

Cell Loading and Scheduling in a Shoe Manufacturing Company

A thesis presented to
the faculty of
the Fritz J. and Dolores H Russ
College of Engineering and Technology of
Ohio University

In partial fulfillment
of the requirements for the degree
Master of Science

Ananthanarayanan K. Subramanian

November 2004

This thesis entitled
CELL LOADING AND SCHEDULING IN A SHOE MANUFACTURING COMPANY

by
ANANTHANARAYANAN K. SUBRAMANIAN

has been approved
for the Department of Industrial and Manufacturing Systems Engineering
and the Russ College of Engineering and Technology by

Gürsel A. Süer
Professor of Industrial and Manufacturing Systems Engineering

Dennis Irwin
Dean, Fritz J. and Dolores H. Russ
College of Engineering and Technology

SUBRAMANIAN, ANANTHANARAYANAN. K. MS. November 2004. Industrial and Manufacturing Systems Engineering.

Cell Loading and Scheduling in a Shoe Manufacturing Company (130 pp.)

Director of Thesis: Gürsel A. Süer

This thesis focuses on a complex cell loading and scheduling problem at a shoe manufacturing facility. A set of four cell loading heuristics, namely – Do Not Split, Split as and when required, and Split Only When Load Greater than MaxCap, were developed and augmented with a simple cell scheduling methodology. The objective of the heuristic procedures is to ensure that all the jobs are loaded without exceeding the given capacity. The heuristics can also be used to minimize makespan by carefully defining certain parameters. The heuristics are encoded using VB 6.0 and a software application was created to test and compare the heuristics. These heuristics are compared with respect to completed jobs, makespan and setup times. Overall, after the cell loading stage, the Split as and when required heuristic can be recommended as the best. But the optimal result from the cell loading stage might not be so at the end of the cell scheduling stage. The choice of heuristics is also driven by the conditions that the user is faced with like overtime costs, machine breakdowns etc. Hence the heuristics have to be repeatedly applied after varying inputs until an optimal result appears at the end of the scheduling stage. However this is not a tedious process given the processing speed of the software application.

Approved:

Gürsel A. Süer

Professor of Industrial and Manufacturing Systems Engineering

Dedication

This thesis is dedicated to the people who have supported me all the way in my life - my mother Vasantha, my father Subramanian, my sister Preethi, my teachers, my advisor Dr. Gürsel A. Sürer, my committee members and all my friends from high school and college in India and in the USA.

Table of Contents

| | |
|--|-----------|
| CHAPTER 1. INTRODUCTION | 13 |
| 1.1 Group Technology | 13 |
| 1.2 Manufacturing Systems | 14 |
| 1.2.1 Dedicated Production Line | 15 |
| 1.2.2 Job Shop | 16 |
| 1.2.3 Manufacturing Cells | 16 |
| 1.2.4 Fixed Layouts..... | 16 |
| 1.3 Cellular Manufacturing Systems (CMS) | 17 |
| 1.4 Cellular Manufacturing System Configurations: | 18 |
| 1.4.1 Independent Cells..... | 18 |
| 1.4.2 Connected Cells | 19 |
| 1.5 Production Control in Cellular Manufacturing Systems | 22 |
| 1.5.1 Cell Loading..... | 22 |
| 1.5.2 Cell Scheduling | 23 |
| 1.6 Solution Techniques | 24 |
| 1.6.1 Optimizing Procedures | 25 |
| 1.6.2 Heuristics | 25 |
| 1.6.3 Meta-Heuristics | 25 |
| 1.7 Computer Programming Language – Visual Basic..... | 26 |
| 1.8 Thesis Objectives | 27 |
| 1.9 Organization of This Thesis | 28 |
| CHAPTER 2. LITERATURE REVIEW..... | 29 |
| CHAPTER 3. PROBLEM STATEMENT | 40 |
| 3.1 A Typical Cell group..... | 40 |
| 3.2 Current Cellular Configuration | 40 |

| | |
|---|------------|
| | 6 |
| 3.3 The Jobs..... | 42 |
| 3.4 The Rotary Machine Cell | 42 |
| 3.5 Rotary machine operation – Loading Shoes..... | 43 |
| 3.6 Molds..... | 48 |
| 3.7 Processing Times | 49 |
| 3.8 Special Considerations in the Cell loading Phase..... | 51 |
| 3.8.1 The Loading Procedure | 51 |
| 3.8.2 MaxCap | 52 |
| 3.8.3 The Shared Cell..... | 53 |
| CHAPTER 4. METHODOLOGY | 54 |
| 4.1 Phase I-A: Demand Aggregation and Number of Cells Determination..... | 54 |
| 4.1.1 Data Input | 56 |
| 4.1.2 Grouping of Products in Super-families | 56 |
| 4.1.3 Calculations of Total Processing Times | 58 |
| 4.1.4 Calculation of the number of cells of each type..... | 61 |
| 4.2 Phase I-B – Family Definition | 63 |
| 4.2.1 Calculation of family processing times | 67 |
| 4.2.2 Arranging Families in LPT | 67 |
| 4.3 Phase II – Cell Loading | 69 |
| 4.3.1 Do not Split Families:..... | 70 |
| 4.3.2 Lot splitting allowed as and when required | 76 |
| 4.3.3 Split families only when load is greater than MaxCap | 83 |
| 4.3.4 Load Jobs First Into the Cells and Then Form Families Independently | 90 |
| 4.4 Scheduling of Families and Jobs..... | 96 |
| 4.5 The Cell Loading And Cell Scheduling (CLACS) Application | 102 |
| CHAPTER 5. ANALYSIS OF RESULTS..... | 109 |
| 5.1 Experimental Conditions | 109 |
| 5.2 Comparison of cell loading heuristics..... | 111 |

| | |
|---|------------|
| | 7 |
| 5.3 Effects of MaxCap on Makespan (lower bound)..... | 113 |
| 5.4 Effects of MaxCap on Makespan (upper bound)..... | 115 |
| 5.5 Comparison of cell idle times | 116 |
| 5.6 Comparison of Makespan (number of families) | 120 |
| CHAPTER 6. CONCLUSIONS AND FUTURE WORK..... | 122 |
| 6.1 Conclusions..... | 122 |
| 6.2 Future Work..... | 124 |
| BIBLIOGRAPHY..... | 126 |

List of Tables

| | |
|--|-----|
| Table 3-1: Sample list of jobs | 43 |
| Table 3-2: Processing Times | 50 |
| Table 4-1: Typical data | 57 |
| Table 4-2: Jobs divided into super-families (Sample) | 59 |
| Table 4-3: Calculation of processing times | 61 |
| Table 4-4: Typical examples of families | 65 |
| Table 4-5: Sample calculation for family processing time | 67 |
| Table 4-6: Typical family data arranged in LPT | 68 |
| Table 4-7: Family FMPN-A | 79 |
| Table 4-8: Families after split | 80 |
| Table 4-9: Another Example | 85 |
| Table 4-10: Family FFPN-E | 99 |
| Table 4-11: The features of CLACS | 103 |
| Table 5-1: Experimental Conditions | 109 |
| Table 5-2: Detailed Experimental Conditions | 110 |
| Table 5-3: Makespan Values for Different Strategies (Pre-Schedule) | 111 |
| Table 5-4: Effect of increase in MaxCap on Makespan (Problem Small3) | 114 |
| Table 5-5: Effect of increase in MaxCap on Makespan (Problem Big1) | 114 |
| Table 5-6: Makespan Values for Different Strategies (Post-Schedule) | 115 |
| Table 5-7: Effect of MaxCap (upper bound) | 116 |
| Table 5-8: Idle Times for Problem Big1 @ MaxCap = 1900 | 117 |
| Table 5-9: Idle Times for Problem Big2 (MaxCap = 2025) | 118 |

| | |
|--|-----|
| Table 5-10: Idle Times for Problem Medium3 (MaxCap = 1900)..... | 119 |
| Table 5-11: Makespan differences at different nf values (MaxCap = 2000)..... | 121 |

List of Figures

| | |
|--|----|
| Figure 1-1: Classification of Manufacturing Systems | 14 |
| Figure 1-2: Independent Manufacturing Cells | 20 |
| Figure 1-3: Serially Connected Cells..... | 21 |
| Figure 1-4: Serial-Parallel Connected Cells (with alternate routes)..... | 21 |
| Figure 1-5: Serial-Parallel Connected Cells (No alternate routes)..... | 22 |
| Figure 3-1: The cell groups at the plant..... | 41 |
| Figure 3-2: Size Distribution and Production volumes (Male shoes) – A Sample | 42 |
| Figure 3-3: Initial Position of the Rotary Injection Molding Machine | 44 |
| Figure 3-4: First pair of shoes loaded at $t = 0$ seconds | 44 |
| Figure 3-5: Loading the second pair at $t = 18$ seconds | 45 |
| Figure 3-6: Loading the third pair of empty positions at $t = 36$ seconds..... | 45 |
| Figure 3-7: Position of the Rotary Machine at $t = 54$ seconds..... | 46 |
| Figure 3-8: Position of the Rotary Machine at $t = 72$ seconds..... | 47 |
| Figure 3-9: Position of the Rotary Machine at $t = 90$ seconds..... | 47 |
| Figure 3-10: Position of the Rotary Machine at $t = 108$ seconds..... | 48 |
| Figure 4-1: Overall Methodology..... | 55 |
| Figure 4-2: Grouping Jobs into Super-families..... | 60 |
| Figure 4-3: Product Structure..... | 64 |
| Figure 4-4: Family Definition | 66 |
| Figure 4-5: Family Loading Decisions | 70 |
| Figure 4-6: Do not Split Families..... | 71 |
| Figure 4-7: Gantt Chart at $T = 0$ minutes..... | 72 |

| | |
|--|-----|
| | 11 |
| Figure 4-8: Final Gantt Chart - Do not Split Families..... | 75 |
| Figure 4-9: Gantt Chart at Tnow = 1214 minutes | 77 |
| Figure 4-10: Split Families as and when required..... | 78 |
| Figure 4-11: Gantt Chart after splitting and loading new family..... | 80 |
| Figure 4-12: Final Gantt Chart - Split As and When Required | 82 |
| Figure 4-13: Split Families only if load > MaxCap | 84 |
| Figure 4-14: MaxCap Gantt Chart at Tnow = 0..... | 87 |
| Figure 4-15: Final Gantt Chart - Split Families only if load > MaxCap | 89 |
| Figure 4-16: Job Loading..... | 91 |
| Figure 4-17: Gantt Chart - Load Jobs: Stage I..... | 93 |
| Figure 4-18: Final Gantt Chart - Load Jobs: Stage II (After Re-ordering)..... | 94 |
| Figure 4-19: Job Re-grouping after Loading | 95 |
| Figure 4-20: Gantt Chart (Do Not Split Heuristic) after scheduling phase | 101 |
| Figure 4-21: CLACS - Front Door form..... | 104 |
| Figure 4-22: CLACS - Manual Data Entry form | 104 |
| Figure 4-23: CLACS - Calculate Load/Cells | 105 |
| Figure 4-24: CLACS - Form Family | 106 |
| Figure 4-25: CLACS - Cell Loading | 107 |
| Figure 4-26: CLACS - No of molds / Scheduling..... | 107 |
| Figure 5-1: Makespan for Problem Big1 @ MaxCap = 2000..... | 112 |
| Figure 5-2: Makespan Trends for Problem Medium1 @ MaxCap = 1400 | 112 |
| Figure 5-3: Makespan Trends for Problem Small3 @ MaxCap = 1650..... | 113 |
| Figure 5-4: Idle Time Trends for Problem Big1 (MaxCap = 1900)..... | 117 |

| | |
|---|-----|
| | 12 |
| Figure 5-5: Idle Time Trends for Problem Big2 (MaxCap = 2025)..... | 118 |
| Figure 5-6: Idle Time Trends for Problem Medium 3 (MaxCap = 1900) | 119 |
| Figure 5-7: Effect of number of families (nf) on Makespan (MaxCap = 2000) | 120 |

CHAPTER 1. INTRODUCTION

The boundaries of manufacturing systems have slowly widened from the assembly lines popularized by the likes of Henry Ford to include the more recent agile manufacturing and intelligent manufacturing systems. But these systems have disadvantages that sometimes make them unpopular because the prerequisites for such systems cannot be achieved with low budgets or with the expertise that is available to every company. Cellular Manufacturing Systems (CMS) is a system that is perhaps anomalous to this trend. This chapter will highlight the features of a CMS and its various driving forces and controls.

1.1 Group Technology

The main driving force behind cellular manufacturing system is the concept of Group Technology. Group technology (GT), as the term indicates brings similar things together. GT is a method that improves manufacturing efficiency by classifying similar products into families [2]. Thus the key concept of GT is to plan a total division of parts or products into groups and families, based on common features shared by all the products considered for induction into the group. These features share similar production processes or techniques. For example, if jobs require shafts of different diameters or some of the jobs require bores of different diameters, these jobs can be grouped together based on these features. Another idea would be to group parts that would require injection molding in one family and so on. The main idea behind such a division is to bring similar products/parts together so that costs resulting out of non-value added

processes could be reduced. These costs mainly include costs due to setups, inventory and material handling [4].

1.2 Manufacturing Systems

Manufacturing systems could be classified in terms of the relationship between the volume of production and also the number of parts in the system. This is illustrated by Figure 1-1. As the number of unique parts increases and the production volume decreases, the manufacturing system changes from being a dedicated production line (i.e., a product layout) to a cellular manufacturing system and then to a job shop (i.e., a process layout). In addition to these three, there is also one more type of layout that is known as a fixed layout. These systems will be discussed below.

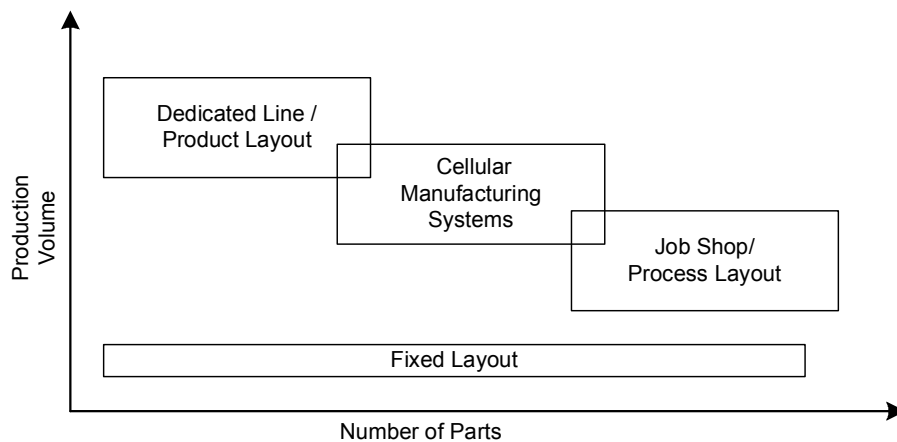


Figure 1-1: Classification of Manufacturing Systems

As Figure 1-1 indicates, there is an overlap in the definition of these systems. This overlap results because of the difference in investments possible in different cases and the

definition of such systems as desired by the management of the particular plant. For example, though our classification places a cellular manufacturing system between a job shop and a production line, Burbidge [4] mentions that a group layout (i.e., a cellular manufacturing system) has been observed to work quite effectively in a shipyard making supertankers (oil tankers which usually displace more than 100,000 tonnes) by the block welding method. So while the planners of a particular plant might decide to go for a production line for a product, another might choose to group this product with others of the same type and form production cells. However, in most cases, it is not difficult to make a choice about the type of system to be used.

1.2.1 Dedicated Production Line

As the name indicates, a dedicated production line processes just one type of product where there is little or no variation in the processes used. A dedicated line is preferred when there is high demand and very low variation in the processes for different versions of the product. An example of such a situation would be in the case of production lines manufacturing engine blocks for automobiles. In such dedicated production lines, the grouping of workers and stations are designed specifically for the product being produced. This layout would have to be ideally redesigned at the end of the product's life cycle. However, this problem is minimized usually by using standard machines as much as possible.

1.2.2 Job Shop

A job shop is used when the demand is very low, but the number of unique products is very high. An example of such a system would be in the case of the boat/yacht building industry, where the uniqueness of every boat is its USP (i.e., its Unique Selling Point). Hence, more often than not, a design is not repeated and a far greater number of unique designs are used in this industry than any other. In such a job shop (also termed as process layout), machines are grouped, based on the processes they are used for. Hence, a process layout would group milling machines in one section while drilling machines in another. While this is a simplistic view, this definition holds true.

1.2.3 Manufacturing Cells

A cellular manufacturing system is preferred when the production volume is moderate, as is the product variety. There is little variation in the processes required. More about cellular manufacturing systems will be discussed later on in this thesis.

1.2.4 Fixed Layouts

In addition to these layout types described above, one more layout type common in certain industries, is called the fixed position layout. This type of layout is common in industries where the mobility of the product being processed presents a major challenge. This is common in the ship building industry (shipyards) and aircraft industry (passenger aircraft) where movement of half built ships and aircraft pose a challenge. In such cases, it might be easier to move the machines around. The product is stationary and the machines are moved as and when required.

Most of the systems described above incorporate one or more features to enable flexibility. This flexibility is desired because current trends in manufacturing have pointed towards lower volumes and greater variety. Such manufacturing systems are termed as Flexible Manufacturing systems. This is because these systems can handle a large variety of related products. An improvement of the flexible manufacturing concept is agile manufacturing. Agile manufacturing systems are flexible manufacturing systems that can be designed for rapid response to sudden unforeseen changes in demands and loads and maintain profitability by incorporating techniques gleaned from various other philosophies such as Total Quality Management and Just-in-Time.

1.3 Cellular Manufacturing Systems (CMS)

The basic definition of a cellular manufacturing system (CMS) is that it is a manufacturing system that comprises of individual machine groups that are considered as separate entities by themselves. A CMS may be defined as a manufacturing environment implemented with the use of Group Technology (GT) principle, which in itself is characterized by the use of a group layout, short cycle flow control and a planned loading technique [4]. GT, in one of the noted definitions listed by Ham [6], is termed as a realization that many problems are similar and, by grouping similar problems, considerable time and effort can be saved by developing a single solution to a set of problems.

CMS are production systems comprising of individual machine cells that are considered as separate entities by themselves. The key concepts of GT that were discussed in previous sections in this chapter are used to form cells. The cellular

manufacturing system is perhaps the most complex system to design, because of the inherent judgments involved in deciding machine groupings and product families, which are critical in determining the system's performance. Such a system would be preferred in an industry such as the footwear industry, which is the subject of this thesis. Each order for footwear contains a large number of shoes of different designs, with the sizes also varying. However, the processes required for each design do not vary significantly. Hence, a cellular manufacturing system seems ideal for such a scenario.

The benefits of cellular manufacturing are many and much discussed in literature. As stated in many works on this subject, these benefits include reduced set-up times, lower work-in-process levels, faster throughput times, improved product quality and reduced material handling [12].

1.4 Cellular Manufacturing System Configurations:

Cells are ideally designed in such a way that raw materials enter the cell and exit as completed product. This factor determines the classification of a cell as Independent or Connected.

1.4.1 Independent Cells

If all the operations required by a family can be performed in one cell alone, then that cell is termed as an Independent Cell (Figure 1-2). The raw materials arrive at the cell at one end and the finished product exits at the other end without being transferred to any other cell for processing. Hence, assigning the product only impacts the cell to which it is originally loaded [18]. Alternatively, a Connected Cell is one that is used in

conjunction with other cells to perform all the operations that a particular family requires. From the product family point of view, an Independent Cell can either be allocated to one family alone or may be shared between two or more families. This factor brings another sub-classification of Independent Cells – Dedicated Cells and Shared Cells.

In most cases, it might be prudent to utilize the unused capacity of other cells, especially when the machines that are underutilized are costly and if these machines are capable of processing families that are usually processed in other cells. This determines whether the cell is a Dedicated Cell or not. So a Dedicated Cell will have individual families entering it and getting out after all the operations possible on it are performed (as in Cells 1, 2 and 3 in Figure 1-2). In such cases, each cell is dedicated to just one particular family and hence, the name. However, sometimes, underutilized capacity, processing requirements or economic reasons (machine cost) result in cells shared by one or more product families. These cells are called Shared Cells (see Cell 4 in Figure 1-2). In such Shared Cells, more than one family is processed in a cell (in this case, Families 1 and 2). But after all this, sometimes there might parts that are unique from others to such a great extent that they cannot be included in any particular family. In such cases, a separate cell called a Remainder Cell is used for the operations required by all these parts (Cell 5 in Figure 1-2).

1.4.2 Connected Cells

As opposed to Independent Cells, Connected Cells are used in conjunction with each other. Two cells are termed “connected” when the output from one cell becomes the input for the other, i.e., it takes at least two cells to complete a product [14]. The

product can be completed only after it passes through all or some of these cells. In such cases, the effect of loading decisions cascades through the system of Connected Cells and any loading decision should take into consideration the loads in all of the affected cells together.

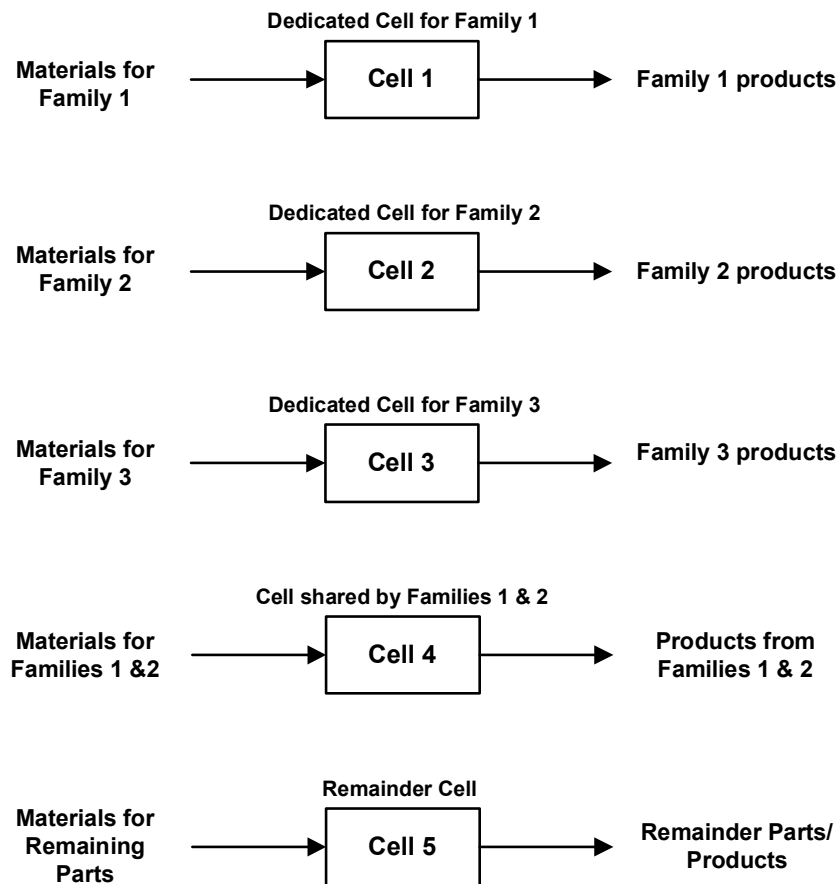


Figure 1-2: Independent Manufacturing Cells

Connected Cells can be classified as Serial Connected Cells or Serial-Parallel Connected Cells [16]. Serial cells (Figure 1-3) are cells where the product is processed in every cell sequentially, though in cases, cell skipping may be the case. Hence, load

determination is much simpler and straightforward. In Serial-Parallel cells, the families might follow different paths depending on the operations required and whether alternate routes exist (Cells A1 and A2 or Cells C1 and C2 in Figure 1-4) or not (Figure 1-5) and the fact that some cells might serve as Shared Cells while some cells may be dedicated complicates the loading process.

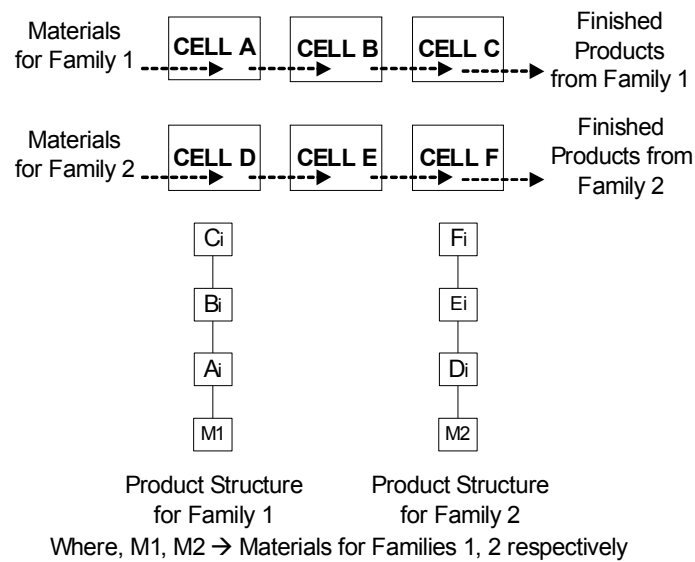


Figure 1-3: Serially Connected Cells

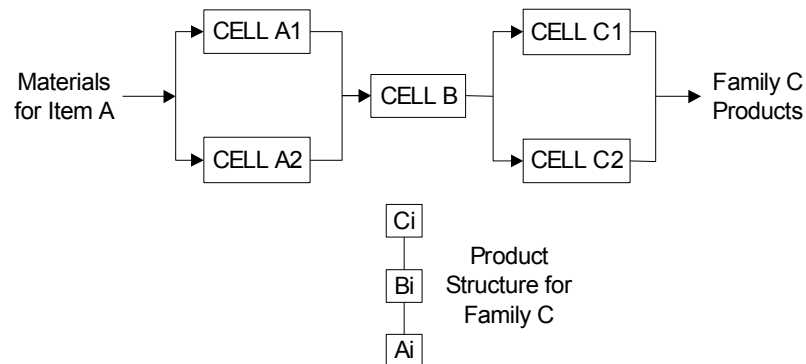


Figure 1-4: Serial-Parallel Connected Cells (with alternate routes)

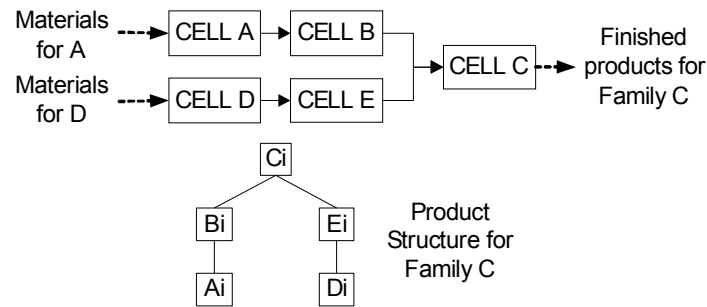


Figure 1-5: Serial-Parallel Connected Cells (No alternate routes)

1.5 Production Control in Cellular Manufacturing Systems

A system cannot work well in the absence of a control system and this holds true in case of manufacturing systems too. Thus, any good implementation of cellular manufacturing requires a planning and control system to take the implementation to the exemplary level. Cell loading and cell scheduling are the two key activities of special interest in cellular manufacturing in connection with how to control the system [18].

1.5.1 Cell Loading

In any manufacturing system, individual cells have to be individually studied and unique planning procedures have to be considered for every single one of them, depending on their configurations and other relevant parameters such as the number of machines, the type of products, the process flow type and the volume of products. Thus, the control systems are critical in the operation of cellular manufacturing systems.

Cell loading is a production planning activity that results in the assignment of products to the cells. Thus, it determines the kind of items and the quantities to be

produced in each cell subject to the production capacity and the demand forecast. Hence, cell loading involves three tasks [18].

1. Selecting the product,
2. Selecting a cell, and
3. Finding the order of the products assigned to each cell in the system.

The first two steps can be done in any order. If a product has to be chosen first followed by a feasible cell, then this process is termed the “product priority” approach. If the cell is chosen first and then feasible products are chosen, the approach is termed “cell priority” approach.

