,98,104
6600 DATA
2.5,5,7.5,10,12.5,15,17.5,20,22.5,25,27.5,30,32.5,35,37.5,
```

```
40,42.5,45,47.5,50,52.5,55,57.5,60,62.5,65,67.5,70,72.5,75
,77.5,80,82.5,85,87.5,90,92.5,95,97.5,99.3
6610 DATA
13,31,44,54,76,85,90,98,109,124,138,147,157,167,174,186,19
5,207,215,226,235,245,255,261,267,284,293,298,305,307,313,
325,335,346,354,373,389,412,424,426
6620 DATA
2.5,5,7.5,10,12.5,15,17.5,20,22.5,25,27.5,30,32.5,35,37.5,
40,42.5,45,47.5,50,52.5,55,57.5,60,62.5,65,67.5,70,72.5,75
,77.5,80,82.5,85,87.5,90,92.5,95,97.5,100
6630 DATA
2,17,37,61,81,89,91,102,105,117,120,130,144,150,162,172,17
6,185,197,210,213,224,235,245,257,272,281,298,303,313,322,
337,345,355,369,386,402,419,425,427
6640 DATA
0.25,0.5,0.75,1,1.25,1.5,1.75,2,2.25,2.5,2.75,3,3.25,3.5,3
.75,4,4.25,4.5,5,5.25,6,6.5,7,8,8.5,9,9.5,10,11,11.5,12,13
,14,14.5,15,16,17,18,18.5,19,19.5,20,21,22,23,24,25,25.5,2
6,27,28,29,30,32,33,35,36,40
6650 DATA
3,53,61,132,143,177,181,214,221,231,233,256,257,259,260,27
1,272,275,287,288,295,297,307,319,321,323,324,342,348,350,
359,363,366,367,376,381,386,387,388,389,390,396,398,401,40
5,409,410,411,413,414,416,417,422,423,424,425,427,428
6660 DATA
0.25,0.5,0.75,1,1.25,1.5,1.75,2,2.25,2.5,2.75,3,3.25,3.5,4
,4.25,4.5
6670 DATA
11,114,149,291,304,350,354,387,388,397,398,417,419,423,425
,426,427
6680 IF STAM(6)=1 AND STAM(7)=2 THEN
CHAIN"A:optyld",630,ALL
6690 IF STAM(6)=2 AND SKIP=1 THEN CHAIN "A:optyld",1760,ALL
```