The overall objectives that cell loading aims at are minimization of WIP inventory, lateness, setups and also the maximization of cell utilization and the balancing of the load among cells [14]. However, cell loading is quite a complex process and in real life environments, it is difficult to simultaneously allocate resources and schedule them too. Hence, the absence of results (ideally without making concessions and assumptions) make cell loading literature a scarce commodity in contemporary technical journals.

1.5.2 Cell Scheduling

In any cellular manufacturing system, not only have the part families for each cell to be decided, but the time frames (i.e., the start and the completion times) for manufacturing the product families and individual products on individual machines or operators have to be determined. Hence, this process is different from loading and is termed cell scheduling. Thus, scheduling is a process of allocation of resources over time

to perform a collection of tasks [3]. In this case, the resources are the manufacturing cells (with the machines in them and the respective operators) and these are allocated over time among different product families and the individual products. Hence, cell scheduling is a process of determining start and completion times on different machines once the product has been assigned to a cell [18].

Scheduling of manufacturing cells depend on the cell configuration and several other details like product due dates, ready times and the processing times of the products on different machines and can be performed with a view of optimizing one or more performance measure.

1.6 Solution Techniques

If the cells are connected serially (Figure 1-3), then the scheduling is very simple because since each part in the family travels through each of the cells in the same order, this system can be considered as a single machine and the single machine scheduling heuristic can be applied accordingly, based on the performance measure. The problem is usually complicated by the presence of alternate routings in hybrid cells (like in Figure 1-4). In such cases there are no ready to use scheduling heuristics and each problem has to be treated uniquely.

Hence, different solution techniques or procedures have to be adopted usually for solving real life manufacturing loading or scheduling problems because most of these systems are much complicated than the system shown in Figure 1-3. These solutions broadly fall into three categories:

1. Optimizing Procedures,

2. Heuristics and
3. Meta-Heuristics.

1.6.1 Optimizing Procedures

Optimizing procedures are usually procedures that are applied to problems that are narrowly defined and specific in nature. Optimizing procedures usually involve much more computational time than any of the other two categories. However, these procedures guarantee an optimal position. Examples of such optimizing procedures are Dynamic Programming and Integer Programming.

1.6.2 Heuristics

Heuristic loading or scheduling procedures are usually used when the problem definition is broad enough that just a feasible solution is acceptable. These heuristic procedures hence, do not guarantee optimal solutions. Heuristics vary usually on a case to case basis and the successful application of a heuristic to one problem does not guarantee the same results for other problems or for a different performance measure in the same problem. An example of a heuristic procedure would be a dynamic dispatching rule such as the Critical Ratio.

1.6.3 Meta-Heuristics

Meta-Heuristics are procedures derived from artificial intelligence principles to arrive upon feasible solutions for manufacturing problems from a wider search space than the previously mentioned categories. These meta-heuristics are usually based on a set of

rigid steps that are followed regardless of the nature of the problem or the performance measure. These procedures have a mixed record in arriving upon optimal solutions, but sometimes provide feasible solutions for problems that are deemed un-solvable when heuristics or optimizing procedures are used. Some examples of meta-heuristics are genetic algorithms, artificial neural networks, tabu search and simulated annealing.

1.7 Computer Programming Language – Visual Basic

Visual Basic (VB) is a high-level programming language from Microsoft. It is built upon the language called Beginners All-purpose Symbolic Instruction Code (in short, BASIC), a low-level programming language developed by John Kemeny and Thomas Kurtz in 1964. Microsoft's VB (the latest version is 6.0) is an improvement on BASIC, in the sense that it can be used to create applications with visual features (i.e., graphics), with much less code using Microsoft's Visual Studio interface.

Any such application begins with the simplest of forms on which other components and features (also known as widgets) such as drop down menus and buttons can be added to just by dragging and dropping them from the appropriate graphical menu. These components or widgets have their own individual functions and properties which the user can tweak and exploit to create customized applications and interfaces to applications such as Excel and Word, the many features of which, by virtue of being products from Microsoft, can be accessed through VB and be programmed for purposes specific to the user's needs.

One major plus point about VB 6.0 is that it supports Object oriented programming, which is the quintessence of present day programming logic. In addition,

VB is relatively easier to learn than most other programming languages. It has in fact, replaced BASIC as the programming language of choice for a beginner with hitherto zero knowledge of programming computers.

VB is so versatile that applications, diverse in complexity can be created. The Cell Loading and Cell Scheduling (CLACS) application 1.0 has been coded using VB6.0. This application works on event-based programming, i.e., all the processing is triggered off by an event, such as the pressing of a particular button in the interface while the underlying code is built heavily on structured coding. The design features of CLACS are described in Chapter 5. This is but the first completely working version of this application (Hence, the nomenclature 1.0). It is hoped that this application will be used as a base for including much more features and rules that might prove to be helpful in loading and multi-resource constrained scheduling in this and similar cellular manufacturing systems.

1.8 Thesis Objectives

This thesis proposes a set of heuristic procedures for Cell loading and Cell scheduling for the Rotary Machine Cells that are part of the shoe manufacturing plant owned by Timberland Inc. in Puerto Rico. The objective of the heuristic procedures is to ensure that all the jobs are loaded without exceeding the given capacity of the cells. The same procedures can also be used to minimize makespan by carefully defining certain parameters used in the heuristics. A software application for loading and scheduling these cells is developed based on the procedures proposed. Using this software application, the procedures are compared and the results are presented.

The software application is used to load and schedule the job orders received at the shoe manufacturing facility and determine the level of the resources needed to fulfill these job orders (i.e., find the number of cells required, types of cells, find the number of molds required of each type and the processing time required on each cell). In other words, the application (i.e., the software program) first groups the shoes from the orders into families based on the characteristics of these shoes and then indicates the sequence in which the families are processed in each cell (based on the current load on the each cell). The application then schedules the jobs in each family in the respective cell.

1.9 Organization of This Thesis

This thesis begins with a general introduction about the background of this problem and defines all its major characteristics, i.e., Cellular Manufacturing, Cell Load, and Cell Scheduling. The various basic concepts involved in this problem are mentioned and defined. Once the background has been clearly illustrated in Chapter 1, a review of relevant literature is presented in Chapter 2 and the ideas proposed by other researchers are mentioned in brief. Chapter 3 defines the problem in detail and presents all its characteristics to give the readers some idea about this case. Chapter 4 presents the methodology used to provide a viable solution to the problem studied. In Chapter 5, the results of this study are presented in detail and in Chapter 6, conclusions and possible future research is suggested.

CHAPTER 2. LITERATURE REVIEW

The summary of relevant research on the topics associated with the loading and scheduling of manufacturing cells is presented in this chapter. The literature on cellular manufacturing is diverse and vast, with most of the work being in the areas of cell formation and cell scheduling. Cell loading does not find much attention devoted to it. This mainly results from the fact that cell loading decisions depend on the job characteristics and the item produced in the cells and hence, every single industry/production plant would have to evolve unique cell loading rules.

Ham and Hitomi [8] analyzed the problem of loading machines on multi-stage production systems. The authors developed a 0-1 linear programming formulation to optimally select parts for production on machines (i.e., for loading) within the given time period to maximize the production rate. The authors ignored the production sequence and defined job production time as the sum of job setup time and unit production time multiplied by lot size. The authors used a branch and bound methodology to solve the problem and make both the loading and product mix decisions and used a numerical example to demonstrate their procedure.

Sinha et al. [10] reviewed various production control problems in cellular manufacturing. The authors discussed several problems in cellular manufacturing such as batch size selection, period batch control and cell scheduling. Though the survey of problems is not comprehensive, it attempts to establish the salient and important features of cell scheduling and the various factors that affect the effective control of cellular manufacturing systems. The characteristics that affect scheduling of manufacturing cells such as production flow, sequencing, and solutions such as group scheduling were

discussed. The authors made a set of nine recommendations based on their study of relevant literature and advocated the use of period batch control for controlling the production in a manufacturing cell, provided that the demand was stable.

Morris and Tersine [11] compared various practices that can be adopted in a group technology environment, i.e., cellular layouts with machine loading, cellular layouts with cell loading and process layouts. The paper discusses the results obtained by simulating these three techniques (using SIMAN) with respect to throughput and WIP inventory levels. These techniques are statistically ranked and compared using a hypothetical job shop. The research concludes that at lower utilization levels, a cellular layout with cell loading, which makes use of the whole cell to perform work on a product, is preferable. A sensitivity analysis that was performed indicated that the results are however, dependent on the arrival rate of the new jobs.

Süer and Saiz [14] focused on the cell loading aspects of the control of manufacturing cells. In this study, cell loading problems are classified on a multilevel basis based on the number of cells in the system. Cell loading issues were also differentiated as Product Family Selection (PFS), Order of Products in each family (OPF) and Cell Selection (CS). This study combined the 48 rules generated by Süer et. al (1993) with 4 main scenarios in cell loading to arrive at 192 different ways of loading manufacturing cells based mainly on whether the cells in question are Independent or Connected and whether pre-emption of jobs can be allowed or not (the 4 main scenarios mentioned above). A Turbo Pascal application developed for this purpose was used to test the performance of these rules and the preliminary results were presented.

Süer et. al. [15] discussed eleven rules and algorithms for cell-loading in a system of connected manufacturing cells. Though none of the single rule combination could be termed the best with respect to all the performance measures, interestingly it was observed that when the rules were combined, the rule combinations that worked well for the CU performance measure did not work well with respect to the performance measures such as number of tardy jobs (nT), Total tardiness (TT) and maximum tardiness (Tmax). This is perhaps an indication that if we strive for increased utilization in manufacturing cells, then the performance with respect to the other performance measures might take a beating.

Süer et. al. [18] evaluated cell loading rules for independent manufacturing cells. This paper focuses on cell loading aspects in CM and is unique because, contrary to conditions usually replicated in other studies, this paper studies a multi-cell environment where cell loading becomes crucial for controlling the entire system. The authors introduce several cell loading rules for such cells and several combinations of such rules are also studied. The study did not produce any clear winners or losers in terms of the rules put forth, but found that few of the rules produced almost identical results. The authors found that if the performance measures have different weights, choosing the best rule becomes more complicated. This is also the case if different performance measures are preferred by the user.

Süer [21] studied and proposed a two phase hierarchical approach to determine optimal manpower requirements for labor intensive manufacturing cells and also the load on the cells, simultaneously using mixed integer and integer programming formulations. The model suggested that the results arrived at after the application of the first phase of

the two phase methodology remain valid as long as processing times and the operator levels remain unchanged and the second phase would have to be re-run as and when demand changes from period to period. The author illustrated his methodology with a real case and found that the methodology suggested is valid for most labor intensive applications.

Akturk and Wilson [22] proposed a hierarchical approach that enhanced Hierarchical Process Planning in several ways using key features of GT based manufacturing systems and utilizing knowledge of GT-based manufacturing systems. The authors have described a hierarchical cell loading approach to solve the production planning problem in cellular manufacturing systems. The authors have aimed to minimize the variable cost of production with respect to the production and inventory balance constraints. The capacity constraints were added to the problem formulation and the production rates were set to be within the designed capacity of the system. A mathematical programming formulation was used to solve the cell loading problem. The effects of several system parameters are evaluated using a full factorial design. A two-way analysis of variance was used to test the observed responses when the total production cost and the computation time was varied. The paper found that the proposed approach allows more accurate portrayal of the operation of cells. The analysis found that the number of GT cells and families had a significant effect on the total production cost and the computation time.

Süer and Bera [23] studied the simultaneous multi-period cell loading and cell size determination in labor-intensive cells. Since the analysis performed is multi-period in nature, the objective is to maximize the products that can be produced at the same

capacity available over a number of periods considered. The solution proposed in this case is a two-phase hierarchical model in which the first phase generates alternate configurations and the second phase generates the optimal operator assignments and the cell loads. It is assumed in this case that a job can be loaded only to one feasible cell, i.e., lot-splitting is not allowed.

Riezebos [24] studied and illustrated the changes to be affected in the production planning systems when applying the cellular manufacturing paradigm to existing production systems and proposes a new hierarchical framework that gives attention to decisions that place emphasis on information available on resources, orders and time with the aim of designing a production planning and control system for cellular manufacturing (CM). The author also discussed different contributions from other studies on the effectiveness of other famous approaches of planning and control to CM, such as Materials Requirement Planning (MRP), Kanban, and Hierarchical Production Planning (HPP). Burbidge's contribution to production control, and the applicability of Period Batch Control as an effective planning system for CM was also discussed.

Süer and Bera [25] extended previous research [21] and studied the simultaneous determination of optimal operator assignment and cell loading in labor intensive manufacturing cells. In this case however, lot-splitting was allowed. The authors modified the mathematical models from their previous work to allow for lot-splitting and to account for set-up times.

Mahmoodi and Mosier [26] presented the results of their group scheduling research and its managerial implications are illustrated with some group scheduling heuristics that can be applied to a small problem. The authors have defined group

scheduling procedures as scheduling procedures that exploit setup similarities between jobs in the same family, by grouping such jobs together. The authors classified group scheduling problems into two categories - scheduling of single stage and scheduling of multi-stage systems. The research found that most group scheduling heuristics such as Work Shortest Processing Time (WOSPT) perform much better than single stage rules with respect to a wide range of factors. The authors suggested that cell performance can be improved by concentrating on production planning activities aimed at reducing variation in inter-arrival times and leveling the cell load. They also found that performance differences between group scheduling rules are much lesser when compared to traditional rules and hence, making the choice is much easier.

Süer et al. [27] developed a hybrid approach of evolutionary programming and local optimizers in cell loading to minimize the number of tardy jobs. The authors suggest a trio of different approaches which were then compared. The first two approaches involved two identical steps i.e., an evolutionary programming approach to generate job sequence followed by a classical rule (minimum load and the Moore's algorithm) to assign jobs to cells. In the second approach (called the three phase approach) each cell was subjected to a local optimizer after the end of the second phase. The third approach discussed is a three phase approach too and is identical to the second, but with a learning mechanism built in. The three methodologies were tested with different crossover and mutation strategies and the authors concluded that the local optimizer improves results drastically and obviously the number of tardy jobs decreased with increase in the number of cells.

Ruben and Mahmoodi [28] concentrated mainly on group scheduling heuristics and presented a number of such heuristics in detail. The topic of scheduling flow line cells was discussed under combinations of a number of different conditions and variations in factors such as arrival times and set-up times. Job Shop cells and their scheduling were discussed too and some heuristics for the same were also illustrated. The positives of group scheduling procedures over single stage scheduling was illustrated and it was suggested that selecting an inappropriate single stage group scheduling rule would affect the performance of the cell much adversely than opting for group scheduling and then choosing an inappropriate group scheduling rule.

Süer et al. [31] introduced the cell loading problem in a shoe manufacturing facility. The objective was to reduce makespan and the authors used a three phase heuristic based on Largest Processing Time (LPT) to arrive at a final schedule and illustrated the procedure with an example. The authors assumed that a single rotary injection molding machine was available per cell group. Each rotary injection machine can process n pairs of shoes in the n pairs of positions available in the machine. The shoes are of different sizes and hence, different molds would be required. The cycle time for each pair of shoe was dependent on the size and the whole injection process needs to be stopped whenever a mold needs to be changed. The authors developed a 3-phase heuristic to generate feasible schedules while minimizing makespan. The first stage of the heuristic determined the production quantity based on the number of the outsoles and uppers available and determines an initial loading sequence based on the number of turns of the rotary machine involved (which again depended on the size of the shoes). The second phase determined intervals for each production cycle and also processing times.

The third phase determined a feasible schedule based on the sequence of jobs as determined in the first stage and the processing times determined in the second stage after adding determining set-up times as and when required. The problem and the heuristic were illustrated using a typical example data set.

Mahmoodi et al. [32] examined the effects caused on the performance of a constrained, random flexible manufacturing system (FMS) due to scheduling rules and routing flexibility. The various experimental factors considered in this study are load on the system, shop configuration, and breakdowns in the system. The study develops a new scheduling rule and compares its performance against several other existing rules like the Shortest Processing Time (SPT) rule. The results obtained in this study have indicated that, if and when total routing flexibility can be achieved, the effects caused by the various experimental factors mentioned above reduced significantly and when total routing flexibility is achieved, the choice of scheduling rules chosen is not critical.

Lozano et al. [33] described the cell design and the loading of the cell when alternative routing is available. This paper defined the problem in 2 phases, the first one dealing with the cell design performed and then the 2nd phase being the cell loading part which is performed again and again until an optimal solution results. The 1st phase consists of 2 alternatives, one considering separate part plans and the other alternative considering aggregate part plans. In both alternatives, the cell formation method used to group the machines was conservative in nature. A multi period Linear Programming formulation calculated how much quantities of parts follow each alternative path in each period in order to minimize transportation and other costs while keeping the utilization of

machine and the cell balanced. The authors illustrated the problem with a manufacturing system consisting of 12 machines and 12 part types.

Gupta et al. [35] aimed to minimize flow-time on identical parallel machines subject to optimal makespan by using a hierarchical multi-criteria scheduling process. The problem in question consisted of a set of n jobs that could be processed on m parallel machines. It is assumed that the number of jobs is an integral multiple of the number of machines and if not artificial jobs with zero processing times are added to make up the numbers. These jobs were sorted by LPT. A number of other assumptions such as deterministic processing times were made. The problem was formulated as a dual level binary integer programming problem. The first level calculates optimal makespan while the second level is used to minimize flow time based on the optimal makespan from the first level. The author modified the lexicographic search based algorithm developed by Ho and Wong [20] for application in this situation and used several numerical examples to state their case. The study empirically showed that the proposed algorithm quickly finds optimal schedules for large problems while minimizing makespan using an existing optimization algorithm.

Hans [37], proposed a modeling approach that offers a generic framework for different types of resource loading and rough cut capacity problems as integer linear programming models. The production environment considered in this research was a Made-To-Order environment in which the resource loading models and methods introduced are used to support order processing by determining reliable due dates for known customer orders. They are also used to determine the resource capacity levels that are required to load these orders on the system. Since detailed order characteristics are

only partly known at the maximum, the author did not perform a detailed planning, but precedence constraints were imposed for, example, between cells. Once the orders have been processed, the resource loading function is used to determine the available resource capacity levels for the underlying scheduling function in the problem. The author also proposed algorithms to solve typical problems to optimality and a deterministic approach for modeling and solving resource loading problems.

Alejandro G., M. [38] proposed a three phase hierarchical hybrid approach for simultaneous cell loading and manpower allocation and then sequencing of jobs in cellular manufacturing systems based on an evolutionary programming methodology. The author illustrated results from the first two phases using an example problem made up of a combination of real time and artificially generated data that characterized the problem.

Saad et. al. [39] discussed a “multiple objective simulation optimization model” for loading flexible cells. A search algorithm called the tabu search (TS) algorithm developed by the authors was used to solve the problem. The system is composed of three main modules. The Generic Process Planning Module determines processing requirements of parts in terms of Resource Elements (RE). The generated process plans are abstract process plans without specific machine. The outcome of the cells loading conditions are used to generate the final machine-based process plans. A Multiple Objective Tabu Search Algorithm is used to generate and evaluate candidate part to cell assignment scenarios. The performance of the generated part to cell assignment scenarios and cell schedules is determined using the Simulation and Scheduling Module. The system proposed, has been applied to a manufacturing facility containing 12 machines

and 4 manufacturing cells. The simulation module determines performances and schedules of the cells. After each simulation, the TS algorithm evaluates the performance measures, which continues until optimal solution results from iterations. Several assumptions are made for this purpose. The output represents the part type assignment and their production schedule. The results reflect the fact that this process causes improvement in cell performance levels.

Babayigit [40] analyzed the Manpower Allocation and Cell Loading problem (MACL) and proposed a mathematical model and genetic algorithm (GA) solution for the problem. The main difference between MACL problem and parallel machine scheduling problems is that in uniform parallel machine problem, the number and the speed of the machines are certain whereas in this problem, both the number of cells and configuration of cells have to be determined by the solution. The models developed were extensions of the one developed by Sürer [21]. Here it is assumed that all the present cells have the capability of all worker configurations. As a result, some of the parameters of the original model have been eliminated. Based on original model, four different mathematical models were developed to include the number of tardy jobs performance measure. A Flexible GA application was developed using Visual Basic and some original aspects such as Multiple League and Extreme League were included into the traditional GA methods. The study showed that the GA outperforms the traditional methods for some performance measures and for large problems finds optimal or near optimal solutions much faster than a mathematical model.

CHAPTER 3. PROBLEM STATEMENT

Timberland Inc. is a major manufacturer of shoes and outdoor apparel with a number of manufacturing facilities around the world. The problem described in this thesis can specifically be observed at the Timberland facility at Mayaguez. The salient features of this problem are described in the subsequent sections of this chapter.

3.1 A Typical Cell group

The shoe manufacturing plant considered in this study consists of six Connected Cell Groups (Figure 3-1). Each of these cell groups consists of a Lasting Cell (LC) - to prepare the shoes for injection molding the soles, a Rotary Machine Cell (RMC) with 6 pairs of stations (a pair of which can process one pair of shoes at any given time) and a Finishing/Packing Cell (FC) to remove extra material from the sole, finish the shoe and to pack the shoe. There is no machine sharing or operation sharing among cell groups, since each cell group is independent of the others.

3.2 Current Cellular Configuration

The shoes that are being processed in the system move sequentially from the LC to the RMC and then to the FC. The processes that the shoes undergo in the Lasting cell are sequential. These processes vary based on the model of the shoe. But for every model, these processes are similar regardless the size of the shoe. Hence, the LC can be treated as a single machine. This treatment simplifies the scheduling of the cell group, but this is deemed acceptable in the context of this thesis which concentrates only on the rotary machine. The single piece flow feature of this system and the presence of a bottleneck

machine (the RMC) that determines the output system add to the relevance of this assumption. Once the RMC is scheduled, the LC can be scheduled by a system of Backward Scheduling. Once the material has been injected, the shoes are unloaded from the rotary molding machine. The shoes are then transferred to the FC. The FC can also be treated similarly as a single machine. In the FC, any extra material that may have been injected into the mold is scrapped. The finished shoes are then packed as per specifications and moved to the warehouse for shipping to different locations later.

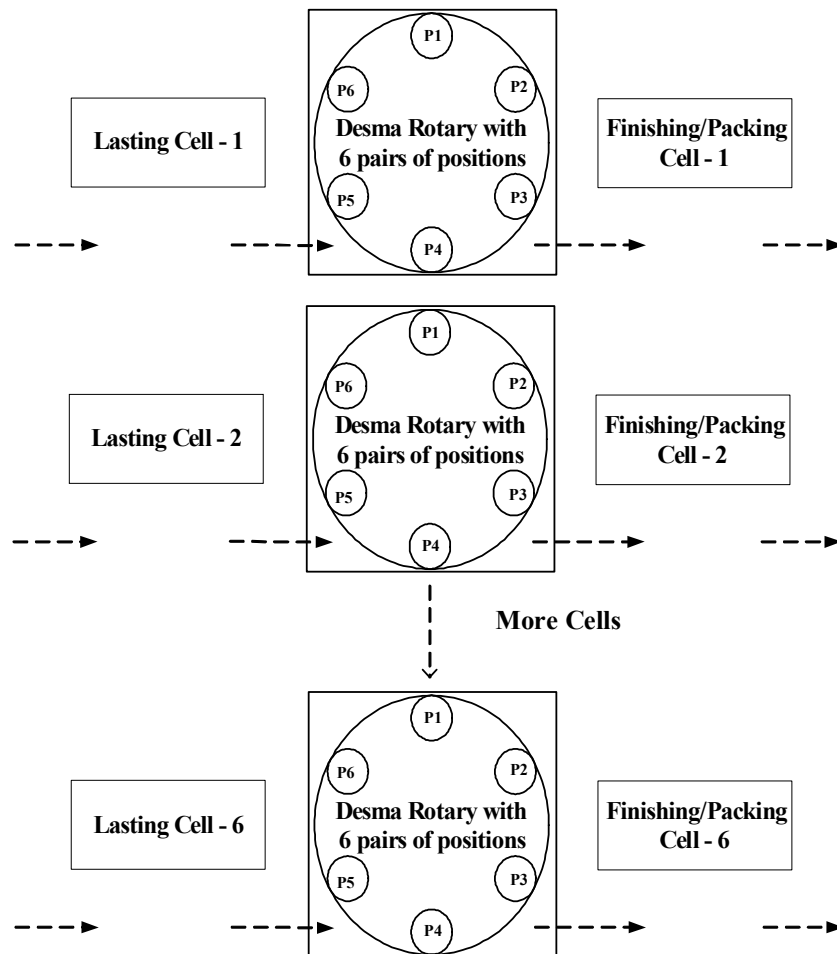


Figure 3-1: The cell groups at the plant

3.3 The Jobs

Each customer order typically contains shoes of different models, each model in turn having shoes for each sex (male or female), different sole types (Full Shot and Mid Sole), colors (Black, Honey and Nicotine) and material (PVC and TPR). Demands are entered on a weekly basis and specified for each size of a shoe individually. The production volume of shoes of particular sizes vary as a distribution as shown in Figure 3-2. Hence, the largest volume is for shoes whose sizes range in the center of the spectrum of sizes, with the smallest and the largest sizes having the lowest production volumes.

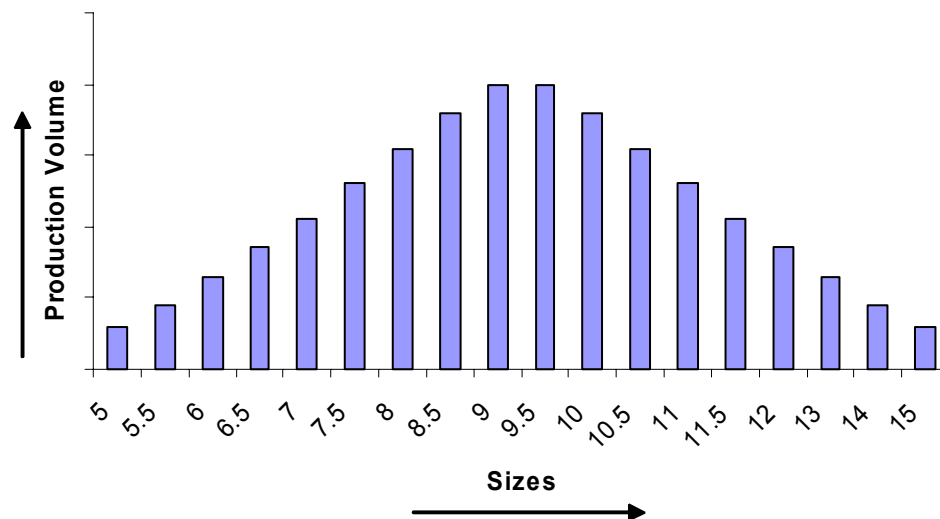


Figure 3-2: Size Distribution and Production volumes (Male shoes) – A Sample

3.4 The Rotary Machine Cell

The Rotary machine Cell consists of a single Rotary injection molding machine. The main parts of the machine are - a tank to hold the material to be injected and 6 pairs

of positions that enable the injection molding to be performed on 6 pairs of shoes. Once the unprocessed shoes are loaded into the rotary machine, material (either PVC or TPR) is injected into the molds, thus forming the soles. It is to be noted that the machine can handle one material and only one pigment color at a time because of the single tank that is present.

3.5 Rotary machine operation – Loading Shoes

Let us consider a set of 7 jobs i.e., 7 pairs shoes (Table 3-1). The process of loading these shoes on the Rotary injection molding machine will now be demonstrated. Initially, all the 12 positions (i.e., 6 pairs) in the machine are empty (Figure 3-3).

Table 3-1: Sample list of jobs

| Jobs | Type | Material | Color | Sex | Sizes | Injection Time (min.) | Injection Time (sec.) |
|-------------|-------------|-----------------|--------------|------------|--------------|------------------------------|------------------------------|
| 1 | Full Shot | PVC | Black | Male | 5.5 | 0.224 | 13.4 |
| 2 | | | | | 6 | 0.247 | 14.8 |
| 3 | | | | | 6.5 | 0.269 | 16.1 |
| 4 | | | | | 7 | 0.281 | 16.9 |
| 5 | | | | | 7.5 | 0.289 | 17.3 |
| 6 | | | | | 8 | 0.290 | 17.4 |
| 7 | | | | | 8.5 | 0.301 | 18 |

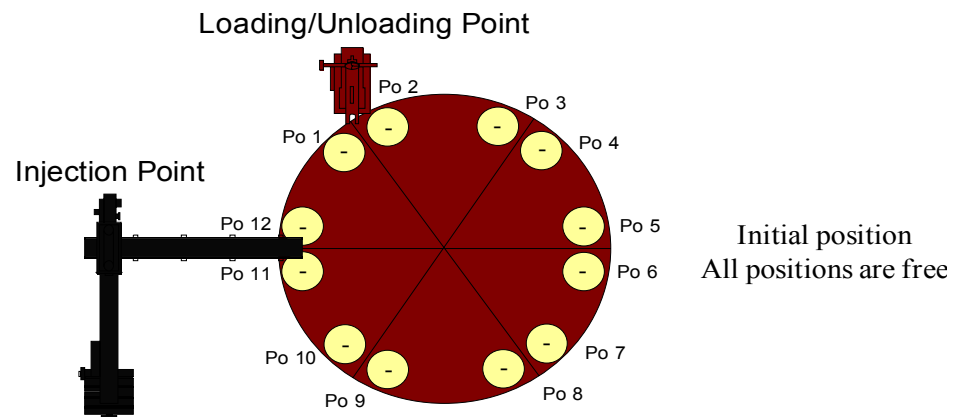


Figure 3-3: Initial Position of the Rotary Injection Molding Machine

Now at time $t = 0$ (Figure 3-4), the first pair of shoes to be loaded (in this case, their size being 5.5) are loaded into positions 1 and 2 (represented by Po 1 and Po 2 in Figure 3-3). The rotary machine is rotated anti-clockwise until the next pair of empty positions (in this case, Po 3 and Po 4) is in position at the loading point and the just loaded positions (in this case Po 1 and Po 2) are at the injection position. At this point, injection of material would begin at the end of 18 seconds.

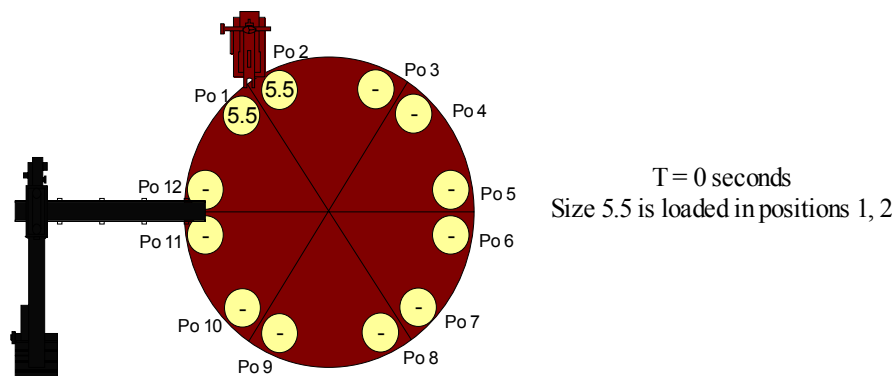


Figure 3-4: First pair of shoes loaded at $t = 0$ seconds

Therefore at time $t = 18$ seconds (Figure 3-5), positions Po 3 and Po 4 are ready to be loaded and the next pair of shoes (size 6) is loaded in these positions. The injection of material into the molds at Po 1 and Po 2 starts and the machine is rotated anti-clockwise again to bring the next pair of positions (Po 5 and Po 6) to the loading point.

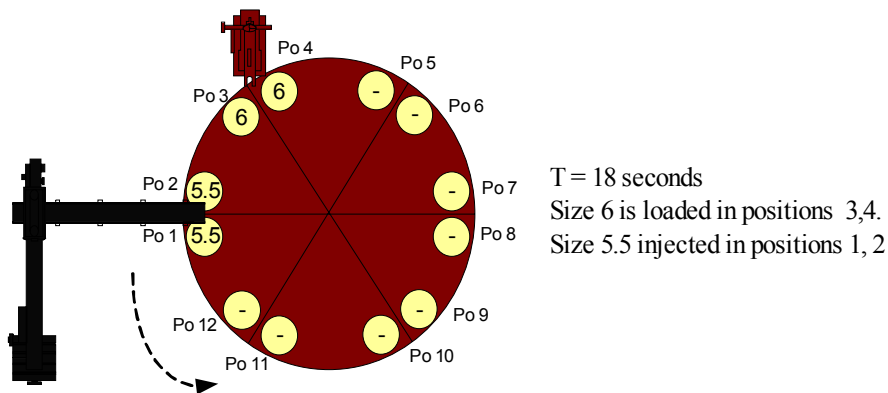


Figure 3-5: Loading the second pair at $t = 18$ seconds

Hence, at $t = 36$ seconds (Figure 3-6), once the positions Po 5 and Po 6 are at the loading point the third pair of shoes in the list (size 6.5) are loaded and material is injected into positions Po 3 and Po 4 and the machine is rotated again.

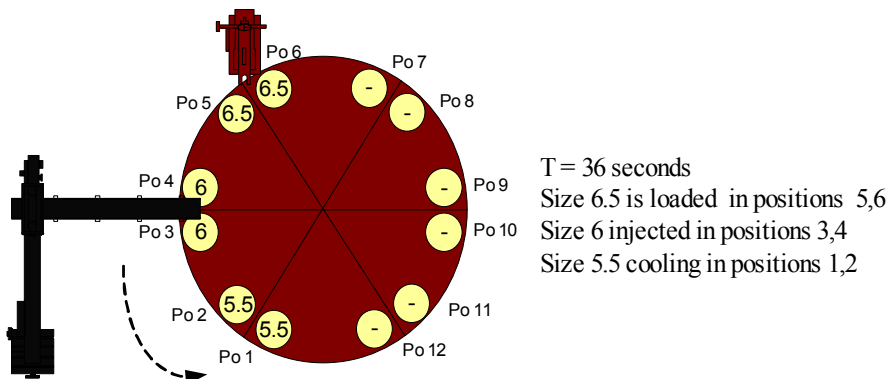


Figure 3-6: Loading the third pair of empty positions at $t = 36$ seconds

Now, the next pair of free positions (Po 7 and Po 8) arrives at the loading position at $t = 54$ seconds (Figure 3-7) and the next pair of shoes (size 7) are loaded into these positions. Material is injected into Po 5 and Po 6 that are at the injection point and the anti-clockwise rotation of the machine begins again. Shoes cannot be removed until the positions come back to the loading position again after one complete rotation.

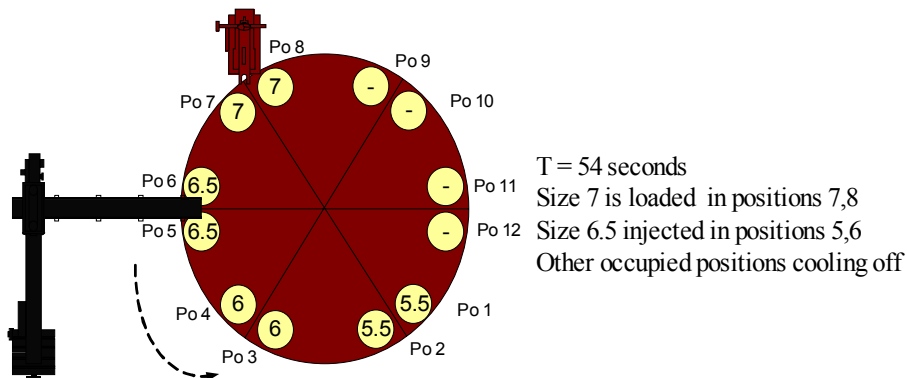


Figure 3-7: Position of the Rotary Machine at $t = 54$ seconds

At $t = 72$ seconds (Figure 3-8), shoes of size 7.5 are loaded into Po 9 and Po 10 and injection begins into Po 7 and Po 8. By this time, the shoes loaded in Po 3 and Po 4 have been completed and are being cooled off until they can be removed completely. The machine is rotated anti-clockwise again to bring the next pair of empty positions to the loading point and the next set of loaded molds to the injection point.

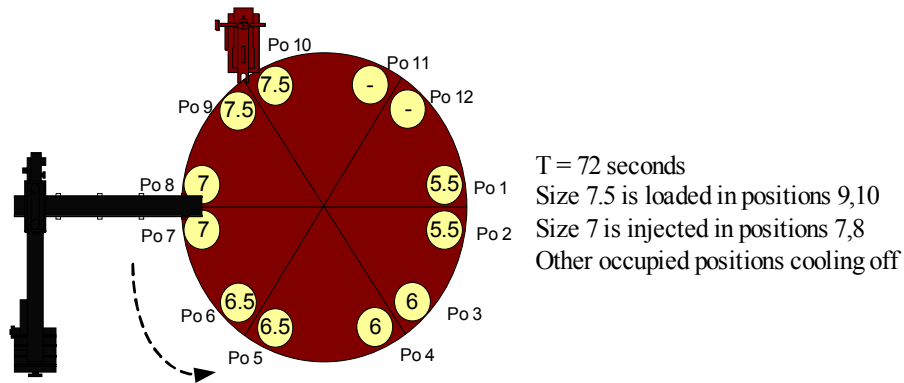


Figure 3-8: Position of the Rotary Machine at $t = 72$ seconds

At time $t = 90$ seconds (Figure 3-9), the next pair of shoes on the list, i.e. size 8, are loaded into the machine and injection starts into the molds in Po 9 and Po 10. By this time, the shoes in positions Po 5 and Po 6 are being cooled down.

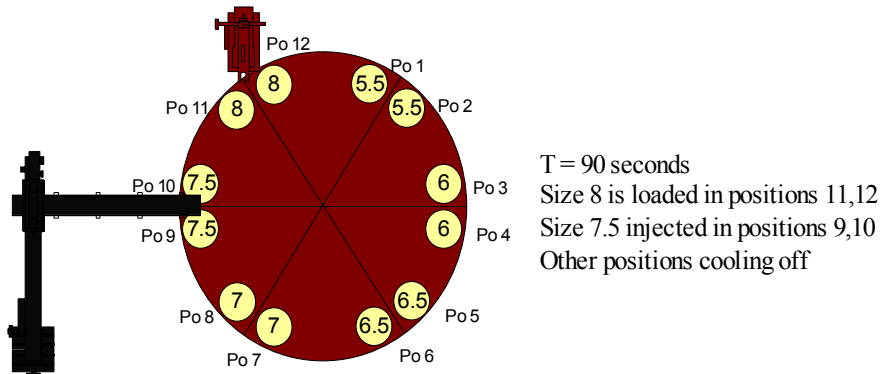


Figure 3-9: Position of the Rotary Machine at $t = 90$ seconds

Now at time $t = 108$ (Figure 3-10), the machine completes one full rotation and positions Po 1 and Po 2 arrive at the loading/unloading point. Now the shoes in these positions have been processed completely and can be unloaded. Hence, these shoes (size 5.5) are removed and transferred to the Finishing/Packing cell. The next pair of shoes to

be loaded (size 8.5 in this case) are loaded into Po 1 and Po2, while injection takes place in Po 11 and Po 12 that are at the injection point. The machine is rotated again to bring the next set of molds into position.

This completes one full rotation of the Rotary injection molding machine. These steps are repeated until a change of color / material becomes necessary, upon which the tank is cleaned to remove all traces of the existing color / material.

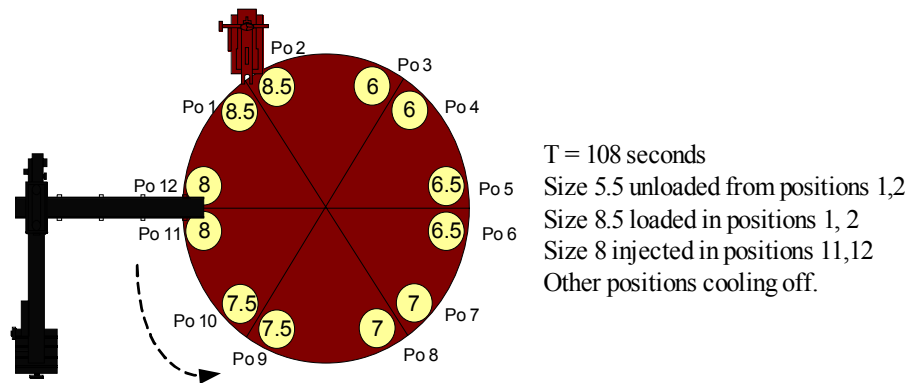


Figure 3-10: Position of the Rotary Machine at t = 108 seconds

3.6 Molds

The molds required for processing each shoe depends on the size of the shoe. And the availability of the molds is also limited by their prohibitive costs. Since the molds vary by size and by type (Full Shot or Mid Sole), they have to be constantly changed as and when the corresponding shoe has to be processed on the molding machine. This change also impacts the setup time between successive usages of every mold because the molds have to be cleaned as and when the material to be injected or the pigment color of the material to be injected change. This is to ensure that the material injected or its pigments are not contaminated. In addition to the molds, the tank holding the material in

the Rotary machine would also have to be cleaned whenever such a change occurs. As a result the rotary machine would have to be stopped while the tank (and the molds too if they have to be used again on the same machine with a different color/material immediately) is cleaned.

3.7 Processing Times

The processing times (Table 3-2) for the rotary injection molding machines are the basis for cell loading and scheduling. These injection molding times vary based on size and sole type. Injection times are higher for Full Shot shoes than the Mid Sole models and this injection time increases as the shoe size increases. On an average, for Full Shot soles, the injection molding time is 0.356 minutes for male and 0.3 for female shoes, while for Mid Sole models it is 0.286 minutes for male and 0.25 for female shoes. Since this processing time increases with size, care must be taken to minimize the variations in the processing times of shoes that have been loaded together in the six pairs of stations in the Rotary Machine as this variation decides the length of time each shoe would spend in the Finishing Cell.

Since the shoes that have been loaded in the Rotary machine cannot be unloaded until all the shoes have been processed, the shoes with the lowest processing times spend much more extra time in the machine as compared to the shoes with the highest processing time. Now the amount of material injected increases with the time spent on the machine and hence, the shoes with lowest processing times would have much more material injected into their molds than required. This extra material would have to be scrapped in the Finishing Cell and hence, the time spent there increases too.

Table 3-2: Processing Times

| Mold Type | Sex | Size | Injection Times | | Mold Type | Sex | Size | Injection Times |
|-----------|--------|------|-----------------|--|-----------|--------|------|-----------------|
| Full Shot | Male | 5 | - | | Mid Sole | Male | 5 | - |
| | | 5.5 | 0.224 | | | | 5.5 | - |
| | | 6 | 0.247 | | | | 6 | 0.160 |
| | | 6.5 | 0.269 | | | | 6.5 | 0.162 |
| | | 7 | 0.281 | | | | 7 | 0.203 |
| | | 7.5 | 0.289 | | | | 7.5 | 0.239 |
| | | 8 | 0.290 | | | | 8 | 0.240 |
| | | 8.5 | 0.301 | | | | 8.5 | 0.242 |
| | | 9 | 0.306 | | | | 9 | 0.245 |
| | | 9.5 | 0.320 | | | | 9.5 | 0.247 |
| | | 10 | 0.342 | | | | 10 | 0.273 |
| | | 10.5 | 0.372 | | | | 10.5 | 0.293 |
| | | 11 | 0.426 | | | | 11 | 0.311 |
| | | 11.5 | 0.428 | | | | 11.5 | 0.349 |
| | | 12 | 0.448 | | | | 12 | 0.398 |
| | Female | 13 | 0.460 | | | | 13 | 0.401 |
| | | 14 | 0.499 | | | | 14 | 0.405 |
| | | 15 | 0.556 | | | | 15 | 0.415 |
| | | 5 | 0.154 | | | Female | 5 | 0.109 |
| | | 5.5 | 0.207 | | | | 5.5 | 0.116 |
| | | 6 | 0.216 | | | | 6 | 0.183 |
| | | 6.5 | 0.229 | | | | 6.5 | 0.190 |
| | | 7 | 0.230 | | | | 7 | 0.196 |
| | | 7.5 | 0.237 | | | | 7.5 | 0.260 |
| | | 8 | 0.308 | | | | 8 | 0.261 |
| | | 8.5 | 0.338 | | | | 8.5 | 0.290 |
| | | 9 | 0.356 | | | | 9 | 0.305 |
| | | 9.5 | 0.368 | | | | 9.5 | 0.310 |
| | | 10 | 0.395 | | | | 10 | 0.342 |
| | | 11 | 0.424 | | | | 11 | 0.344 |
| | | 12 | 0.440 | | | | 12 | 0.347 |

3.8 Special Considerations in the Cell loading Phase

The main objective of this thesis is to develop procedures that will load and schedule jobs such that they can all be completed with the available capacity. However some of these procedures can also be used to minimize makespan as well.

The cell loading procedures mentioned in this thesis have been formulated after making the following special considerations.

3.8.1 The Loading Procedure

Re-arranging job/families using the Largest Processing Time procedure is an integral part of any heuristic procedure whose objective is to reduce the makespan. The main objective behind developing methodologies to load and schedule the cells in this case is the reduction of makespan too. Hence arranging the families in LPT is the first step in three of the four the heuristics developed in this thesis.

The use of the LPT algorithm can be justified by the fact that it helps to balance loads and to maximize utilization in the cells. This is because if the families would be loaded in random (or any other) order, then there is always a chance that even if the cells have leftover capacity, some of the families that have not been loaded yet would possess much more loads than the available capacity in the cells. When the LPT algorithm is used, then the largest families will be processed first and the leftover capacity can be easily utilized to process the small families.

3.8.2 *MaxCap*

The Timberland factory works for 40 hours of regular time every week. This translates to a total of 2400 minutes that will be available on each Rotary Injection Molding Machine each week. However some of these 2400 minutes will be spent idle if the molds required for injecting material in the shoes loaded on the machine are not available, if the machines themselves are down, if shoes arrive late from the Lasting Cell. Also some allocation will have to be made for times spent on setups due to color and/or material change etc.

The cell loading phase determines the sequence of families that will be processed on each Rotary Machine and does not consider the idle times, simply because these idle times are still not known at this stage. Hence some allowance has to be made in this regard. This allowance is determined by the value of MaxCap.

MaxCap or the Maximum Capacity to be loaded is a value always less than the 2400 minutes totally available. It is a user specified value that indicates the availability of each cell per period. All the heuristic procedures proposed in this thesis use this MaxCap value and hence it is critical to understand how it affects the loading procedures.

The value of MaxCap not only determines the number of cells required for processing any given set of families and if a shared cell is needed, but also how the load is spread over the different cells in the facility. The details of the impact of MaxCap on the performance of the procedures will be discussed in later sections.

3.8.3 The Shared Cell

The concept of the Shared Cell results from the fact that it is disadvantageous not to use excess capacity available even if it sometimes means that the setups required between different families would be diverse enough to avoid doing the same in all the cells. It might however be advantageous to use a single shared cell if in doing so, we could avoid creating two specific cells instead which will anyways be functioning with much less utilization than the other cells. But a shared cell also means problems due to logistics and manpower, not to mention an irregular flow of finished material into the finishing cells that immediately follow the Rotary Injection Machines. Hence, the presence of a shared cell should be avoided as much as possible.

In a way, the value of MaxCap also drives the decision whether to create a shared cell or not. In fact, for some values of MaxCap, the need for a shared cell can be eliminated by spreading the load evenly among the other cells. Hence this means that the choice of MaxCap will increase the efficiency of cell loading and cell scheduling methodology by much more than it is apparent after just the cell loading phase.

CHAPTER 4. METHODOLOGY

This research aims to formulate a methodology to determine the number of cells (each comprising of a single rotary injection molding machine with six pairs of stations) needed to manufacture orders (each comprising different models of shoes with various different sizes at different volumes) and load these orders on to the cells after grouping them into families. The research also aims to develop a cell scheduling methodology to manufacture these orders once they have been loaded on to the cells.

The methodology for loading and scheduling the jobs (which will henceforth be alternately referred to as shoes too) is a multi-phase one. The three main phases are the Background Phase (Demand Aggregation and number of cells determination for super-families), Cell Loading Phase and Cell Scheduling Phase. Each phase of the methodology is a multi-step process by itself. The overall methodology is illustrated in Figure 4-1.

4.1 Phase I-A: Demand Aggregation and Number of Cells Determination

This phase involves the setting up of the data for the shoes from the orders generated. The steps (each of which will be explained in detail in the following pages) in this phase are data gathering, product grouping in super-families, estimation of the load for individual jobs and finally the determination of the number of cells for each super-family.

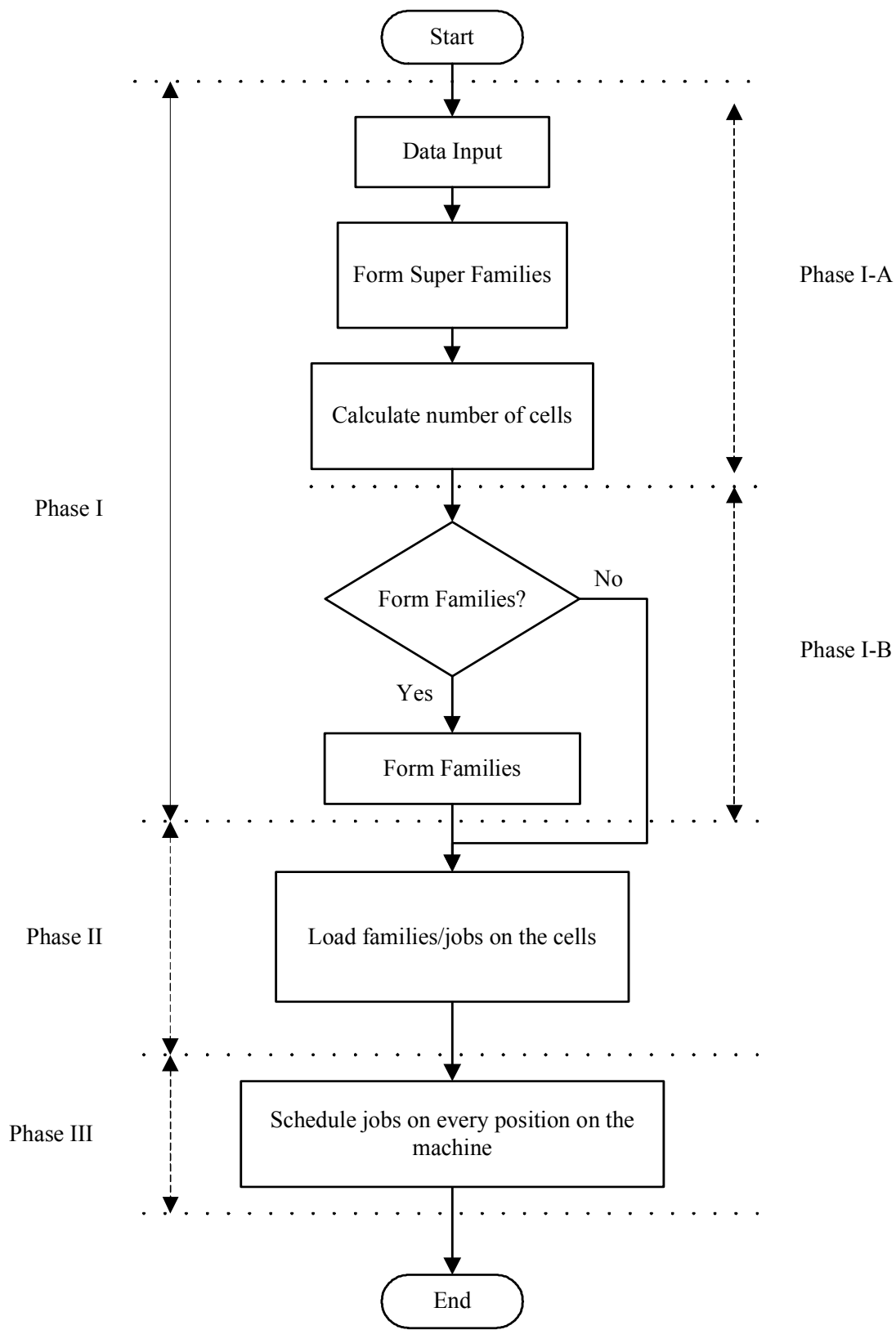


Figure 4-1: Overall Methodology

4.1.1 Data Input

The first step of Phase I involves the input of data. The shoe data taken from the customer consists of an order id, shoe model id, shoe sole type (Full Shot, Mid Sole), shoe sex (male or female), material (PVC, TPR), color (black, honey or nicotine), shoe size and demand (in number of pairs). Using this data, a key that incorporates all the relevant data of a particular shoe is generated automatically. A section of a typical example of the data is shown in part by Table 4-1. The key generated for the first job in the example in Table 4-1, is FFTB-K, a string formed by the first letters of the sole type, sex, material, and color with the model id after the hyphen.

4.1.2 Grouping of Products in Super-families

The data is grouped into super-families based on the different types of soles, which in this case would be into two groups – Full Shot and Mid Sole. This separation (shown in Table 4-2) is critical to the family formation later because of the comparative large difference in the processing times of these two types of shoe soles. The reasons for defining super-families is that the molds for the full shot and the mid sole shoes are different and it is best that the two types are separated (whenever possible) and loaded on different cells to avoid confusion and minimize problems associated with accountability.

The processing times are as shown in Table 3-2. This difference in process times, if the Full Shot and Mid Sole are loaded together, affects the cycle time of the shoes loaded on the machine together. The Cycle Time, CT, is the time for the duration of which the material is injected into the molds. This cannot be varied for every mold and

hence, will have to be set as constant and is the maximum of the injection times of the shoes currently loaded on the machine. Hence,

$$CT = \max \{ \text{Injection times of currently loaded shoes} \}$$

Table 4-1: Typical data

| Order ID | Model ID | Mold Type | Sex | Material | Color | Size | Auto Key | Demand |
|----------|----------|-----------|-----|----------|----------|------|----------|--------|
| asd | k | FS | F | TPR | Black | 6 | FFTB-K | 269 |
| asd | u | FS | M | PVC | Black | 8 | FMPB-U | 688 |
| asd | c | FS | M | TPR | Black | 5.5 | FMTB-C | 1045 |
| asd | c | FS | M | TPR | Black | 7 | FMTB-C | 208 |
| asd | l | MS | F | TPR | Black | 5 | MFTB-L | 881 |
| asd | t | MS | M | TPR | Black | 6 | MMTB-T | 831 |
| asd | t | MS | M | TPR | Black | 10 | MMTB-T | 277 |
| dgh | o | FS | F | PVC | Black | 8 | FFPB-O | 250 |
| dgh | e | FS | M | PVC | Black | 8 | FMPB-E | 636 |
| dgh | w | FS | M | PVC | Black | 9 | FMPB-W | 384 |
| dgh | w | FS | M | PVC | Black | 10 | FMPB-W | 329 |
| dgh | f | MS | F | TPR | Black | 5 | MFTB-F | 440 |
| dgh | f | MS | F | TPR | Black | 9 | MFTB-F | 321 |
| dgh | n | MS | F | TPR | Black | 8 | MFTB-N | 355 |
| dgh | n | MS | F | TPR | Black | 8 | MFTB-N | 255 |
| dgh | x | MS | M | PVC | Black | 6 | MMPB-X | 788 |
| fds | e | FS | F | PVC | Nicotine | 5 | FFPN-E | 574 |
| fds | e | FS | F | PVC | Nicotine | 8 | FFPN-E | 245 |

So if the variations in these injections times are high (which would typically happen if Full Shot and Mid Sole shoes are loaded together) more material would get injected during the extra time that the Mid Sole shoe would spend in the machine (the cycle time being driven by the injection times of the Full Shot shoes in the machine). This obviously has an adverse effect on efficiency and also leads to lower utilization of

the positions on the machine. Hence, it is not practical to mix the Full Shot and the Mid Sole shoes while processing the shoes on the injection molding machine.

Also, at the end of the injection molding process, when the shoes are sent to the Finishing and Packing Cell, the Mid Sole shoes that have been processed together with Full Shot shoes have to be worked on for extra time to remove the redundant material that has been injected. This really means that this material is wasted. Typically this wastage would be almost 50 percent. This also means that the labor costs for the Finishing Cell would increase due to the extra manpower hours required.

It has to be mentioned however that this concept of Cycle Time will be relevant only during the scheduling phase since details of all the jobs that have been loaded together into the injection molding machine will not be known until then.

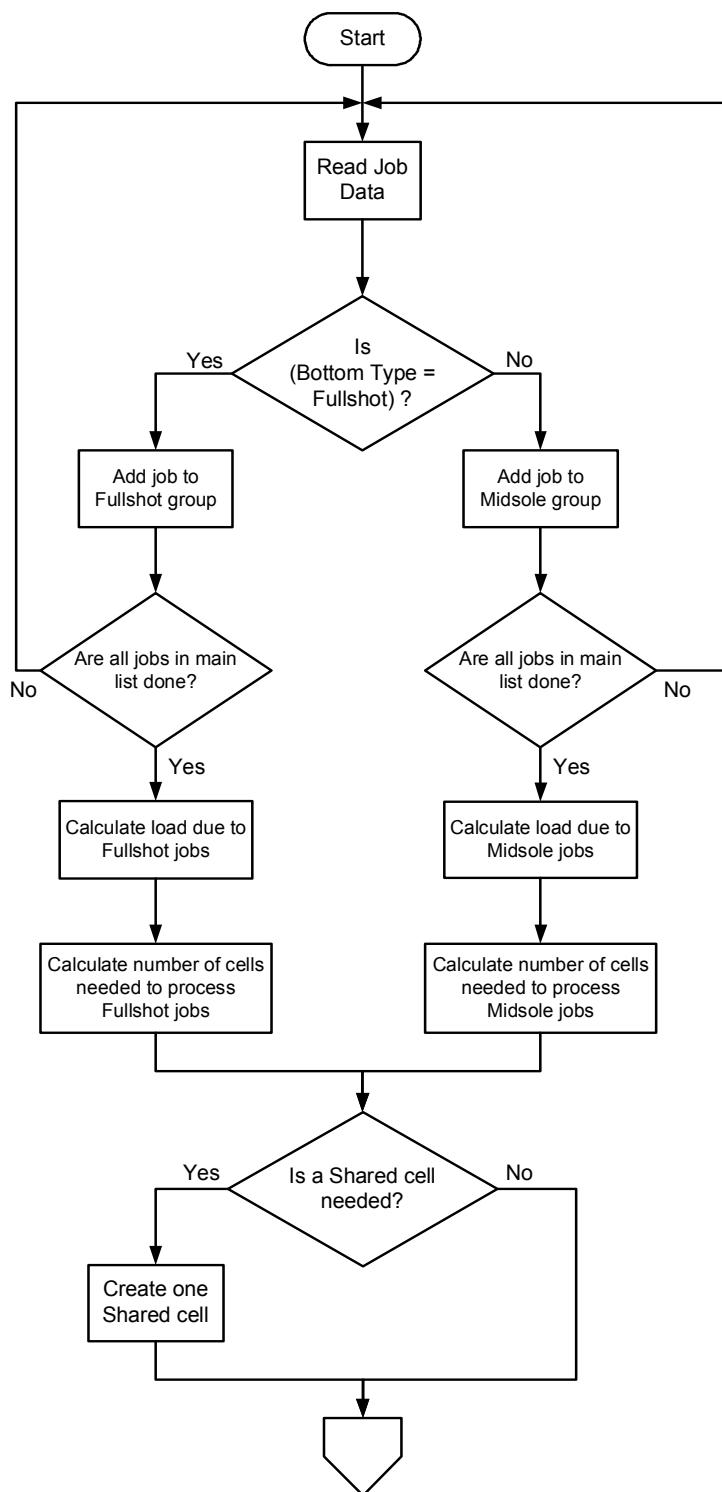
4.1.3 Calculations of Total Processing Times

The time required to process each shoe is calculated. If there are n different shoes (i.e. of different sizes and bottom types as shown in Table 4-1), the individual processing time, P_i (varies by sex, size and the bottom type, i.e. Full Shot or Mid Sole) of shoe “i” is multiplied with the demand (D_i) for that shoe to find the total processing time, T_i (see Table 4-3). So,

$$T_i = P_i * D_i \quad (4.1)$$

Table 4-2: Jobs divided into super-families (Sample)

| Order Id | Model Id | Mold Type | Sex | Material | Color | Size | Auto Key | Demand |
|----------|----------|-----------|-----|----------|----------|------|----------|--------|
| asd | k | FS | F | TPR | Black | 6 | FFTB-K | 269 |
| asd | u | FS | M | PVC | Black | 8 | FMPB-U | 688 |
| asd | c | FS | M | TPR | Black | 5.5 | FMTB-C | 1045 |
| asd | c | FS | M | TPR | Black | 7 | FMTB-C | 208 |
| dgh | o | FS | F | PVC | Black | 8 | FFPB-O | 250 |
| dgh | e | FS | M | PVC | Black | 8 | FMPB-E | 636 |
| dgh | w | FS | M | PVC | Black | 9 | FMPB-W | 384 |
| scf | y | FS | F | PVC | Honey | 6 | FFPH-Y | 456 |
| scf | g | FS | M | PVC | Honey | 8 | FMPH-G | 345 |
| scf | g | FS | M | PVC | Honey | 10 | FMPH-G | 657 |
| scf | o | FS | M | TPR | Honey | 7 | FMTH-O | 234 |
| dgh | w | FS | M | PVC | Black | 10 | FMPB-W | 329 |
| fds | e | FS | F | PVC | Nicotine | 5 | FFPN-E | 574 |
| fds | e | FS | F | PVC | Nicotine | 8 | FFPN-E | 245 |
| fds | m | FS | M | PVC | Nicotine | 9 | FMPN-M | 621 |
| fds | w | FS | M | TPR | Nicotine | 10 | FMTN-W | 206 |
| asd | l | MS | F | TPR | Black | 5 | MFTB-L | 881 |
| asd | t | MS | M | TPR | Black | 6 | MMTB-T | 831 |
| asd | t | MS | M | TPR | Black | 10 | MMTB-T | 277 |
| dgh | f | MS | F | TPR | Black | 5 | MFTB-F | 440 |
| dgh | f | MS | F | TPR | Black | 9 | MFTB-F | 321 |
| dgh | n | MS | F | TPR | Black | 8 | MFTB-N | 355 |
| dgh | n | MS | F | TPR | Black | 8 | MFTB-N | 255 |
| dgh | x | MS | M | PVC | Black | 6 | MMPB-X | 788 |
| wer | p | MS | F | TPR | Nicotine | 8 | MFTN-P | 657 |
| wer | p | MS | F | TPR | Nicotine | 10 | MFTN-P | 234 |
| xcv | h | MS | F | PVC | Black | 5 | MFPB-H | 329 |
| xcv | z | MS | F | PVC | Black | 6 | MFPB-Z | 574 |
| fgh | l | MS | F | PVC | Honey | 7 | MFPH-L | 116 |
| fgh | j | MS | F | PVC | Nicotine | 6 | MFPN-J | 432 |
| fgh | v | MS | F | TPR | Honey | 6 | MFTH-V | 354 |
| fgh | r | MS | F | TPR | Nicotine | 8 | MFTN-R | 230 |



Go to Family Formation
Decisions

Figure 4-2: Grouping Jobs into Super-families

For the purpose of this thesis, it has been assumed here that the processing time for Full Shot shoes are as shown in Table 3-2.

Table 4-3: Calculation of processing times

| Mold Type | Sex | Auto Key | Size | Demand | Processing Time (in minutes) | Total Processing Time (in minutes) |
|------------------|------------|-----------------|-------------|---------------|-------------------------------------|---|
| FS | F | FFTB-K | 6 | 1614 | 0.216 | 1614 * 0.216 = 347.85 |
| FS | M | FMPB-U | 8 | 4128 | 0.290 | 4128 * 0.290 = 199.52 |
| FS | M | FMTB-C | 5.5 | 6270 | 0.224 | 6270 * 0.224 = 1404.5 |
| MS | M | MMTB-C | 7 | 1248 | 0.281 | 1248 * 0.281 = 350.7 |
| MS | F | MFTB-L | 5 | 5286 | 0.109 | 5286 * 0.109 = 576.2 |

4.1.4 Calculation of the number of cells of each type

Once the shoes are separated into the two super-families, the total loads due to each of the two super-families is calculated by summing up the loads of the shoes (T_i) in that group, based on which the number of cells required of each type are calculated. The number of cells is calculated based on the assumption that the total load on the cells (not including setups) cannot be more than MaxCap (where MaxCap is the maximum capacity per cell, as defined by the scheduler). Hence, the number of Full Shot cells (NC_f) and Mid Sole (NC_m) are calculated using the equations,

$$NC_f = \frac{\sum T_i}{MaxCap} \quad i \in Fullshot \quad (4.2)$$

$$NC_m = \frac{\sum T_i}{MaxCap} \quad i \in Midsole \quad (4.3)$$

For example, if the total load due to full shot shoes is 5077 minutes and the load due to the Mid Sole shoes is 3183 minutes, and assuming that the full capacity, MaxCap is 1500 minutes, the number of cells of each type, are calculated as follows:

$$NC_f = \frac{5077}{1500} = \mathbf{3.4}$$

$$NC_m = \frac{3183}{1500} = \mathbf{2.1}$$

It is recommended that a maximum of one shared cell be created as and when required. This decision is based on the following calculation. If the sum of the decimal part of the calculated number of Full Shot cells (in this case, 0.4) and the decimal part of the calculated number of Mid Sole cells (in this case, 0.1) is less than 1 (in this case, it is only 0.5), then one shared cell should be created. If however, this number is greater than 1, then we just add one cell each to Full Shot and the Mid Sole parts with no shared cells. This is just because having two shared cells would be similar to having one extra cell of each type. In fact, this decision relatively reduces setups. Hence, equation (4.2) can be modified as follows:

$$NC_f = Integer\left(\frac{\sum T_i}{MaxCap}\right) \quad i \in Fullshot \quad (4.4)$$

$$NC_m = Integer\left(\frac{\sum T_i}{MaxCap}\right) \quad i \in Midsole \quad (4.5)$$

$$N_{shared} \leq 1 \quad (4.6)$$

In this example, since the decimal parts sum up to just 0.5, one shared cell is created. So, the example given to illustrate equation (4.2) is modified as follows;

$$NC_f = \text{Integer}\left(\frac{5077}{1500}\right) = 3$$

$$NC_m = \text{Integer}\left(\frac{3183}{1500}\right) = 2$$

$$\text{No. of Shared Cells} = 1$$

However, this is just a recommendation and would have no bearing on the methodology described in the following sections. So, the users are free to override the recommendation and to create more than one shared cell if they want to. Also this calculation can be repeated as and when required, typically at the start of every planning period.

4.2 Phase I-B – Family Definition

This phase in the methodology is pre-requisite to some of the loading procedures that will be described in later sections. This means that, in later stages, depending on our choices, the results from this particular stage (i.e., the families formed from the jobs from the orders) can be disregarded. In that case the user would move directly to the cell loading phase from the background demand aggregation phase, thus skipping the family formation phase in the methodology. This phase is inherently based on the product structure represented in Figure 4-3 and is illustrated in Figure 4-4.

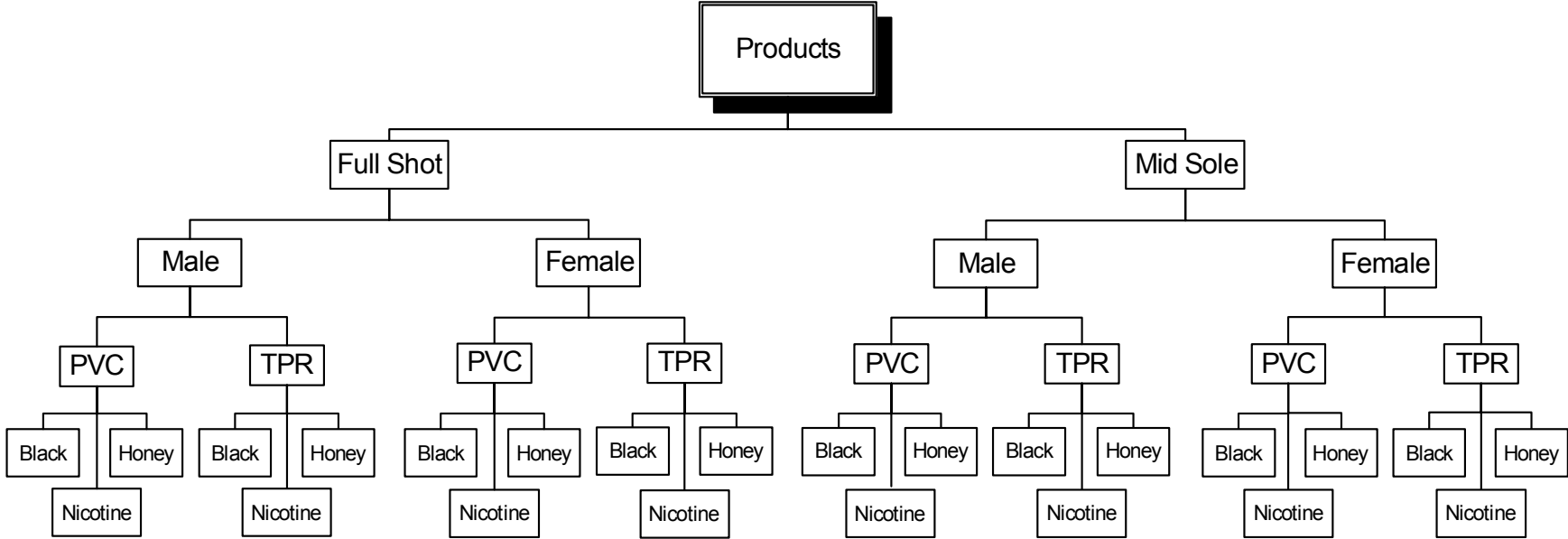
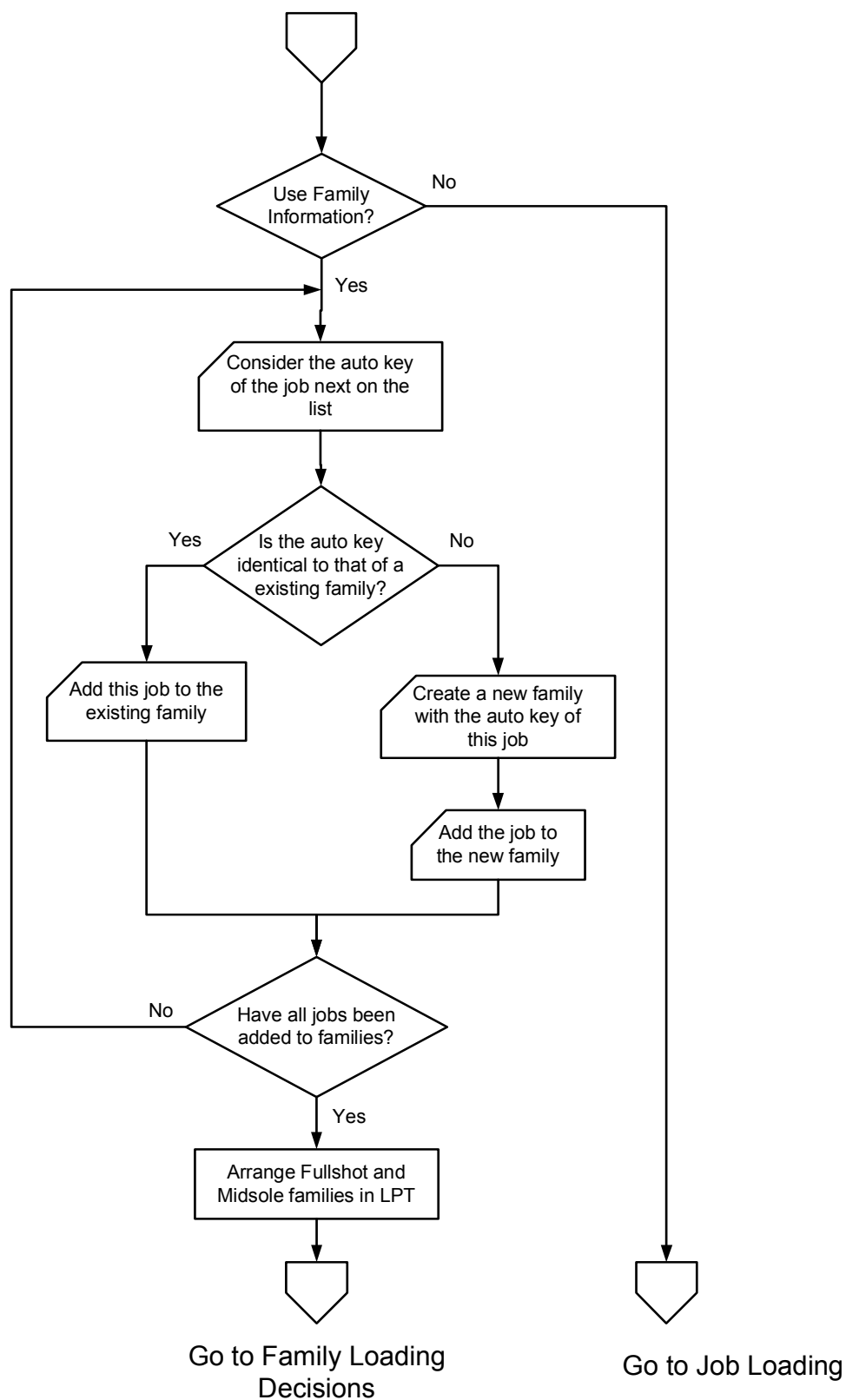


Figure 4-3: Product Structure

In this phase, the shoes from the various orders are grouped into the families based on the key generated (for each type of shoe) for the orders in the background phase. All the shoes with the same key are grouped into the same family. The key is used to name the family. So typical families – let us take for example FMPB-XYZ and FMPB-ABC, would be as shown in Table 4-4. Hence, a particular family is composed of shoes of the same sole type, sex, material and sole color, and the same model (as defined by the model id). So such a family would require molds of different sizes depending on the sizes of the shoes in the family. At the end of this phase, the whole set of jobs is separated into different job families based on whether they are Full Shot or Mid Sole, intended for Men or Women, whether the sole material is to be PVC or TPR, the color being Black (denoted as B in Figure 4-3), Honey (H) or Nicotine (N) and the different model of shoes available.

Table 4-4: Typical examples of families

| Family | Order Id | Model id | Sole | Sex | Material | Color | Size | Demand | Time |
|---------------|-----------------|-----------------|-------------|------------|-----------------|--------------|-------------|---------------|-------------|
| FMPB-Q | A123 | Q | FS | M | PVC | Black | 8 | 400 | 116 |
| | B456 | Q | FS | M | PVC | Black | 7 | 300 | 84.3 |
| | C234 | Q | FS | M | PVC | Black | 9 | 250 | 76.5 |
| | D123 | Q | FS | M | PVC | Black | 5.5 | 220 | 49.4 |
| MFPN-V | FDS | V | MS | F | PVC | Nicotine | 6 | 721 | 132 |
| | FDS | V | MS | F | PVC | Nicotine | 10 | 117 | 39.9 |

**Figure 4-4: Family Definition**

4.2.1 Calculation of family processing times

The next step in the process is to find the processing times for each family. The processing time of family j (Fp_j) is nothing but the sum of the processing times P_i for every shoe in the family as shown in Table 4-5. Hence,

$$Fp_j = \sum_{i \in j} P_i \quad (4.7)$$

Table 4-5: Sample calculation for family processing time

| Family | Size | Demand | Order Id | Model Id | Processing Time (In min.) | Family Processing Time (In min.) |
|---------------|------|--------|----------|----------|---------------------------|--|
| FMTN-Q | 8 | 400 | A123 | Q | 116 | (116 + 84.3 + 76.5 + 49.4) = 326.2 |
| | 7 | 300 | B456 | Q | 84.3 | |
| | 9 | 250 | C234 | Q | 76.5 | |
| | 5 | 220 | D123 | Q | 49.4 | |

4.2.2 Arranging Families in LPT

The next step in the process is to arrange the families in the decreasing order of their processing times, i.e., according to LPT. This would help us achieve one of our objectives - dividing the load evenly across the cells. An example of typical family data would be as shown in Table 4-6.

Table 4-6: Typical family data arranged in LPT

| Family | Family Processing Time (in minutes) | Family | Family Processing Time (in minutes) |
|--------|--|--------|--|
| FMPB-Q | 420.5 | MMTB-T | 208.5 |
| FMPH-G | 406 | MMTH-D | 200.2 |
| FMTN-Y | 387.5 | MFTN-R | 196.2 |
| FMPN-A | 330 | MFPB-Z | 185.7 |
| FFTH-U | 328 | MFTH-X | 184.5 |
| FMTB-C | 292.5 | MFTH-B | 182.2 |
| FMPH-S | 284.5 | MFPN-V | 171.8 |
| FMTH-O | 281 | MFTB-N | 159.3 |
| FMPB-W | 230 | MFTN-P | 159.3 |
| FMTH-M | 210 | MFTB-F | 145.8 |
| FMPB-U | 199.5 | MMPN-N | 143.3 |
| FMPN-M | 190 | MFTN-H | 133.5 |
| FMPB-E | 184.5 | MFPN-J | 131.7 |
| FMPH-K | 177.5 | MMTH-J | 131.2 |
| FFPN-E | 164 | MMPB-X | 126.0 |
| FMTB-I | 156.5 | MFTH-F | 119.3 |
| FFPH-Y | 153.5 | MFTH-T | 119.3 |
| FMPN-S | 142 | MFTH-V | 108.0 |
| FMTN-G | 124 | MFTB-L | 96.0 |
| FMTN-Q | 109 | MFPB-H | 82.2 |
| FMPN-I | 101 | MFTN-D | 82.2 |
| FFPB-O | 77 | MFTH-P | 64.2 |
| FMTN-W | 70.5 | MFTB-R | 52.8 |

4.3 Phase II – Cell Loading

The families are ready to be loaded into the cells now. The cells are usually loaded on the basis of “Minimum Load” – the family to be loaded is loaded to the cell (of its own type) that has the minimum load. The key condition in this case, however, would be to ensure that family integrity is preserved. By this, we mean that shoes (i.e., jobs) from one family will not be loaded until the family being processed has been processed completely. In fact, breaking the family integrity is not possible in most cases because the color or the material varies from one family to the next succeeding one and then the tank holding the material for the molds has to be emptied and cleaned to remove any trace of the material and its coloring pigments.

In the cell loading phase, the Full Shot and the Mid Sole cells are loaded first. If a particular cell does not have enough capacity to process a particular family that had to be loaded next, then the question of splitting families (also known as lot splitting) arises. If the splitting of families is to be allowed then the “Minimum Load” basis is abandoned and cells are loaded sequentially – each cell is loaded to the maximum before proceeding to the next cell. This decision would ultimately have bearing on the even distribution of loads. Hence, the different cell loading strategies used during the cell loading phase (as shown in Figure 4-5) are discussed in the following sections and illustrated with the help of examples and flowcharts. These strategies are:

1. Do not split families.
2. Lot Splitting allowed as and when required.
3. Split Families only when load is greater than MaxCap.
4. Load Jobs into cells and then form families in each cell independently.

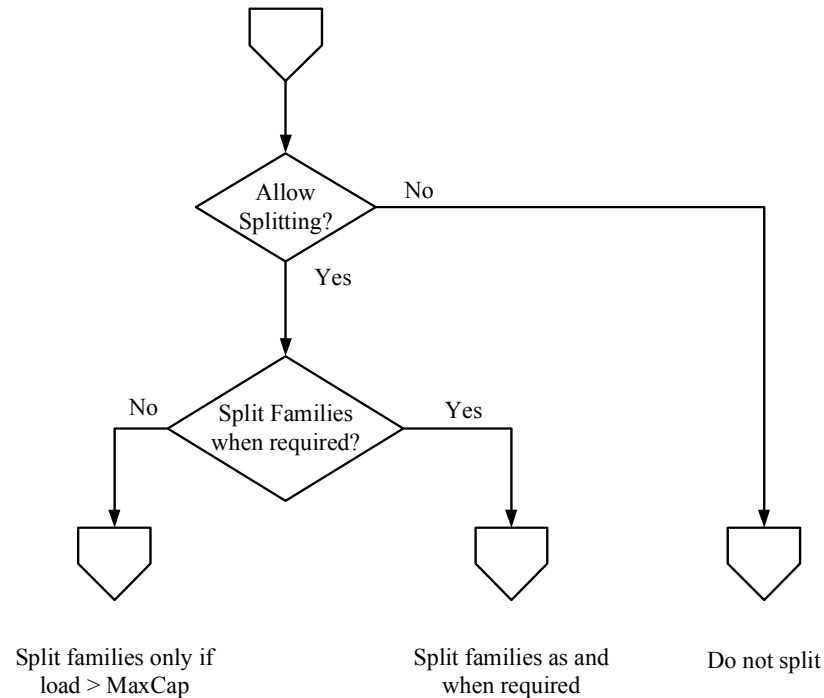


Figure 4-5: Family Loading Decisions

4.3.1 Do not Split Families:

The first strategy involves using the families formed in the previous stage and loading them into the cells as per their sole type and load. In this strategy, the families will not be split, regardless of the load. This means that the families whose loads are lesser than MaxCap would be considered for loading first. If the load due to any individual family is greater than MaxCap, then that family will not be loaded at all. This strategy is illustrated in Figure 4-6.

The first group of families – let us say the Full Shot families from Table 4-6, are considered. We have already determined, from the super-family definition phase, the number of Full Shot cells, NC_f , and Mid Sole cells NC_m , has been calculated to be 3 and 2, respectively. The families are now ready to be loaded on the cells.

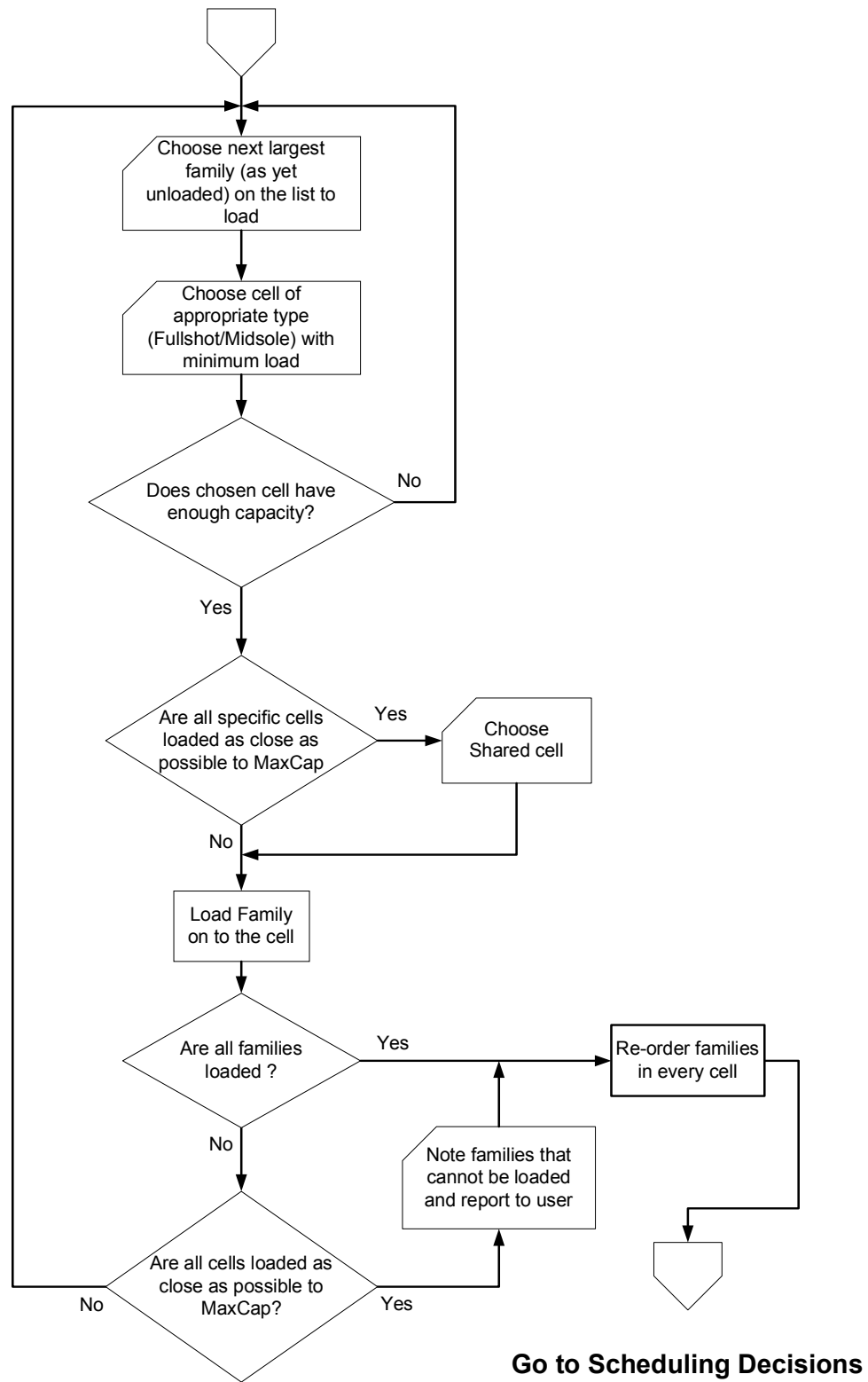


Figure 4-6: Do not Split Families

At time $T_{\text{now}} = 0$, the load on all the cells is 0. Assuming that the load due to the first family in the list is lesser than MaxCap , the family (which has the highest load) is loaded on to the first Full Shot cell, i.e., FS1. The next family is loaded on to FS2, the third to FS3 until the first N_f families on the list are loaded into the NCf cells. In this example, the first family on the list (Table 4-6) is FMPB-Q. This family, whose total processing time is 420.5 minutes, is loaded into FS1. The next family on the list, FMPH-G is loaded to FS2 and the third family to FS3 (Figure 4-7). If the loads due to first family and any succeeding individual families are greater than MaxCap , then these families are not loaded at all (because of insufficient capacity and because splitting is not allowed).

| | |
|------|---------------|
| FS 1 | FMPB-Q, 420.5 |
| FS 2 | FMPH-G, 406 |
| FS 3 | FMTN-Y, 387.5 |

Figure 4-7: Gantt Chart at $T = 0$ minutes

Once all the cells have one family loaded on them, the next family (i.e., the N_f+1 th family) on the list is considered. The next T_{now} would be the lowest processing time among the families already loaded. The loads on all the cells are considered and the N_f+1 th family is loaded onto the cell with the minimum load (i.e., the cell with the earliest finishing time). In our example, looking at the three families loaded on to cells FS1, FS2 and FS3 respectively, we find that the family that has been loaded on FS3 (i.e., FMTN-Y) gets completed first. So, the next family on the list (FMPN-A) is loaded on

FS3, i.e., the cell with the minimum load at the present. The fifth family in the list, FFTH-U, will then be loaded on to FS2 which will be the next cell that completes its processing.

This process continues until all the Full Shot cells have been loaded to their maximum capacity (i.e., MaxCap) possible without splitting lots. At this point, if any of the families have still not been loaded, they are set aside to be loaded into the Shared cell later. Once all the Full Shot cells have been loaded, the Mid Sole families are taken into consideration and are loaded to the Mid Sole cells. The same procedure that was followed for the Full Shot families is followed for the Mid Sole families as well. Once the Mid Sole cells are loaded to their maximum capacity, the remaining Mid Sole families that have not been loaded to the cells are set aside for loading into the Shared cell(s).

The Full Shot and the Mid Sole families that still remain are now ready to be loaded on the Shared Cell, Sh1. The Full shot families that have not yet been loaded are arranged in LPT. Once they are rearranged, they are loaded one after the other (starting with the family with the largest load) into the Shared cell(s). In our example, after the full shot cells have been loaded completely, the families still left behind are FMTB-I, FFPH-Y, FMPN-S, FFPB-O, FMTN-W, FFTB-K. These families are first rearranged in the decreasing order of their loads and are then loaded into the Shared Cell, Sh1. This is repeated in case of the Mid Sole families too.

At this point some of the families in every cell share the same color and material combination. A reduction in set-ups can be achieved if the families in every cell are regrouped to make sure the families with the same color-material combination are loaded

successively and so the families in every cell are regrouped. The Gantt chart that results when this procedure is applied to our example set is shown in Figure 4-8.

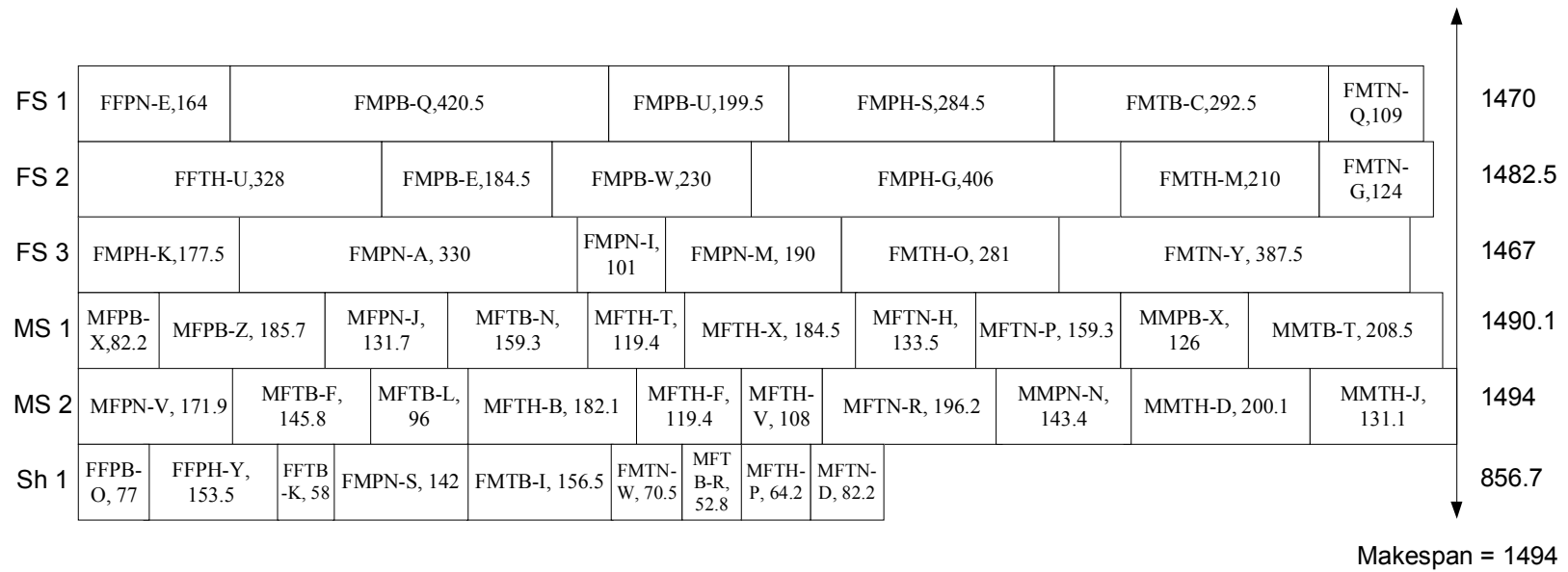


Figure 4-8: Final Gantt Chart - Do not Split Families

If there are any families whose durations are greater than the maximum load left on the Shared Cell, the user is informed that these families cannot be loaded unless they are split. The user is then advised to try alternate options, i.e., to allow lot splitting or to ignore these families/jobs. Another alternative available would be to increase the number of cells and re-run the loading algorithm.

4.3.2 Lot splitting allowed as and when required

The second strategy involves splitting lots (families) as and when required. This strategy is expected to result in the better utilization of the specific cells, with lower idle times. This strategy is illustrated in Figure 4-10.

At $T_{now} = 0$, the first group of families – let us say the Full Shot families (shown in Table 4-6), are considered. Let the number of Full Shot cells be N_{cf} , which in our example would be 3. At this point in time, the load on all the cells is zero. Assuming that the load due to each single family is less than $MaxCap$, the each of the first N_f Full Shot families are loaded successively into the first Full Shot cell FS1, until the cell is loaded to the maximum extent possible. If the load due to any of the first N_f families is greater than $MaxCap$, then the family is split.

So families FMPB-Q, FMPH-G and FMTN-Y (their cumulative load being 1214) are loaded in the same sequence as mentioned into cell FS1 (Figure 4-9). At this point, their cumulative load is still less than $MaxCap$ ($MaxCap$ being 1500) and so the next family in the list (FMPN-A) is also considered. But the load due to family FMPN-A is 330 which is more than the capacity still available on FS1. Hence this family may be split if possible so that the 286 minutes still left on FS1 can be utilized.

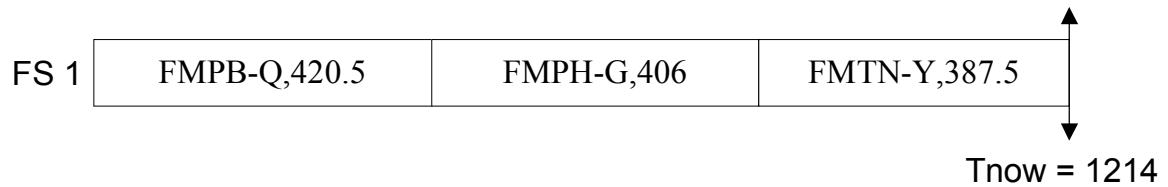


Figure 4-9: Gantt Chart at Tnow = 1214 minutes

Splitting is performed when the available capacity of the cell to be loaded, in this case, FS1, though considerable, is not enough to load the next family in the list. The process of splitting is done carefully at the family level alone and NOT at the shoe level. This means that after splitting, the two parts will comprise of complete job orders. Splitting is performed based on the fulfillment of several conditions.

These conditions and the inherent motivations are:

1. Number of shoes (i.e., individual jobs) in the family that is split should be greater than 1.
2. The demand due to the at least one of the orders in the family should be lesser than or equal to the time remaining in the cell.

The first condition is used to prevent dummy splits and to prevent the creation of dummy families comprising of zero jobs. This clause is purely motivated by trends noticed on studying previous results from the algorithm gathered before introducing this clause. Otherwise the algorithm would create a dummy family that does not have shoes and whose processing time is zero. The second condition ensures that splitting should serve its purpose.

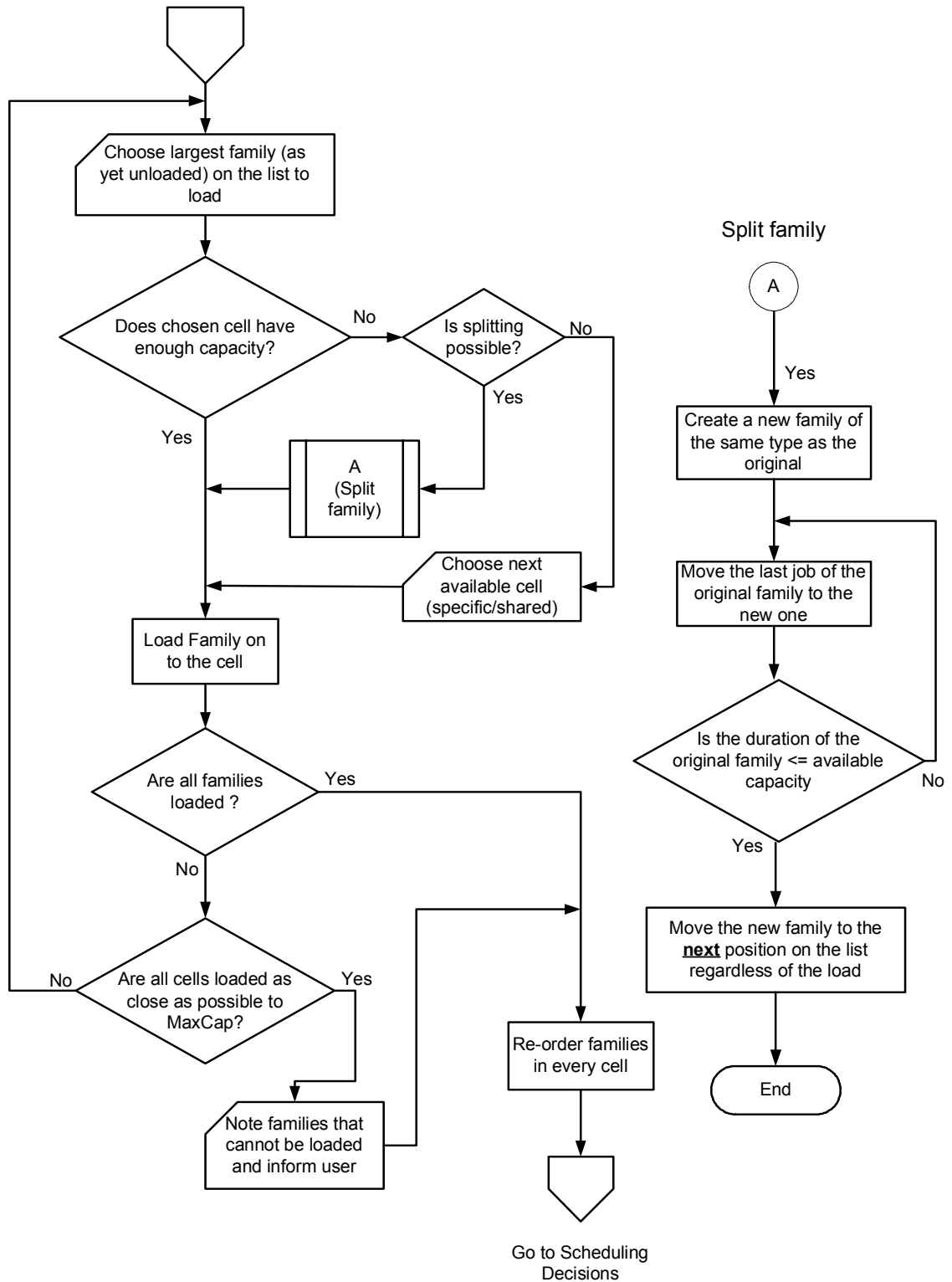


Figure 4-10: Split Families as and when required

For example, if a family is comprised of 3 jobs whose processing times are each greater than 100 minutes and if the capacity available is less than 100 minutes, splitting is not going to serve any purpose. Using this clause, this condition will be prevented and such families will not be subject to any split. After checking for these conditions, the family is split. First a family with identical key and details as the original family is created. Now keeping track of the time remaining in the cell, the shoes with the lowest durations are moved to the new family until the original family can be loaded.

The total load due to FMPN-A is 330.1 minutes. The next step in the procedure is to check whether a split in the family would be acceptable. The family FMPN-A is shown in Table 4-7. There are two jobs in this family and the processing times of both these jobs are lesser than the capacity left over in FS1. So this family can be split since it satisfies both of the clauses specified.

Table 4-7: Family FMPN-A

| Family | Size | Demand | Injection Time | Job Processing Time | Family Processing Time |
|---------------|-------------|---------------|-----------------------|----------------------------|-------------------------------|
| FMPN-A | 9 | 672 | 0.306 | 205.6 | $205.6 + 124.5 = 330.1$ |
| | 5.5 | 556 | 0.224 | 124.5 | |

Hence, a new family FMPN-A_1 (with all the same characteristics as FMPN-A) is created (see Table 4-8) and the second job in the original family is moved to the new family. So the load due to FMPN-A is now reduced to 205.6 minutes and can now be loaded on to FS1. The new family FMPN-A_1 is loaded to the next cell since FS1 has been loaded to the maximum extent possible. This is as per McNaughton's algorithm for

lot splitting in parallel machine scheduling [1]. Hence the new family FMPN-A1 is loaded as the first family (Figure 4-11) to be processed on cell FS2 at time $T_{now} = 0$.

Table 4-8: Families after split

| Family | Size | Demand | Injection Time | Family Processing Time |
|----------|------|--------|----------------|------------------------|
| FMPN-A | 9 | 672 | 0.306 | 205.6 |
| FMPN-A_1 | 5.5 | 556 | 0.224 | 124.5 |

Once the new family has been loaded, the families on the original list are loaded successively into this cell, until the cell is loaded to the maximum extent possible or if a family has to be split. The process is repeated until all the Full Shot cells have been loaded to the maximum extent possible. Any family that has not been loaded yet is moved to the Shared list. These unloaded families can be loaded into the Shared cell later.

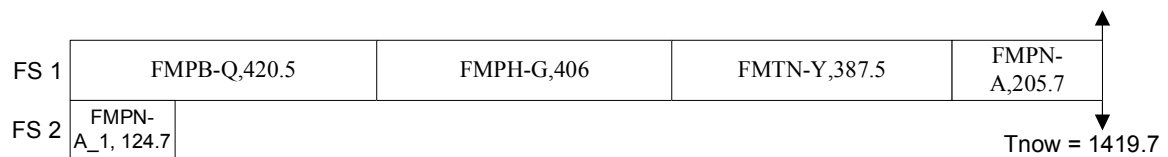


Figure 4-11: Gantt Chart after splitting and loading new family

Next, the Mid Sole families and the respective cells are considered. The whole process is repeated with these Mid Sole families. After loading the Mid Sole cells, if there are Mid Sole families remaining, they are moved to the Shared list. A Shared cell, Sh1, is now created. First the Full Shot families on the Shared list are loaded on to Sh1.

Then the Mid Sole families in the list are loaded. Since only one shared cell exists, splitting is not allowed and families are loaded as a whole until no more loading is possible. Now since all the cells are scheduled to the maximum extent possible, if there are still any more families that have still not been loaded, then the user is informed of this development and allowed to make the decision on whether to increase capacity by exercising his options such as overtime, process them during the next planning period, i.e. during the next week.

At this point some of the families in every cell share the same color and material combination. A reduction in set-ups can be achieved if the families in every cell are regrouped to make sure the families with the same color-material combination are loaded successively and so the families in every cell are regrouped. The final Gantt chart that results when this procedure is applied to our example is as shown in Figure 4-12.

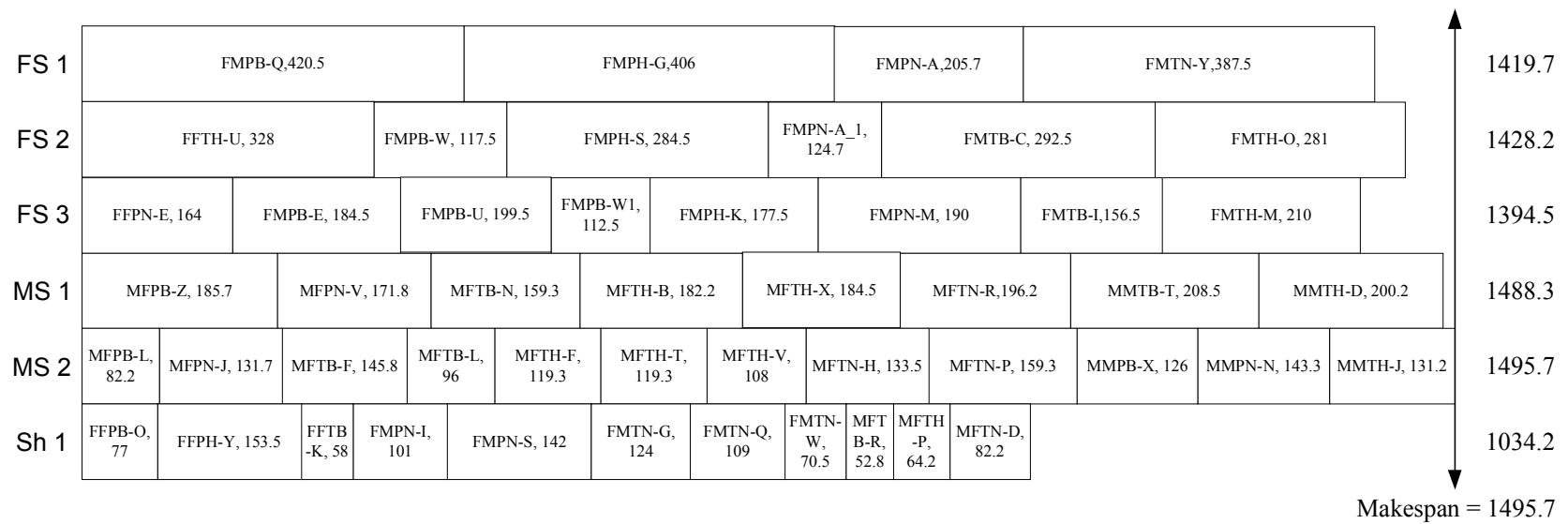


Figure 4-12: Final Gantt Chart - Split As and When Required

4.3.3 *Split families only when load is greater than MaxCap*

The third strategy entails the splitting of families only when absolutely required and the families cannot be loaded in any cell at all, i.e. when the load due to the family is greater than MaxCap. This strategy is illustrated in Figure 4-13 and is demonstrated using the example from Table 4-9.

The first group of families – let us say the Full Shot families, are considered. Let the number of Full Shot cells be NC_f . At time zero, the load on all the cells is zero. Since the families are arranged in LPT, the jobs with the maximum loads are on the top of the list and have to be loaded first. The duration of the first family on the list is compared against MaxCap. If the load due to the family is greater than MaxCap then the family is split.

In our example, the family with the largest load is FMPB-Q (1902). Assuming that MaxCap is 1500 minutes, this load is far greater than MaxCap and hence this family would have to be split. This family consists of two different jobs whose load is 1481.5 and 420.5 respectively. The process of splitting is performed as described in the previous section. The split is performed at the point at which the load (comprising of a set of complete jobs in the family) will result in a cumulative load that is either equal to or just lesser than MaxCap. Hence in our case, the job with load of 1901.5 minutes will be moved to a new family and the original family now has only 1481.5 minutes and so can be loaded into one cell. So FMPB-Q is split into FMPB-Q (1481.5 minutes) and FMPB-Q1 (420.5 minutes). Now once the original family has been split into two parts, the new family that has been formed is moved to end of the list (for the moment). So FMPB-Q1 is moved to the end of the list.

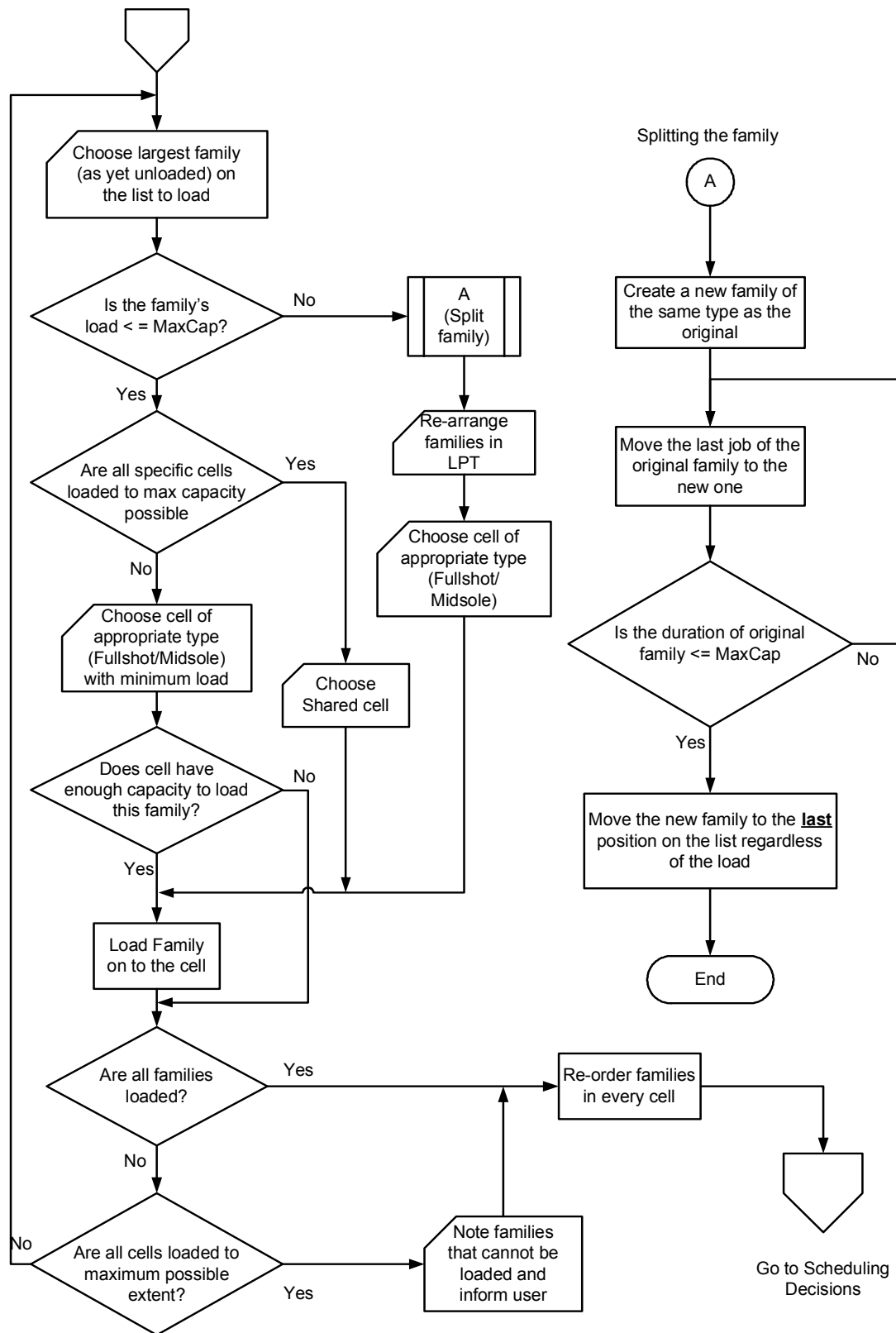


Figure 4-13: Split Families only if load > MaxCap

Table 4-9: Another Example

| Family | Family Processing Time | Family | Family Processing Time |
|--------|------------------------|--------|------------------------|
| FMPB-Q | 1902 | MMTB-T | 208.5 |
| FMPH-G | 406 | MMTH-D | 200.2 |
| FMTN-Y | 387.5 | MFTN-R | 196.2 |
| FMPN-A | 330 | MFPB-Z | 185.7 |
| FFTH-U | 328 | MFTH-X | 184.5 |
| FMTB-C | 292.5 | MFTH-B | 182.2 |
| FMPH-S | 284.5 | MFPN-V | 171.8 |
| FMTH-O | 281 | MFTB-N | 159.3 |
| FMPB-W | 230 | MFTN-P | 159.3 |
| FMTH-M | 210 | MFTB-F | 145.8 |
| FMPB-U | 199.5 | MMPN-N | 143.3 |
| FMPN-M | 190 | MFTN-H | 133.5 |
| FMPB-E | 184.5 | MFPN-J | 131.7 |
| FMPH-K | 177.5 | MMTH-J | 131.2 |
| FFPN-E | 164 | MMPB-X | 126.0 |
| FMTB-I | 156.5 | MFTH-F | 119.3 |
| FFPH-Y | 153.5 | MFTH-T | 119.3 |
| FMPN-S | 142 | MFTH-V | 108.0 |
| FMTN-G | 124 | MFTB-L | 96.0 |
| FMTN-Q | 109 | MFPB-H | 82.2 |
| FMPN-I | 101 | MFTN-D | 82.2 |
| FFPB-O | 77 | MFTH-P | 64.2 |
| FMTN-W | 70.5 | MFTB-R | 52.8 |

Now the next family in the list is considered. If it needs to be split (i.e., if the load is greater than MaxCap), then it is split. This process continues until all the jobs whose loads are greater than MaxCap are split. However, if the load is lesser than MaxCap then splitting can be abandoned and we can proceed with loading. But before the loading can begin, the families are re-arranged in LPT (i.e., in the decreasing order of their processing times). This is to facilitate the moving of the families created as a result of splitting to their proper positions on the list based on their load. Hence in our example, the families would be rearranged and FMPB-Q1 will be the second family on the list since its load is higher than all but one family on the list. Once all the families that need to be split have been split, the loading process starts.

At time $T_{\text{now}} = 0$, the load on all the cells is 0. So, the family with the highest load (i.e., the first family on the list) is loaded on to the first Full Shot cell, i.e., Full Shot1. The next family on the list is loaded to Full Shot2. The first N_f families on the list (i.e., the families with the highest loads) are loaded into one cell each.

In our example, the number of full shot cells required is 4. Hence the first four families on the list, i.e. FMPB-Q, FMPB-Q1, FMPH-G and FMTN-Y are loaded into FS1, FS2, FS3 and FS4 respectively at time $T_{\text{now}} = 0$ (as shown in Figure 4-14). Once all the cells have one family loaded on them, the next family (i.e., the $N_f + 1$ th family) on the list is considered. The next T_{now} would be the lowest processing time among the families already loaded. The loads on all the cells are considered and the $N_f + 1$ th family is loaded onto the cell with the minimum load (i.e., the cell with the earliest finishing time). Hence, family FMPN-A would be loaded on to FS4 in this case.

The next family on the list is loaded to the cell with the minimum load at the moment after the last job has been loaded. If a cell does not have enough capacity to process a particular family then the family is moved in a separate Shared list. The families on this list would be loaded to the Shared cell once the Mid Sole families have been loaded into their cells. The next family is considered and loaded.

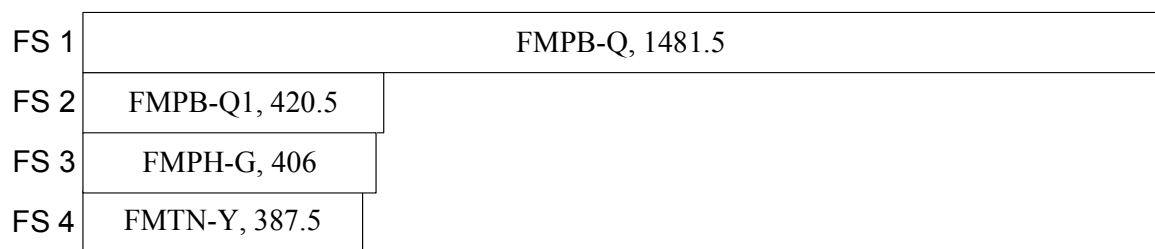


Figure 4-14: MaxCap Gantt Chart at Tnow = 0

This process continues until all the families in the Full Shot list are loaded into the Full Shot cells to the maximum extent possible. Once all the Full Shot cells have been loaded to the maximum feasible level or when all the Full Shot families in the list have been loaded, the focus shifts to the Mid Sole families. The whole process is repeated with the Mid Sole families until they have been loaded (after splitting them if needed) to their cells. In case any of the Mid Sole families cannot be loaded into the cells, these families are added to the Shared list behind the Full Shot families that are already on it.

The families on the Shared list have to be loaded now. First a Shared cell is created. Then the Full Shot families on the shared list are loaded on to the cell. Now since there can be no more splitting of families, the family is loaded only if the load due to the family is less than MaxCap. Once all the feasible Full Shot families are loaded into the Shared cell, the remaining Mid Sole families are loaded one by one on to the cell. This

process continues until all feasible families on the shared list have been loaded into the cells. Now once all the cells are scheduled to the maximum extent possible, if there are still any more families that have still not been loaded, then the user is informed of this development. The user can then decide how to actually proceed with this set of shoes.

At this point some of the families in every cell share the same color and material combination. A reduction in set-ups can be achieved if the families in every cell are regrouped to make sure the families with the same color-material combination are loaded successively and so the families in every cell are regrouped. The final Gantt chart for our example is as shown in Figure 4-15.

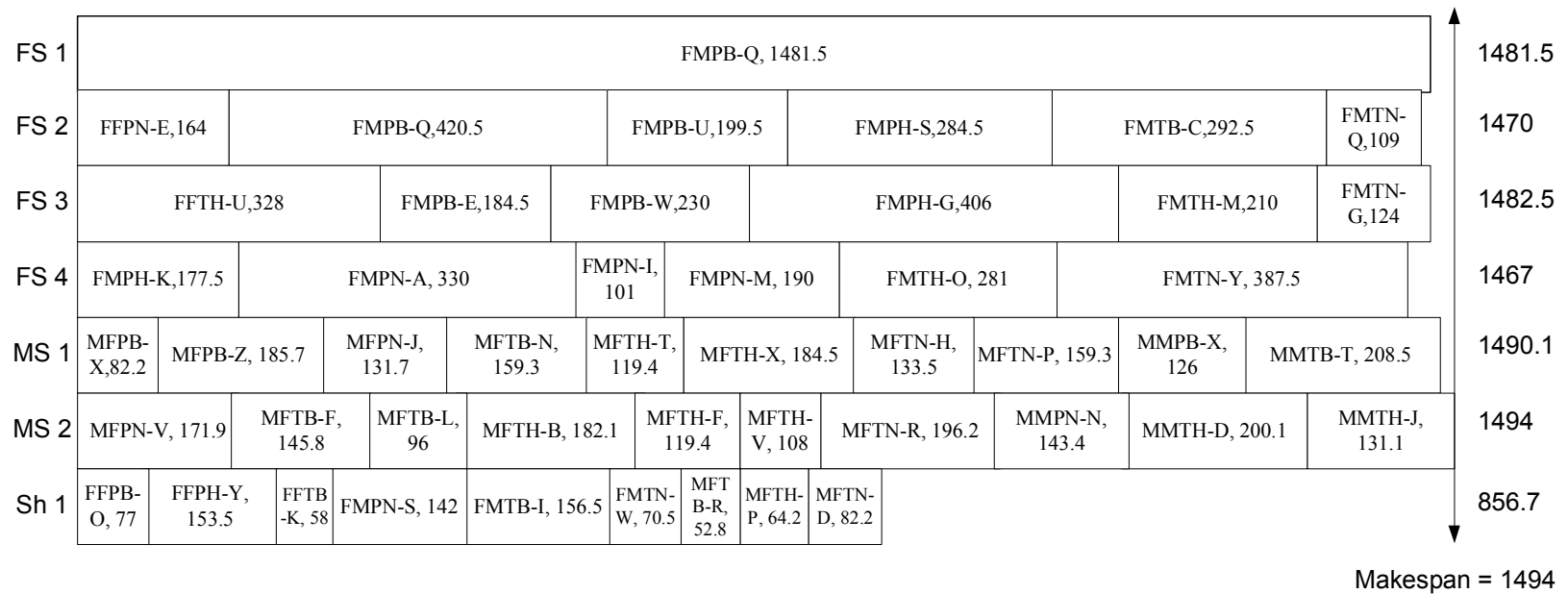


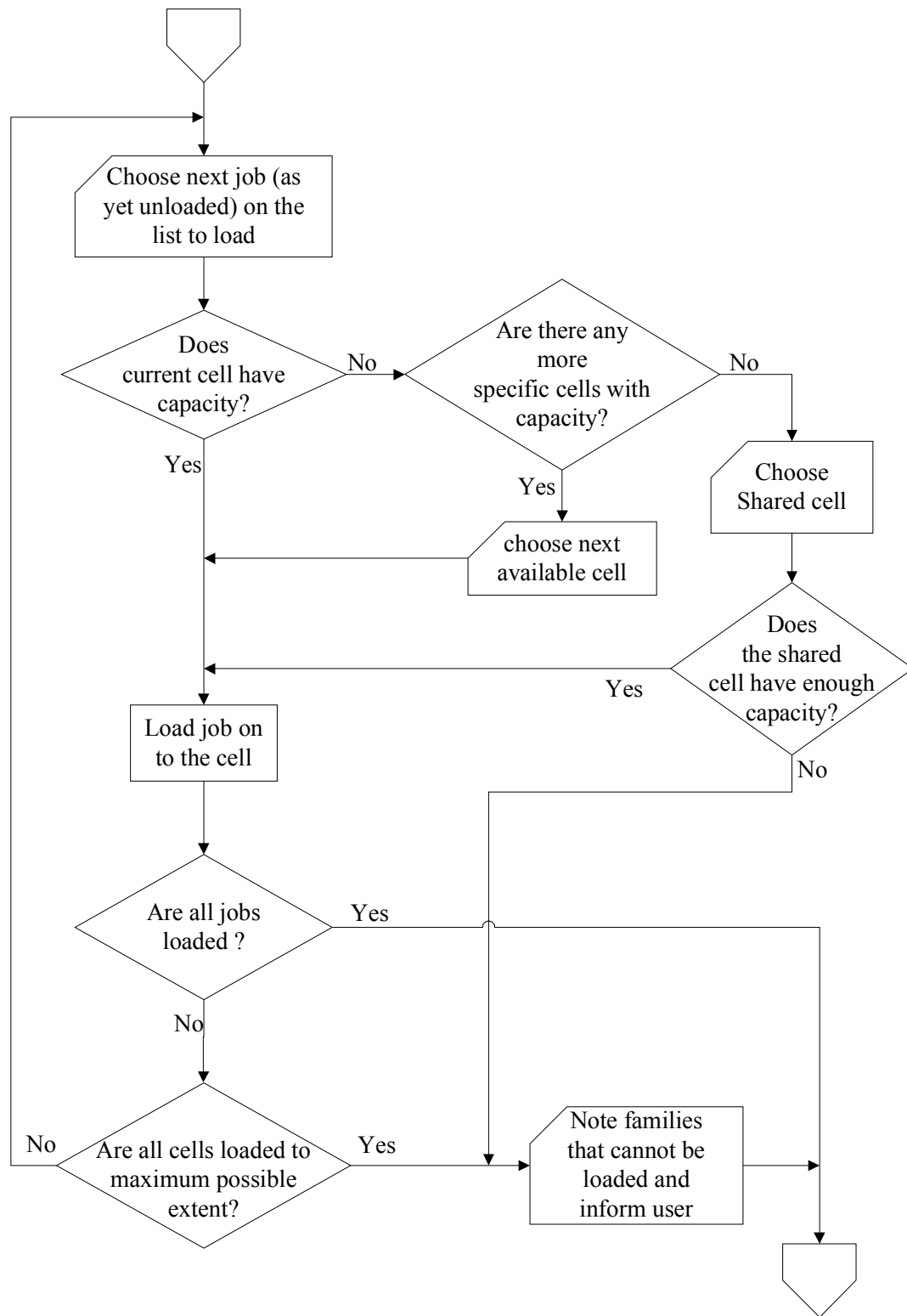
Figure 4-15: Final Gantt Chart - Split Families only if load > MaxCap

4.3.4 Load Jobs First Into the Cells and Then Form Families Independently

This procedure differs from the other procedures because the loading is on the job level itself. Hence, this strategy ignores the families formed in the previous phase and the original jobs are considered. This strategy is used as a highlight the inefficiencies when families are not formed and to test and emphasize the positives inherent to the family formation process in cellular manufacturing systems.

This strategy is applied after the jobs are initially separated into the Full Shot and Mid Sole groups. They are however, NOT re-ordered in any manner whatsoever. This means that the jobs would be in the same order as they arrive. After this separation, the jobs are loaded into the cells. The jobs are to be loaded as is – i.e. in the same order as they arrive. Families are not formed until the jobs have been loaded into the cells. This strategy is illustrated in Figure 4-16 and is similar to the “Do not Split Families” strategy already discussed earlier in this chapter, except that instead of families we deal with the actual jobs (in the first stage) and hence no splitting is allowed.

Let us first consider the Full Shot group of jobs. At time $t = 0$, all the cells are empty. So the first N_f Full Shot jobs can be loaded into the n_f Full Shot cells. The first job is loaded to cell FS1, the second job to FS2 until the first n_f jobs have been loaded on to the N_{Cf} Full Shot cells. Once these jobs are loaded, the cell with the earliest finishing time (i.e., lowest load) is chosen and the $(N_f + 1)$ th job is loaded to this cell. The $(N_f + 2)$ th job is loaded to the cell with the next earliest finishing time (i.e., the cell with the next higher load) among the currently loaded families. This process continues until all the cells have been loaded to the maximum capacity possible.



Go to Job Regrouping

Figure 4-16: Job Loading

The same procedure is performed for the Mid Sole jobs too. Since we do not allow splitting of lots at the job level, if there are any jobs left after the specific cells have been loaded to the maximum extent possible, Shared cell, Sh1, is created. The remaining Full Shot jobs are first loaded into the Shared cell. Then the Mid Sole jobs left are also loaded into the Shared Cell. If, after this exercise there are still jobs to be loaded, the user has to be informed of the fact that the system possesses insufficient capacity. To counter this situation, the user can either increase the capacity by opting for over time or simply delay these jobs until the next planning period. The Gantt chart that will result when the first stage of this procedure is applied to our example data is represented in Figure 4-17.

Once all the jobs have been loaded to the cells, these jobs are regrouped to form families within their own cell (Full Shot/Mid Sole/Shared). All the shoes with the same keys are grouped into the same family. For example, in cell Full Shot1, all the Full Shot shoes for the Male sex in model X that require PVC sole in Black color are grouped together and so on. The key, comprising of the first letters of the various deciding properties of the shoe, i.e., Sole type, Sex, Material, Color and Model is used to name the family.

At the end of this exercise, if two families have the both the same sole color and sole material, they will be loaded one after the other on the cell. Such a grouping will help reducing setups by reducing (as already mentioned in earlier chapters) the number of mold and tank cleanings between families. This process is repeated to all the cells until families have been formed in every cell. The Gantt Chart at the end of this stage is represented in Figure 4-18.

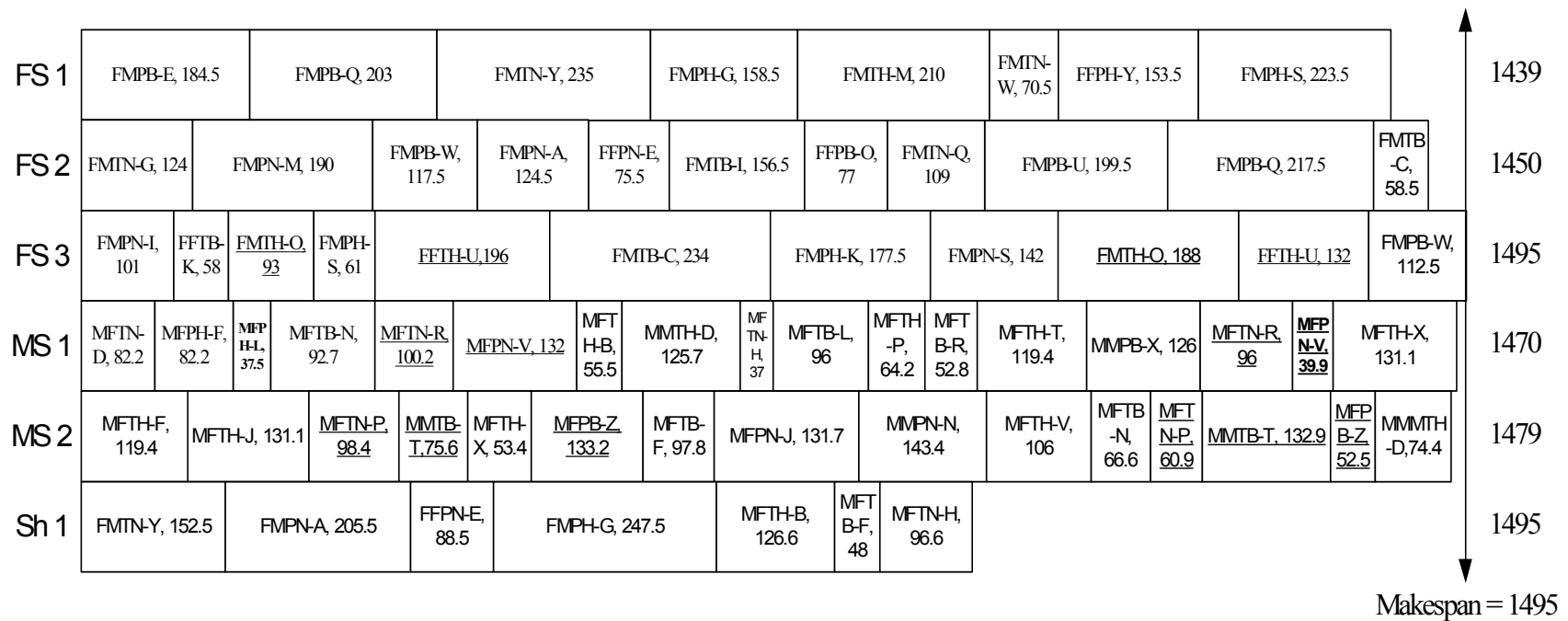


Figure 4-17: Gantt Chart - Load Jobs: Stage I

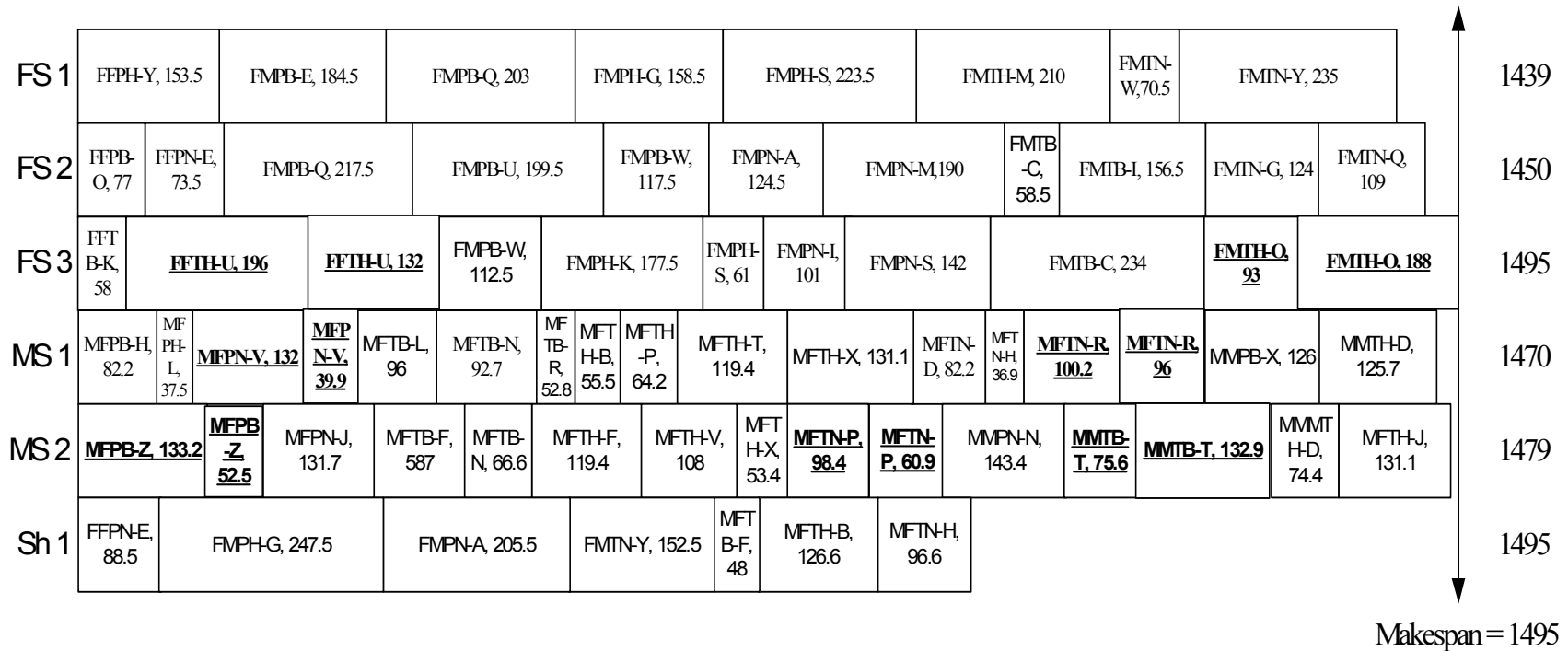
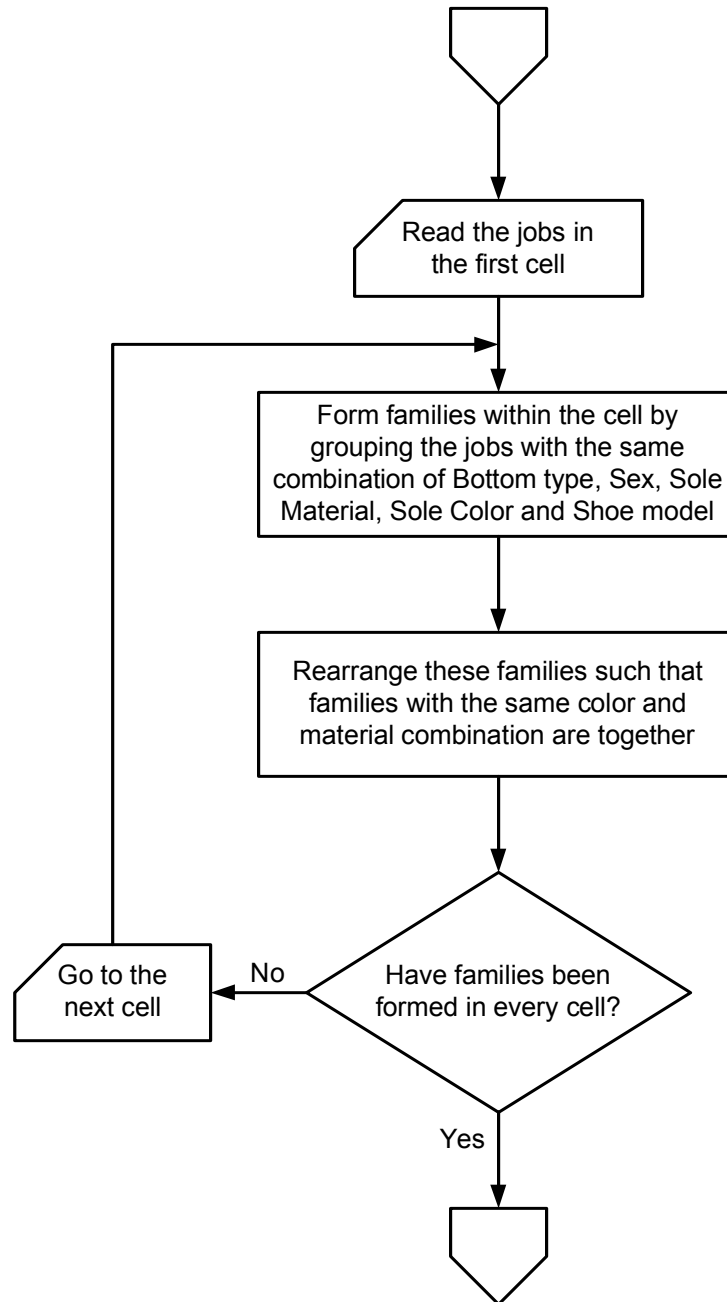


Figure 4-18: Final Gantt Chart - Load Jobs: Stage II (After Re-ordering)



Go to Scheduling Decisions

Figure 4-19: Job Re-grouping after Loading

Looking at Figure 4-17 and Figure 4-18, the difference will not be apparent till we observe carefully. In Figure 4-17, some of the jobs on the Gantt Chart have been underlined to emphasize that they are similar, but at loaded at different points in the same cell. Now the next stage is mainly used to group these similar jobs together. So after re-ordering (Figure 4-18), these jobs are moved right next to each other, forming families. This regrouping also brings together other shoes which are of the same bottom type, sex, material and color, but of different models. This obviously reduces the setup even more.

4.4 Scheduling of Families and Jobs

Once the initial loading sequence of families is determined for each cell, the next step in the process is to attach the start times and end times for every family / job in the cells. This is where the molds enter into the equation. The molds used in the injection molding process have to be cleaned each and every time the material or the coloring pigment used changes and transferred to the cell where it is needed next. The tank in the injection molding machine also has to be cleaned every time the material or color changes. These times contribute to the setup times in the cells.

The family formation process that has been applied, aids the reduction of the number of setups that we need to perform while working towards the fitting of the loads in the available capacity. This is because the combination of the times taken to change color or material in the tank and clean it is the setup time for each family (since the color and the material change only when making the change from one family to another) in the cell. It is assumed that regardless of the change in color or material, the total time required to prepare the tank (cleaning and refilling), remains constant at 20 minutes.

Another assumption that has been made is that the first family processed in every cell has zero setup, i.e., the molds required by the family are available at time $t = 0$ and the tank in the injection molding machine has been cleaned and refilled in time for the first family. So at time zero, the first family in the list for every cell is loaded and the processing begins. We will attempt to explain the scheduling methodology with the same simple example that was used to illustrate the loading heuristics. It has to be noted that regardless of the heuristic used for loading, the scheduling methodology is going to be unchanged for all the four different loading heuristics.

Let us consider the results of the loading process shown in Figure 4-8, which is the load Gantt chart after the application of the “Do Not Split” heuristic. Now in this case the first family loaded in cell FullShot-1 is FFPN-E whose two jobs have a cumulative load of 164 minutes. Now the Rotary machine has 6 pairs of positions and hence six pairs of shoes can be processed at any given time after all positions have been loaded.

The processing time is calculated by first estimating the number of turns required to process every job. Since the job will be processed simultaneously on all six positions of the Rotary Machine, every single complete turn of the machine will result in the output of six pairs of shoes. Assuming that the demand for the i th job is D_i , the number of turns required to process that particular set of shoes (Nt_i) can be calculated as follows.

$$Nt_i = \frac{D_i}{6} \quad (4.8)$$

We assume that any instant shoes of the same size only will be loaded on the Rotary Injection Molding Machine. This means that if the demand for a particular shoe

and of a particular size is not a multiple of 6, some positions will remain idle till all the shoes are complete. So, if demand is not a multiple of 6, the result from equation (4.8) would be a fraction. Hence, equation (4.8) has to be modified as below.

$$Nt_i = \begin{cases} \frac{Di}{6}, Di \in \text{Multiples of } 6 \\ \text{Integer}\left(\frac{D_i}{6}\right) + 1, \text{otherwise} \end{cases} \quad (4.9)$$

Using this formula, the number of turns required to produce the two different sizes of shoes in the family FFPN-E are calculated as 96 and 41 respectively (Table 4-10).

Now that the number of turns for each job has been calculated, the next order of business is to calculate the cycle times for each job. Since every position is going to be processing the same job, cycle times for every job are going to be equal to the six times the injection times (in general the cycle time is the maximum among all the injection times of the shoes currently loaded on the machine, and hence this would probably vary every time a new shoe is loaded) of the shoes (T_i). So the total processing time for every job “i”, TP_i , is defined as follows.

$$TP_i = Nt_i \times T_i \times 6 \quad (4.10)$$

Using this formula the processing times for the jobs in family FMPB-Q are found to be 88.70 and 75.77 minutes (Table 4-10). Hence the total processing time for the family would be 164.47 minutes.

Table 4-10: Family FFPN-E

| Order | Shoe | Size | Demand (Di) | Injection Time (Ti) | Number of Turns (Nti) | Processing time (Pi) |
|-------|--------|------|-------------|---------------------|-----------------------|----------------------|
| fds | FFPN-E | 5 | 574 | 0.154 | 96 | 88.70 |
| fds | FFPN-E | 8 | 245 | 0.308 | 41 | 75.77 |

At time $T = 0$, it is assumed that each rotary machine is ready to start processing the first job that has been loaded. The processing time for the first job in family FFPN-E in cell FS-1 is 88.7 minutes. At the end of the 88.7 minutes, the machine would have turned 96 times to complete the 574 pairs of size 5 shoes of type FFPN-E.

Then the size 8 shoes of the same type are loaded. Since there is no color or material change involved, the setup time is 0 minutes. At time $T = 164.47$ minutes, the size 8 shoes of the family will be completed and the next family on the list that has been loaded on cell FS1 would have to be loaded.

The color and the material of the next family on the list are considered. If either the color or the material changes then the tank in the injection molding machine is emptied and cleaned. The molds are also cleaned. Hence, during this time, the injection molding machine (which in this case, is the cell) will not be available for use and this setup time is added to the start time of the first job of the next family in the list. The end time for the job and the start time of the subsequent jobs in that family are adjusted to account for this setup time.

The next family on the list in this example is FMPB-Q. Though the material for both these shoes is PVC, the color changes and hence the tank has to be cleaned. This is the setup time and the duration for this setup is 20 minutes. So the processing of FMPB-

Q will start 20 minutes after the end of processing of family FFPN-E i.e., processing of FMPB-Q will begin at $T = 184.47$ minutes.

This process is repeated until all the jobs loaded on to the cell have been scheduled. Similarly the next cell is considered and the all the jobs in the cell are scheduled in the same way as the first cell until all the jobs in all the cells have been scheduled. The Gantt chart for this scheduling phase is represented in Figure 4-20.

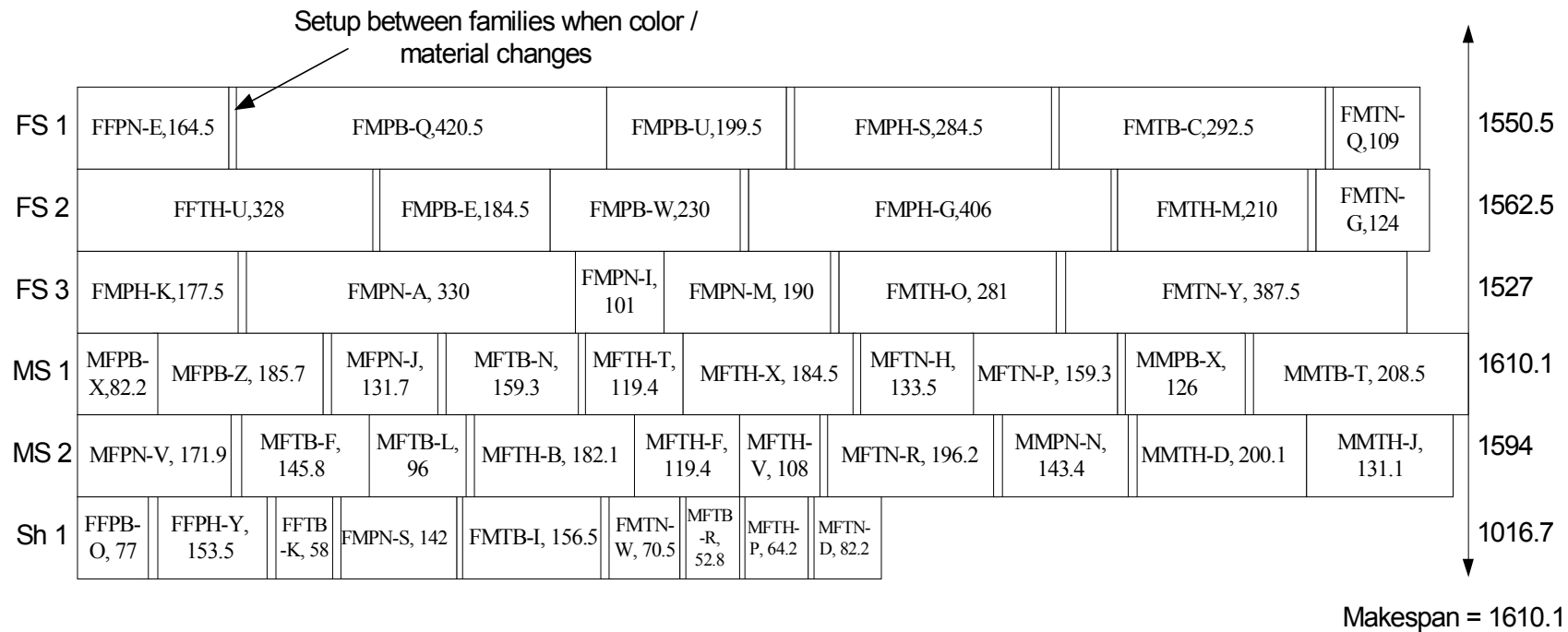


Figure 4-20: Gantt Chart (Do Not Split Heuristic) after scheduling phase

4.5 The Cell Loading And Cell Scheduling (CLACS) Application

Cell loading and scheduling operations require a number of mathematical calculations (usually simple and arithmetic in nature) and re-ordering of jobs and resources as the previous sections and chapters of this thesis have illustrated. These operations and calculations have to be repeated in ever increasing number of cycles as the number of jobs and cells increase and as the jobs vary in characteristics. Even in our case study at Timberland, the planners would work for close to two whole shifts just to develop a feasible schedule. However such a schedule was not reactive in nature and the non-static nature of the jobs prevented the static schedules from being met. Hence a software application is needed to develop schedules that are not just feasible, but reactive as well.

The Cell Loading And Cell Scheduling (CLACS) software application to encode the loading algorithms developed in this thesis and to schedule the jobs in the problem has been developed with Visual Basic which is a programming language whose features have been explained in previous chapters. Since the algorithms are in essence a sequence of events, the application is event-based with a number of functions. There are five major forms in the application and these forms incorporate user interface features like command buttons, decision boxes, drop-down lists and option buttons. The CLACS application enables the smooth and seamless transfer of control from one phase of the methodology to another. The application's input and output is through comma separated text files and enables the user to stop and re-start the process by storing intermediate data at the end of every stage in similar comma separated files that can be accessed later to re-start or

continue the loading and scheduling process. The major forms that form a part of CLACS are listed in Table 4-11.

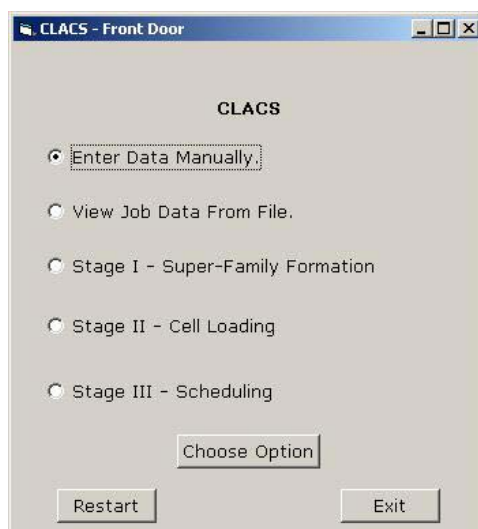
Table 4-11: The features of CLACS

| Form Title | Design Features | Function |
|------------------------------|--|---|
| Front Door | Option boxes, command buttons | To move back and forth between the different phases |
| Manual Data Entry | Drop down lists, text boxes, command buttons | To create data files of jobs |
| Calculate Load/Cells | Text boxes, command buttons | To calculate the number of cells and loads due to each super-family |
| Form Family | Command buttons | To form families based on job characteristics |
| Cell Loading | Command buttons, text boxes | To initiate and performs the cell loading process |
| Cell Loading Options | Option boxes | To choose the cell loading heuristic to apply |
| Number of Molds / Scheduling | Command buttons | To calculate the number of molds and schedule the jobs |

The Front Door form (Figure 4-21) incorporates an option box to which links to every stage of the methodology. The user can jump to any stage of the methodology from the front door – data entry, data file viewing, super family formation, cell loading and cell scheduling.

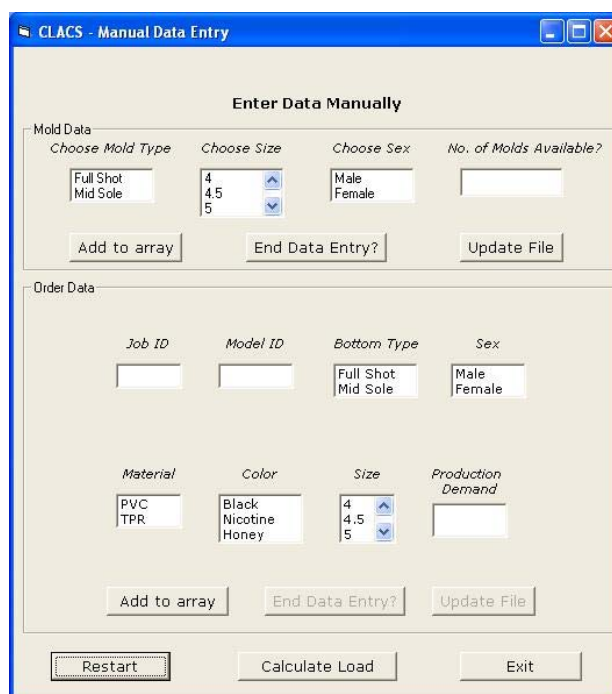
The data entry form (Figure 4-22) uses drop-down lists to help the user choose the bottom type (Full Shot / Mid Sole), sex, color, material and size with text boxes for

model id and order id, to create data files and also to generate an auto-key that is used to group families. This form also would be used in the future to create a data file for molds.



The image shows a software window titled "CLACS - Front Door". Inside the window, the title "CLACS" is centered at the top. Below it, there are five radio button options: "Enter Data Manually." (which is selected), "View Job Data From File.", "Stage I - Super-Family Formation", "Stage II - Cell Loading", and "Stage III - Scheduling". Below these options is a button labeled "Choose Option". At the bottom of the window are two buttons: "Restart" on the left and "Exit" on the right.

Figure 4-21: CLACS - Front Door form



The image shows a software window titled "CLACS - Manual Data Entry". The window is divided into two main sections: "Mold Data" and "Order Data".
 In the "Mold Data" section, there are four input fields: "Choose Mold Type" (with a dropdown menu showing "Full Shot" and "Mid Sole"), "Choose Size" (with a dropdown menu showing "4", "4.5", and "5"), "Choose Sex" (with a dropdown menu showing "Male" and "Female"), and "No. of Molds Available?" (an empty text box). Below these fields are three buttons: "Add to array", "End Data Entry?", and "Update File".
 In the "Order Data" section, there are four input fields: "Job ID" (empty text box), "Model ID" (empty text box), "Bottom Type" (with a dropdown menu showing "Full Shot", "Mid Sole", and "Female"), and "Sex" (with a dropdown menu showing "Male" and "Female"). Below these fields are three buttons: "Add to array", "End Data Entry?", and "Update File".
 At the bottom of the window are three buttons: "Restart", "Calculate Load", and "Exit".

Figure 4-22: CLACS - Manual Data Entry form

The view data file option button on the Front Door form is used to generate a Microsoft Excel sheet to enable viewing of data files in a manner that is much easier for a lay man to comprehend than a comma separated file that is ideal for automated input and output of data.

The super family formation option button brings up the relevant form (Figure 4-23) that incorporates the code to calculate and display loads due to the two super-families and to calculate the number of cells required from the data file and the MaxCap value chosen by the user.

CLACS - Calculate Load/Cells

Calculation of Loads and Cells

Browse for File:

Maximum Capacity per Cell
Enter the maximum capacity per cell here (mins/week).

Default is 1500 mins/week

Total Loads

Load on Cells

Full Shot: Mid Sole:

Total Load:

Number of Cells

Full Shot: Mid Sole: Shared Cells:

Total Cells:

Figure 4-23: CLACS - Calculate Load/Cells

Similarly the cell loading option button brings up another form (Figure 4-24) that initiates the cell loading process in a new form (Figure 4-25). This cell loading form offers the user flexibility by allowing the use of historical data (number of cells and MaxCap values) from the stored data files or to manually enter the same in the text boxes present for the purpose. When the “load cells” command button is clicked, a pop up form enables the user choose the heuristic and apply it. The last form (Figure 4-26) enables the user calculate the number of molds and to schedule the jobs by introducing setup times as and when these setup times are necessary.

Each of these forms can be accessed in sequence to apply the heuristic to the data in one go, through the use of transition buttons placed strategically. In addition, data is stored after every stage and each succeeding stage offers the user some choice in either using historical data or entering the same manually.

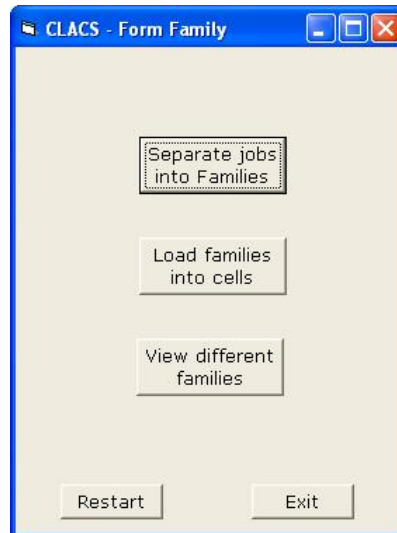
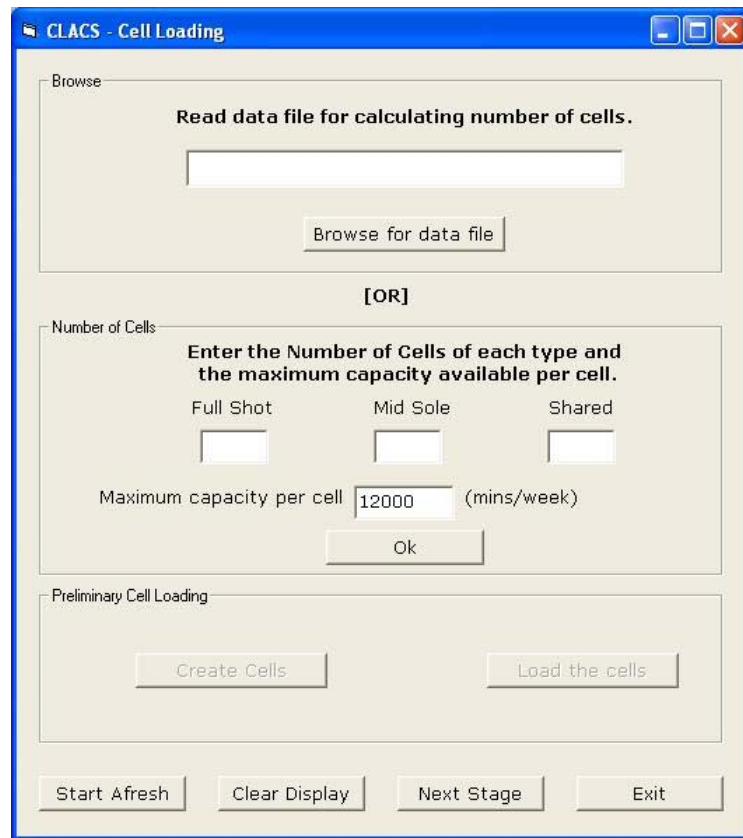


Figure 4-24: CLACS - Form Family



CLACS - Cell Loading

Browse

Read data file for calculating number of cells.

Browse for data file

[OR]

Number of Cells

Enter the Number of Cells of each type and the maximum capacity available per cell.

Full Shot Mid Sole Shared

Maximum capacity per cell (mins/week)

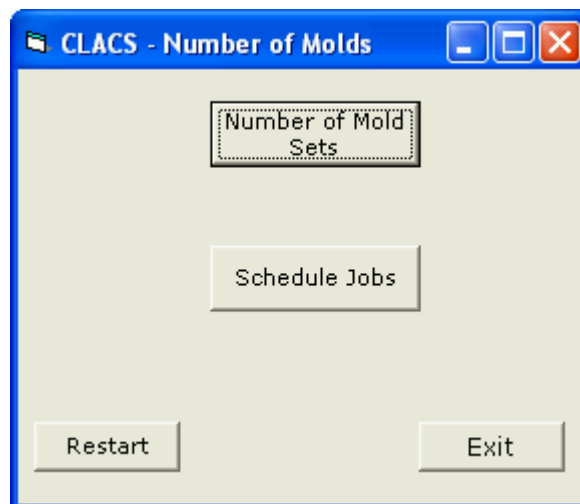
Ok

Preliminary Cell Loading

Create Cells Load the cells

Start Afresh Clear Display Next Stage Exit

Figure 4-25: CLACS - Cell Loading



CLACS - Number of Molds

Number of Mold Sets

Schedule Jobs

Restart Exit

Figure 4-26: CLACS - No of molds / Scheduling

One of the main strong points of the program is its flexibility and its relative lack of bulk. The user can exit the program at any stage as required and re-start where the process left off. Almost every decision is suggested, but not thrust on the user. This means that the results from every stage of the heuristics can be modified to suit the user's needs and the user is constrained to continue with the program's results if he does not want to. The relative lack of bulk means that the processing is relatively much faster than a mathematical programming model.

In fact a simple mathematical programming model to minimize makespan in a set of parallel machines, used in an attempt to generate results for comparison with the "No splitting allowed" heuristics developed in this thesis was running for more than 24 hours without generating anything more than an initial solution, while using the same computing resources, our application finds a solution within the space of seconds. This means that the software can be used repeatedly to compare feasible schedules resulting from the use of different values for variables such as MaxCap and number of cells.

CHAPTER 5. ANALYSIS OF RESULTS

In this chapter, the software application developed to implement the cell loading methodologies is tested using nine different problems, each comprising of anywhere between 12 to 45 different families. The data is based upon historical data from the company, with each family comprising of anywhere between two and fifteen different jobs.

First, the cell loading methodologies are compared using makespan as the objective. Then the effect of the MaxCap (both low and high values) on the makespan is studied. Then the idle times at the shared cell (if it exists) is compared at different values of MaxCap. The difference between the makespan during the cell loading and the cell scheduling phases are also compared as the number of families' increases.

5.1 Experimental Conditions

The different experimental conditions are as shown below in Table 5-1. Nine different problems were initially generated. Three of these families were small in size (12 - 20 families) while three were large sized (28 to 45 families) while the last three were medium sized (with 20 to 28 families).

Table 5-1: Experimental Conditions

| Problems | Number of problems | Number of cells | Number of families |
|----------|--------------------|-----------------|--------------------|
| Small | 3 | 3-4 | 12-20 |
| Medium | 3 | 4-5 | 20-28 |
| Large | 3 | 5-6 | 28-45 |

The lowest theoretical makespan value called the MaxCap* was calculated for each problem assuming that there are 6 cells available. This value is the lowest makespan that the particular problem can have, for that particular value of number of cell and would hence the load will be balanced across the cells to the maximum extent possible. The minimum theoretical value of makespan, MaxCap* is calculated as follows.

$$MaxCap^* = \frac{\sum_{i=1}^n Load}{NC_f + NC_m + NC_s} \quad (5.1)$$

Where,

NCf = Number of Full Shot Cells,

NCm = Number of Mid Sole Cells,

NCs = Number of Shared Cells,

A detailed description of the various problems and their characteristics can be seen in Table 5-2.

Table 5-2: Detailed Experimental Conditions

| Problem | No. of Families (nf) | | Load | | | MaxCap* |
|---------|----------------------|----------|-----------|----------|----------|---------|
| | Full Shot | Mid Sole | Full Shot | Mid Sole | Total | |
| big1 | 25 | 22 | 6261.332 | 4647.16 | 10908.49 | 1818 |
| big2 | 16 | 9 | 8406.51 | 2718.218 | 11124.73 | 1854 |
| big3 | 20 | 13 | 7488.622 | 3629.03 | 11117.65 | 1853 |
| medium1 | 14 | 9 | 4797.243 | 3050.345 | 7847.588 | 1308 |
| medium2 | 13 | 11 | 4936.167 | 3014.887 | 7951.053 | 1325 |
| medium3 | 15 | 11 | 5764.072 | 3807.737 | 9571.808 | 1595 |
| Small1 | 11 | 9 | 3661.713 | 2487.008 | 6148.722 | 1025 |
| Small2 | 9 | 6 | 4571.775 | 1791.338 | 6363.113 | 1061 |
| Small3 | 12 | 11 | 4899.31 | 2928.375 | 7827.685 | 1305 |

5.2 Comparison of cell loading heuristics

The four different cell loading methodologies developed are compared at different MaxCap values. The CLACS application developed to implement these four methodologies is for this purpose. The comparison is performed based on the data from 9 different problems whose details are given in Table 5-1. The comparison is against the makespan values for each of these problems at different MaxCap values.

The four different methodologies were applied to 5 different problems using the CLACS application at different MaxCap values (Table 5-3).

Table 5-3: Makespan Values for Different Strategies (Pre-Schedule)

| Problem | No of Cells | MaxCap | No Split | Split Few | Split All | Load Jobs |
|---------|-------------|--------|----------------|----------------|----------------|----------------|
| big1 | 6 | 2000 | 1993.19 | 1993.19 | 1999.12 | 1995.86 |
| big2 | 6 | 2200 | 2199.03 | 2199.03 | 2197.69 | 2148.60 |
| big3 | 6 | 1950 | 1922.89 | 1922.89 | 1933.94 | 1900.91 |
| medium1 | 6 | 1400 | 1377.74 | 1377.74 | 1392.67 | 1398.68 |
| medium2 | 6 | 1500 | 1485.96 | 1485.96 | 1496.96 | 1499.86 |
| medium3 | 6 | 1900 | 1874.63 | 1874.63 | 1896.30 | 1899.67 |
| small1 | 6 | 1200 | 1190.77 | 1190.77 | 1190.63 | 1197.00 |
| small2 | 4 | 1700 | 1694.27 | 1694.27 | 1695.06 | 1699.20 |
| Small3 | 5 | 1650 | 1561.43 | 1561.43 | 1641.20 | 1631.82 |

The difference in makespan values across strategies is not large and is not enough to pinpoint the effectiveness of one strategy over the other. Hence at this point, the choice of strategy is not clear. This choice should become increasingly apparent once the results from the scheduling phase are studied. However, the makespan values increase as the level of splitting allowed increases from none allowed to as much as required (Figure 5-1,

Figure 5-2, Figure 5-3 and Figure 5-4). While we might be tempted to pick the lowest makespan and hence the No-Split allowed option, we might actually be making the wrong choice then. This is because, these MaxCap values have actually been chosen randomly.

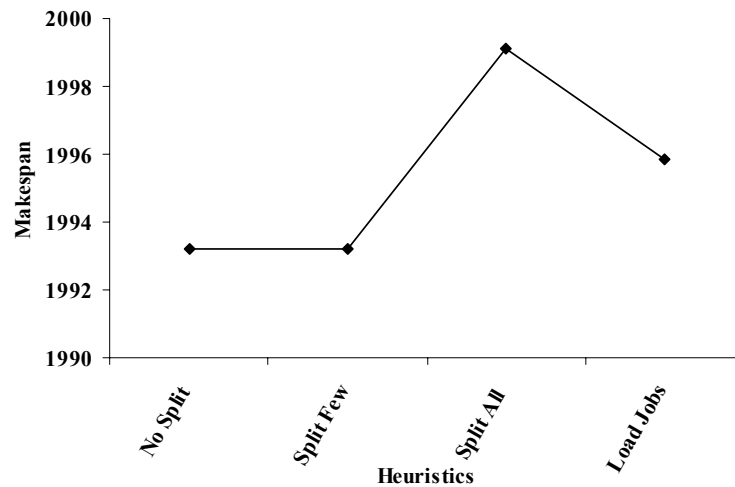


Figure 5-1: Makespan for Problem Big1 @ MaxCap = 2000

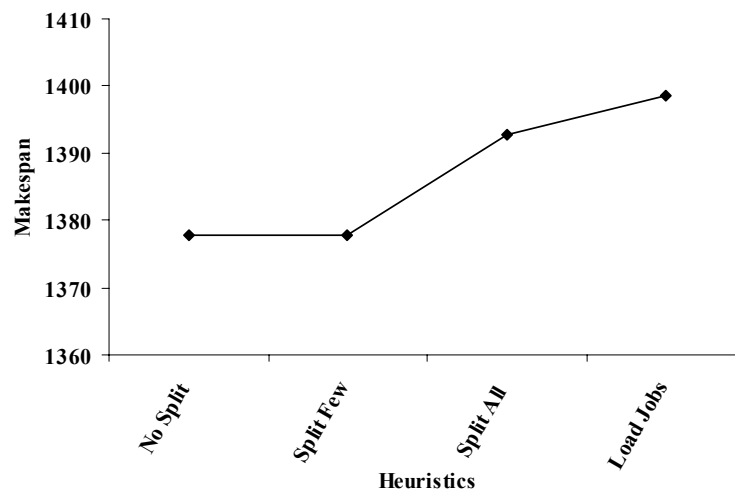


Figure 5-2: Makespan Trends for Problem Medium1 @ MaxCap = 1400

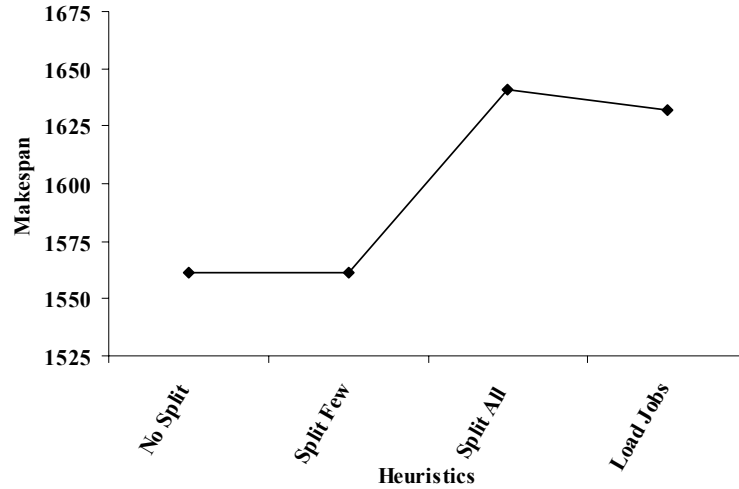


Figure 5-3: Makespan Trends for Problem Small3 @ MaxCap = 1650

5.3 Effects of MaxCap on Makespan (lower bound)

A logical grasp of the situation indicates that the “Split as and when required” (also referred as “Split all” in Table 5-3) heuristics might be a good choice as it would result in an even balance of load (which is one of our major objectives). It is also true that at lower MaxCap values (closer to the theoretical minimum makespan), not all the heuristics would result in feasible solutions (i.e. completion of all jobs) and as MaxCap increases, it is almost always the “Split as and when required” heuristic that will provide us with a feasible solution even when the other heuristics do not.

In order to test this hypothesis, for one problem (Small3), the value of MaxCap was varied from the theoretical minimum makespan (in this case, 1566 minutes) to the lowest value of MaxCap when all the four heuristics result in feasible solutions. When the value of MaxCap is 1566 minutes, none of the heuristics result in feasible solutions. The first feasible result for the “Split as and when required” heuristic appears at a MaxCap value of 1625 () and all four heuristics result in feasible solutions when the

MaxCap value is 1700 minutes. The presence of even lower makespan for the “Load Jobs” heuristics can be disregarded due to the randomness associated with the procedure.

Table 5-4: Effect of increase in MaxCap on Makespan (Problem Small3)

| Strategy | 1566 | 1575 | 1600 | 1625 | 1650 | 1675 | 1700 |
|-----------|------|------|------------|------------|------------|------------|------------|
| No Split | n/a | n/a | n/a | n/a | n/a | n/a | 273 |
| Split Few | n/a | n/a | n/a | n/a | n/a | n/a | 273 |
| Split All | n/a | n/a | n/a | 273 | 273 | 273 | 273 |
| Load Jobs | n/a | n/a | 273 | 273 | 273 | 273 | 273 |

Similar results were seen when the same procedure was applied to another problem (Big 1). In this case too, the Split All and the Load Jobs resulted (Table 5-5) in the earliest instance of a feasible solution with all the jobs completed within 2400 minutes.

Table 5-5: Effect of increase in MaxCap on Makespan (Problem Big1)

| Strategy | 1825 | 1850 | 1875 | 1880 | 1890 | 1900 |
|-----------|------|------------|------------|------------|------------|------------|
| No Split | n/a | n/a | n/a | n/a | n/a | 461 |
| Split Few | n/a | n/a | n/a | n/a | n/a | 461 |
| Split All | n/a | n/a | 461 | 461 | 461 | 461 |
| Load Jobs | n/a | 461 | 461 | 461 | 461 | 461 |

So it is obvious that if the MaxCap values are chosen randomly, then it is sometimes possible that we miss the best solution which might appear at a much lower value of MaxCap than the one chosen. And as the MaxCap value increases, the makespan also increases, which indicates that the makespan value is driven by the MaxCap. So the

choice of MaxCap key to heuristic performance and the lowest MaxCap value that results in a feasible solution should be chosen if makespan is to be minimized.

5.4 Effects of MaxCap on Makespan (upper bound)

We have shown earlier that the choice of a MaxCap value close to the lower bound (i.e. the minimum theoretical makespan) is critical to minimizing makespan. But the effects of a higher value of MaxCap can be seen only after scheduling the jobs.

After the scheduling phase (Table 5-6), the trends seen in the Pre-Schedule stage (Table 5-3) are still mostly apparent at the end of scheduling stage. But as mentioned earlier the choice of MaxCap has been random until this point and so the effect of a high MaxCap value is still not apparent.

Table 5-6: Makespan Values for Different Strategies (Post-Schedule)

| Problem | No. of Cells | MaxCap | No Split | Split Few | Split All | Load Jobs |
|---------|--------------|--------|----------|-----------|-----------|-----------|
| big1 | 6 | 2000 | 2088.35 | 2088.35 | 2095.62 | 2095.86 |
| big2 | 6 | 2200 | 2175.68 | 2175.68 | 2294.19 | 2248.60 |
| big3 | 6 | 1950 | 2013.49 | 2013.49 | 1973.94 | 1960.91 |
| medium1 | 6 | 1400 | 1397.74 | 1397.74 | 1452.67 | 1435.21 |
| medium2 | 6 | 1500 | 1551.60 | 1551.60 | 1576.96 | 1539.86 |
| medium3 | 6 | 1900 | 1914.63 | 1914.63 | 1996.30 | 1939.67 |
| small1 | 6 | 1200 | 1250.77 | 1250.77 | 1249.72 | 1236.28 |
| small2 | 4 | 1700 | 1738.56 | 1738.56 | 1735.06 | 1739.20 |
| small3 | 5 | 1650 | 1641.43 | 1641.43 | 1661.20 | 1671.82 |

Hence, the MaxCap values were increased for one problem to show how a high value of MaxCap would affect the results. This experiment was performed on problem

Big1. In this experiment, the MaxCap value for problem Big1 was increased from 1818 (theoretical makespan value for this problem from Table 5-2) to 1900. From the results (Table 5-7), at lower MaxCap values, all the four heuristics result in a post-schedule makespan value that is much lesser than the 2400 minutes available to the user per cell. But when the MaxCap is increased to 2300, the post-schedule makespan value for the Split All heuristic is higher than 2400, which cannot be permitted. So the choice of MaxCap has to be made carefully to ensure that such a situation does not arise.

Table 5-7: Effect of MaxCap (upper bound)

| Strategy | MaxCap = 1900 | | MaxCap = 2000 | | MaxCap = 2300 | |
|-----------|---------------|---------------|---------------|---------------|---------------|----------------|
| | Pre-Schedule | Post-Schedule | Pre-Schedule | Post-Schedule | Pre-Schedule | Post-Schedule |
| No Split | 1888.33 | 2038.21 | 1993.19 | 2088.35 | 2271.51 | 2390.96 |
| Split Few | 1888.33 | 2038.21 | 1993.19 | 2088.35 | 2271.51 | 2390.96 |
| Split All | 1898.25 | 2098.06 | 1999.12 | 2095.62 | 2296.44 | 2400.61 |
| Load Jobs | 1898.84 | 1998.84 | 1995.86 | 2095.86 | 2298.30 | 2398.30 |

5.5 Comparison of cell idle times

Cell idle times are a good indicator of the relative performance of the different heuristics. In this case however, it is the MaxCap values that will be used rather than the total capacity available. The CLACS application is used to test problems big1, big2 and medium 3 and the results are compared.

The results (seen on Table 5-8, Table 5-9 and Table 5-10) indicate that the shared cell idle times increase with an increase in the level of splitting. This is mainly because the other specific cells are loaded to a greater extent than when splitting is not allowed.

As the splitting increases (Figure 5-4, Figure 5-5 and Figure 5-6), there is a perceptible shift in idle times from the specific cells toward the shared cell, with the maximum shared cell idle time to be found in the Split All heuristic. The shared cell idle times vary when the Load Jobs heuristic is used and this is indicative of the randomness involved in this heuristic.

Table 5-8: Idle Times for Problem Big1 @ MaxCap = 1900

| Cell | Heuristic | | | |
|------------|-----------|-----------|-----------|-----------|
| | No Split | Split Few | Split all | Load Jobs |
| Full Shot1 | 103.66 | 103.66 | 31.79 | 1.66 |
| Full Shot2 | 30.51 | 30.51 | 29.03 | 0.52 |
| Full Shot3 | 14.30 | 14.30 | 2.71 | 3.83 |
| Mid Sole1 | 72.22 | 72.22 | 13.35 | 1.31 |
| Mid Sole2 | 83.30 | 83.30 | 1.38 | 1.65 |
| Shared | 62.82 | 62.82 | 469.92 | 357.83 |

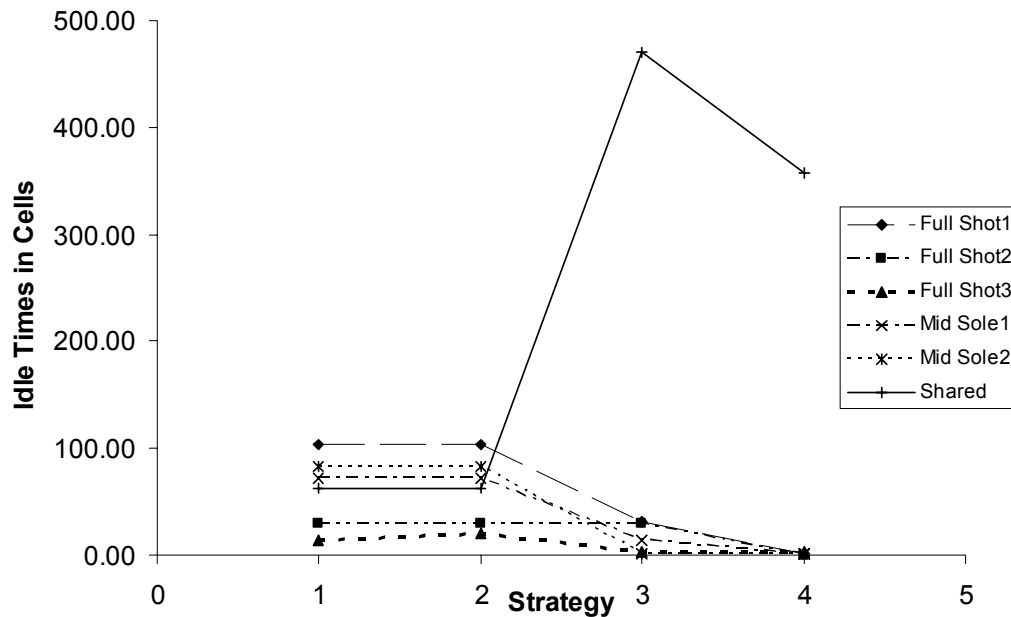


Figure 5-4: Idle Time Trends for Problem Big1 (MaxCap = 1900)

Table 5-9: Idle Times for Problem Big2 (MaxCap = 2025)

| Cell | Heuristic | | | |
|------------|-----------|-----------|-----------|-----------|
| | No Split | Split Few | Split all | Load Jobs |
| Full Shot1 | 176.02 | 48.19 | 176.02 | 3.00 |
| Full Shot2 | 95.95 | 34.09 | 95.95 | 7.26 |
| Full Shot3 | 45.45 | 45.45 | 11.06 | 0.29 |
| Mid Sole1 | 110.06 | 110.06 | 1.71 | 1.05 |
| Mid Sole2 | 86.78 | 86.78 | 86.78 | 2.61 |
| Shared | 178.61 | 178.61 | 399.59 | 756.91 |

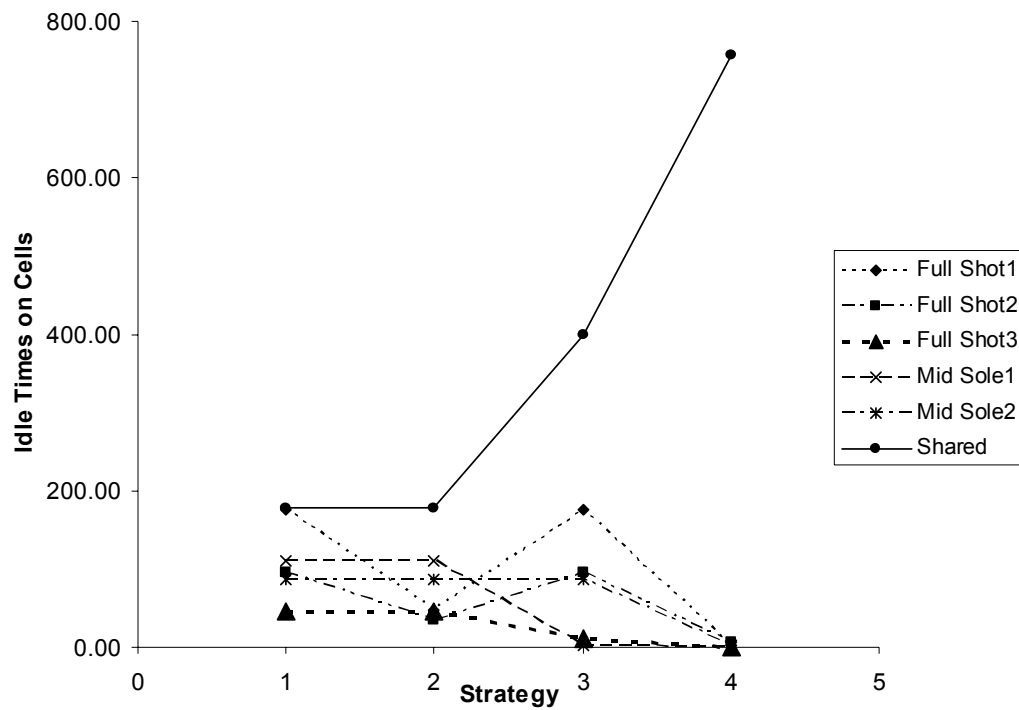
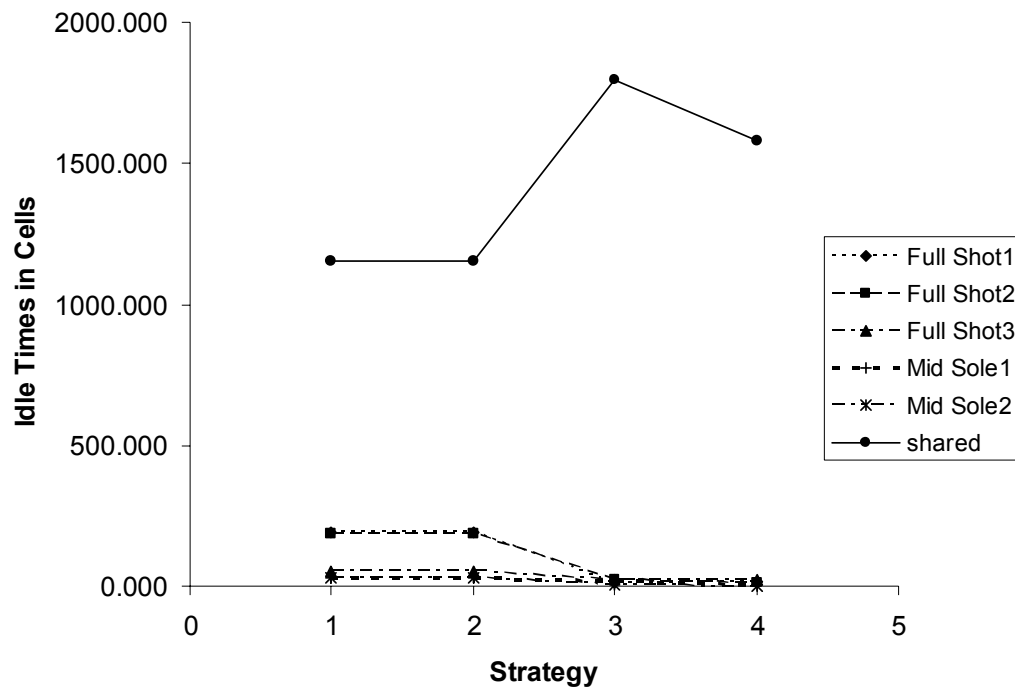
**Figure 5-5: Idle Time Trends for Problem Big2 (MaxCap = 2025)**

Table 5-10: Idle Times for Problem Medium3 (MaxCap = 1900)

| Cell | Heuristic | | | |
|------------|-----------|-----------|-----------|-----------|
| | No Split | Split Few | Split all | Load Jobs |
| Full Shot1 | 192.78 | 192.78 | 12.04 | 19.08 |
| Full Shot2 | 184.08 | 184.08 | 24.65 | 9.26 |
| Full Shot3 | 51.69 | 51.69 | 24.59 | 22.20 |
| Mid Sole1 | 32.88 | 32.88 | 18.91 | 6.15 |
| Mid Sole2 | 27.18 | 27.18 | 5.20 | 2.46 |
| shared | 1150.79 | 1150.79 | 1793.63 | 1580.25 |

**Figure 5-6: Idle Time Trends for Problem Medium 3 (MaxCap = 1900)**

As seen from the results, the idle times in the shared cell increase as the level of splitting increases. This could be an indication that the shared cell could be used as a part time cell whenever is very high. This means that the shared cell does not need to be used all the time and incase the system does not have the capability of including a shared cell

(i.e. if there are only five cells, but the heuristics indicate that a sixth shared cell be used), one of the cells in the system can be converted into the shared cell only in the overtime period and used to process the jobs that the heuristic loads on the shared cell.

5.6 Comparison of Makespan (number of families)

The increase in the number of families is expected to have an adverse effect on the makespan due to the increases in the number of setups potentially required. Hence it is imperative to test the effects of such an increase in the number of families. Three different problems (one each from each group – big, small and medium) are used to test the effects of increase in the number of families on the makespan (Table 5-11). The results of this experiment are represented in Figure 5-7. The results reiterate the logical hypothesis that as the number of families increase, so does the number of setups. This means that the gap between the makespan values at the end of the loading phase and scheduling phase will also increase.

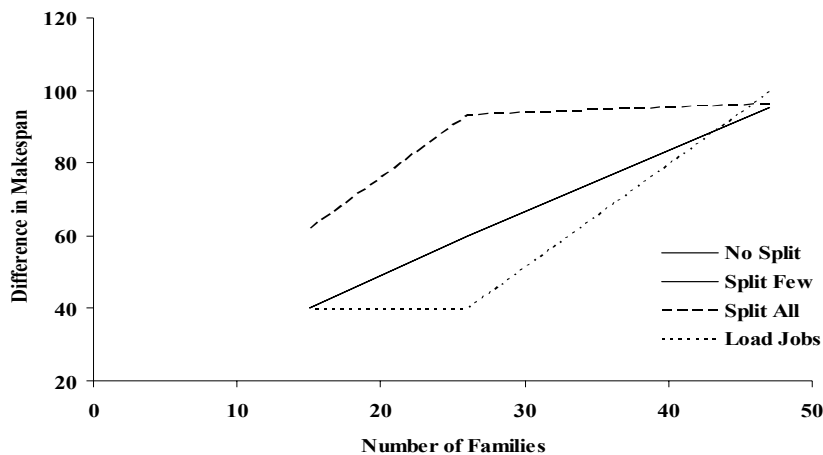


Figure 5-7: Effect of number of families (nf) on Makespan (MaxCap = 2000)

Table 5-11: Makespan differences at different nf values (MaxCap = 2000)

| Cell Loading Strategy | Big1 (nf = 47) | | | Medium3 (nf = 26) | | | Small1 (nf = 15) | | |
|-----------------------|----------------|------------|------------|-------------------|------------|------------|------------------|------------|------------|
| | Makespan | | | Makespan | | | Makespan | | |
| | Loading | Scheduling | Difference | Loading | Scheduling | Difference | Loading | Scheduling | Difference |
| No Split | 1993.19 | 2088.35 | 95.16 | 1981.95 | 2041.95 | 60.00 | 1940.92 | 1980.92 | 40.00 |
| Split Few | 1993.19 | 2088.35 | 95.16 | 1981.95 | 2041.95 | 60.00 | 1940.92 | 1980.92 | 40.00 |
| Split All | 1999.12 | 2095.62 | 96.51 | 1994.66 | 2088.23 | 93.57 | 1996.58 | 2058.68 | 62.10 |
| Load Jobs | 1995.86 | 2095.86 | 100.00 | 1998.82 | 2038.82 | 40.00 | 1833.42 | 1873.42 | 40.00 |

CHAPTER 6. CONCLUSIONS AND FUTURE WORK

In this chapter, results and conclusions of experimentation with the four cell loading algorithms developed are summarized and then future work is discussed.

This research was aimed at formulating a methodology to determine the number of cells (each comprising of a single rotary injection molding machine with six pairs of stations) needed to manufacture orders (each comprising different models of shoes with various different sizes at different volumes) and load these orders on to the cells after grouping them into families. Four heuristic procedures were implemented with the objective of balancing the load among all the cells and to maximize the utilization of the cells. These heuristics were compared with respect to completed jobs, makespan and setup times. A simple cell scheduling methodology was also developed.

The relationship between MaxCap and makespan was studied. The performance of the different heuristics was compared by varying the MaxCap values and the results were analyzed in an attempt to identify optimal values. The effect of the increase in number of families on the amount of setup times was studied. This was accomplished by applying testing the heuristics on different problems at a common MaxCap value. The performance of the different heuristics was also compared with the objective being the reduction of cell idle times.

6.1 Conclusions

The CLACS application is used to implement the four loading strategies – Do not Split, Split Only When Load Greater Than Maxcap, Split As And When Required and Load Jobs. The four loading strategies are compared with respect to the resulting

makespan values. Testing shows that little separates the four strategies after the loading phase. But upon applying MaxCap values carefully we find that the Split as and When Required and the Load Jobs heuristic gives a feasible result at a lower value of MaxCap than the other two heuristic.

When MaxCap values are varied for the same cell configurations, the heuristics perform along the same lines as before, but the difference between the makespan values is more pronounced with the makespan converging with the MaxCap values closest in the Split As and When Required Heuristic. Hence it can be perhaps concluded that for higher values of MaxCap, the Split As and When Required heuristic would work better. However possible delays due to machine downtimes and setups have to be accounted for and it is always imperative that this is done as early as possible hence the makespan can never be allowed to go beyond a fixed point. Moreover if the MaxCap value is too high, the post schedule makespan might increase to a value that would be greater than the time available per cell. Hence the value of MaxCap seems to be a key variable in the performance of these heuristics and so a logically optimum value of MaxCap is ideal toward increasing the efficiency of the system.

Overall, the Load Jobs heuristic seems to perform the best. But due to doubts that arise regarding its consistency, the “Split as and when Required” heuristic is recommended as the best. This is a reaffirmation of our logic that points out to the fact that the load is balanced better when splitting is possible. But it has to be noted that the optimal solution from the cell loading level might not be significant as the results from the cell scheduling phase may vary.

When different problems were tested with a constant value of MaxCap, it was observed that the set up times increase with an increase in the number of families, which is a logical conclusion. As expected, the Split As and When Required performs the worst in this experiment.

6.2 Future Work

The Load Jobs heuristic deliberately uses the First In First Out method for loading the jobs on the cells. This method has been adopted because the initial view was to attempt to illustrate the positives of grouping jobs into families which helps to minimize setups. But since the number of families is quite large, this minimization is not apparent and hence the “Load Jobs” heuristic seems to perform better in some cases. Hence the use of some other rule (maybe LPT) other than the First in First Out rule might help us weigh the positives of this last heuristic better.

The four loading methodologies have to be complemented with an appropriate cell scheduling methodology to ensure that the facility can be run smoothly. The methodology can be extended to consider the possibility of running different sizes simultaneously on the rotary machine at any point of time. Furthermore mold restrictions can be added to make it more realistic.

Cost analysis of solutions might have to be performed. The trade-offs between even load (and idle time) distribution among all the cells versus the loading all the specific cells with the maximum idle times in the shared cell have to be studied. These trade-offs would be critical because increased idle times in the shared cell could imply

that the shared cell may be used on a part time basis and/or overtime only in one of the dedicated cells.

Since this work concentrates only on the Rotary Scheduling Machine, similar methodologies have to be developed for the other cells in the system – The Lasting Cell and the Finishing Cell. It is hoped that the methodologies developed by this thesis would simplify the scheduling of those cells.

BIBLIOGRAPHY

1. McNaughton, R., 1959, 'Scheduling with deadlines and loss functions', *Management Science*, 6, 1-12.
2. Mitrofanov, S. P., 1966, *Scientific Principles of Group Technology*. Boston Spa, Yorkshire, England, National Lending Library for Science and Technology, translated from Nauchnye Osnovy Gruppovoi Tekhnologii, Leningrad.
3. Baker, K., R., 1974, *Introduction to Sequencing and Scheduling*, John Wiley and Sons, Inc., New York.
4. Burbidge, J. I., 1975, *The Introduction of Group Technology*, New York: John Wiley & Sons).
5. Hax, A. C., and Harlan C. M., 1975, 'Hierarchical integration of production planning and scheduling', M. A. Geisler, ed., *Studies in Management Sciences*, 1: Logistics, New York: Elsevier.
6. Ham, I., 1976, 'Introduction to Group Technology', *SME Technical Report MMR76-03*, Society of Manufacturing Engineers, Dearborn, MI.
7. Lang, D., W., 1977, *Critical Path Analysis*, New York: David McKay.
8. Ham, I. and Hitomi K., 'Group Technology Applications for Machine Loading under multi-resource constraints', *Proceedings of IX North American Manufacturing Conference and 1981 SME transactions*, Society of Manufacturing Engineers, 515 – 518.
9. Green, T. J., and Sadowski, R. P., 1984, 'Cellular Manufacturing Control', *Journal of Manufacturing Systems*, 4, 137-145.

10. Sinha, R.K., and Hollier, R.H., 1984, 'A review of production control problems in cellular manufacturing', *International Journal of Production Research*, 22(5), 773-789.
11. Morris, J.S., and Tersine, R. J., 1989, 'A Comparison of Cell Loading Practices in Group Technology', *Journal of Manufacturing and Operations Management*, 2 (4), 299-313.
12. Wemmerlov, U., and Hyer, N., L., 1989, 'Cellular Manufacturing In The US Industry: A Survey Of Users', *International Journal of Production Research*, 27, 1511-1530.
13. Garza, O. and Smunt, T. L., 1991, 'Countering the Negative Impact of Flexibility on Manufacturing Performance', *Journal of Operations Management*, 10, 92 - 118.
14. Süer, G., A., and Saiz, M., 1993, 'Cell Loading in Cellular Manufacturing Systems', *Computers and Industrial Engineering*, 25, No.1 - 4, 247 – 250.
15. Süer, G., A., and Saiz, M., 1995, 'Cell Loading in Cellular Manufacturing Systems', *Proceedings of the 15th Conference on Computers and Industrial Engineering*, 1993.
16. Süer, G.A., Saiz, M., Dagli, C. and Gonzalez, W., 1995, 'Manufacturing Cell Loading Rules and Algorithms for Connected Cells', *Planning, Design and Analysis of Cellular Manufacturing Systems*, Elsevier Science.
17. Benjaffar, S., 1995, 'Machine Sharing and Performance Of Cellular Manufacturing Systems', *Journal of Materials Processing Technology*, 52, 91 – 101.
18. Ashby, J.R and Uzsoy Reha, 1995, 'Scheduling And Order Release In A Single Stage Production System', *Journal of Manufacturing Systems*, Vol.14 / No.4.

19. Süer, G.A., Saiz, M., Dagli, C. and Gonzalez, W., 1995, 'Evaluation Of Manufacturing Cell Loading Rules For Connected Cells', *Planning, Design, and Analysis of Cellular Manufacturing Systems*, ed. A.K. Kamrani, H.R. Parsaei and D.H. Liles, Elsevier Science B.V, 97 – 127.
20. Ho, J.C., and Wong, J., S., 1995, 'Makespan minimization for parallel identical processors', *Naval Research Logistics*, 42, 935-948.
21. Süer, G.A., 1996, 'Optimal Operator Assignment and Cell Loading in Labor Intensive Manufacturing Cells', *Computers and Industrial Engineering*, 31, 1-2, 155 – 158.
22. Akturk, S.M. And Wilson, G.R., 1998, 'A Hierarchical Model For The Cell Loading Problem Of Cellular Manufacturing Systems', *International Journal of Production Research*, 36, 7, 2005 – 2023.
23. Süer, G.A., and Bera, I., S., 1998, 'Multi-period Cell Loading and Cell Size Determination', *Computers and Industrial Engineering*, 35, 1-2, pp 85 – 88.
24. Riezebos, J., 1998, 'Production planning systems for cellular manufacturing', *Group Technology & Cellular Manufacturing: a state-of-the-art synthesis of research and practice*, edited by N.C. Suresh and J.M. Kay, Kluwer Academic Publishers, Boston, USA.
25. Süer, G.A., and Bera, I., S., 1998, 'Optimal Operator Assignment And Cell Loading In Labor Intensive Manufacturing Cells When Lot-Splitting Is Allowed', *Computers and Industrial Engineering*, 35, 3-4, 431 – 434.
26. Mahmoodi, F., and Mosier, C., T., 1998, 'Scheduling Rules for Cellular Manufacturing', *Group Technology & Cellular Manufacturing: a state-of-the-art*

- synthesis of research and practice*, edited by N.C. Suresh and J.M. Kay, Kluwer Academic Publishers, Boston, USA, 321 – 338.
27. Süer, G.A., Vasquez, R., Cortes, M. and Peria, Y., 1998, *Evolutionary Programming Based Cell Loading*, Proceedings of the Second Japan Australia Workshop on Intelligent and Evolutionary Systems, ed. O.Katai, X.Yao, M. Gen, Y. Tsujimura.
 28. Ruben, R., and Mahmoodi, F., 1999, *Scheduling Cellular Manufacturing Systems*, Handbook of cellular manufacturing systems, edited by Shahrukh A. Irani, John Wiley & Sons.
 29. Süer, G.A., Saiz, M. and Gonzalez, w., 1999, Evaluation Of Manufacturing Cell Loading Rules For Independent cells, *International Journal Production Research*, 37, 15, 3445 – 3468.
 30. Baykasoglu, A., Gindy, N., Z., 1999, ‘Loading Flexible Cells: Tabu Search Based Simulation Optimization Approach’, *Proceedings of the 15th International Conference on Production Research*, Ed. M.T. Hillery and H.J. Lewis, 2, 1441-1444.
 31. Süer, G.A., Santos, J., And Vasquez, R., 1999, ‘Scheduling Rotary Injection Molding Machines’, *Proceedings of the Second Asia-Pacific Conference on Industrial Engineering and Management Systems*, 319 – 322.
 32. Mahmoodi, F., Mosier, C.T., And Morgan, J.R., 1999, ‘The Effects of Scheduling Rules and Routing Flexibility on the Performance of a Random Flexible Manufacturing System’, *International Journal of Flexible Manufacturing Systems*, 11, 271–289.

33. Lozano, S., Guerrero, F., Eguia, I., and Onieva, L., 1999, 'Cell design and loading in the presence of alternative routing', *International Journal of Production Research*, 37 (14), 3289 - 3304.
34. Shambu, G. and Suresh, N.C., 2000, 'Performance Of Hybrid Cellular Manufacturing Systems: A Computer Simulation Investigation', *European Journal of Operational Research*, 120, 436 – 458.
35. Gupta, J. N. D., Ho, J. C., 2000, 'Minimizing flow time subject to optimal makespan on two identical parallel machines', *Brazilian Journal of Operational Research*, 20, (1), pp. 5-17.
36. Süer, G.A., Vasquez, R., and Cortes, M, 2000, 'A Hybrid Approach of Evolutionary Programming and Local Optimizers in Cell Loading' (submitted to *Computers and Industrial Engineering*).
37. Hans, E., 2001, 'Resource Loading by Branch-and-Price Techniques', Ph.D. thesis, University of Twente.
38. Gómez, A.M., 2001, 'A Hierarchical Hybrid Approach to Cell Loading, Manpower Allocation and Job Sequencing in Cellular Manufacturing', M.S Thesis, University of Puerto Rico, Mayagüez, Puerto Rico.
39. Saad, S.M., Gindy, N.Z. and Baykasoglu, A., 2002, 'A New Integrated System For Loading And Scheduling In Cellular Manufacturing', *International Journal of Computer Integrated Manufacturing*, 15 (1), 37–49.
40. Babayiğit, C., November 2003, Genetic Algorithms and Mathematical Models in Manpower Allocation and Cell Loading Problem, M.S. Thesis, Ohio University.