Dynamic Optimization of Job Distribution on Machine Tools Using Time Decomposition into Constant Job-Mix Stages

by

Subramanian Natarajan

Department of Mechanical Engineering
McGill University
Montréal, Canada

March, 1992

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfilment of the requirements for the degree of Master of Engineering

To Amma & Appa...

With Respect, Gratitude and Admiration
for all they have done.

Abstract

This thesis deals with the development, analysis and application of a new method to optimize the allocation of jobs on machine tools. The benefits of this method are derived through time-decomposition of the scheduling horizon.

The decomposition scheme is based on the scheduled flow of jobs *i.e.*, the input of jobs to the shop floor and their departure after processing. The partitioning procedure divides the planning horizon into 'stages', or time periods, at which the job-mix remains constant. The optimization of job allocation is carried out within each partition and successive stages are treated sequentially. The dynamic nature of the problem is such that the solution at a stage affects the boundary conditions of the subsequent stage. The Constant Job-Mix Stage (CMS) algorithm developed to solve the job allocation problem, accounts for the setup times and enables one to obtain integer solutions while reducing slack on machines and enforcing due date on jobs.

The application of the algorithm is demonstrated for three different cases. The first two cases focus on single operation jobs and represent two different approaches to scheduling. The former is concerned with scheduling jobs from starting times illustrating 'push' system while the latter is due date driven, representing a 'pull' system which corresponds to the just-in-time (JIT) philosophy. The third case deals with the assignment of multiple operation jobs to machine tools which are grouped according to processes. The results indicate that the optimization based on the constant job-mix stages, leads to increased utilization of machine tools, higher production rate, with shorter makespan of individual jobs and reduced computational time.

The possibility of interfacing the CMS algorithm with the other components of manufacturing systems is also discussed. The opportunities for the enhancement of integrated intelligent systems, through open and feedback loops are pointed out.

Résumé

Cette thèse traite du développement, de l'analyse, et de l'application d'une nouvelle méthode pour optimiser la répartition de tâches sur des machines-outils. Les avantages de cette méthode proviennent de la décomposition temporelle du calendrier de planification.

Le plan de décomposition est basé sur le flux planifié des tâches, c'est-à-dire sur l'arrivée de tâches dans l'atelier et leur départ une fois accomplies. Le processus de fractionnement divise le calendrier en étapes ou intervals de temps pendant lesquelles le *job-mix* reste constant. L'optimisation de la répartition des tâches s'effectue au sein de chaque fraction et les étapes successives sont traitées de façon séquentielle. La nature dynamique du problème est telle que la solution adoptée à une étape affecte les conditions aux limites de l'étape suivante. L'algorithme CMS (Constant Job-Mix Stage) développé pour résoudre le problème de répartition de tâches tient compte des temps d'installation et permet d'obtenir des solutions entières tout en réduisant les temps-morts des machines et en assurant l'accomplissement des tâches avant leur date d'échéance.

L'algorithme peut s'appliquer dans trois cas differents. Les deux premiers cas ne concernent que les tâches ne comportant qu'une opération et représentent deux approches différentes de planification. Le premier concerne la planification des tâches à partir des temps de départ, illustrant le système "*push* ", alors que le second est basé sur la date d'échéance, représentant un système "*pull* " qui correspond au principe du Juste-à-Temps. Le troisième cas traite de l'affectation de différentes tâches sur des machines-outils regroupées par procédé. Les résultats montrent que l'optimisation basée sur les étapes à *job-mix* constant conduit à un taux de production plus élevé avec un temps total d'accomplissement de chaque tâche plus court et un temps de calcul moins important.

La question de la communication entre l'algorithme CMS et d'autres composantes du système de production est traitée aussi. Les possibilités de renforcer les systèmes intelligents intégrés grâce à des boucles ouvertes et des boucles fermées sont également discutées.

Statement of Originality and Contribution to Knowledge

The author of this thesis claims originality for the development of the following concepts:

- The development of a new method to partition the scheduling horizon. The scheme is based on the scheduled flow of jobs, their input and output from the shop floor, such that the partitions have constant job-mix.

- The dynamic optimization principle. The optimization procedure moves from one stage to the next providing a link between the stages, exposing the dynamic nature of the problem.

The constant job-mix stage (CMS) algorithm developed on the basis of the above two concepts, is construed as an original contribution to the field of production systems. This algorithm enables one to obtain integer solutions, reduces slack on machines and enforces due dates on jobs. An equation is derived to compute the computational time savings when CMS algorithm is applied for single and multiple operation jobs, in comparison with aggregated approach.

Acknowledgements

The author wishes to express his sincere gratitude and appreciation to his research supervisor, Prof. L. Kops, for his thoughtful suggestions and constructive guidance which contributed to the present study, and for his continuous encouragement throughout the course of this work.

Sincere thanks to Dr. Micheal Mizrach, RAFAEL, Israel (on sabbatic leave at McGill University) for fruitful discussions on various aspects of this research.

The work presented in this thesis was conducted under the support of the Natural Sciences and Engineering Research Council of Canada which the author greatly appreciates.

Contents

List of Figures

List of Tables

List of Notations

j = Type of component (job), $j = A, B, ..., N$

i = Stages or time periods, $i = 1, 2, ..., P$

m = Type of machines, $m = 1, 2, ..., M$

l = Index for sections (operations), $l = 1, 2, ..., L$

r_{im} = Machine availability index

$$r_{im} = \begin{cases} 1 & \text{if machine } m \text{ is available at stage } i \\ 0 & \text{otherwise} \end{cases}$$

s_{jm} = Processing capability index

$$s_{jm} = \begin{cases} 1 & \text{if machine } m \text{ is capable of processing job } j \\ 0 & \text{otherwise} \end{cases}$$

X_{ijm} = Quantity of job j manufactured at stage i and at machine m

Q_j = Quantity of job j to be processed

T_{im} = Time available at work center m during the stage i

R_{jm} = Processing time of job j on machine m

S_{jm} = Setup time of job j on machine m

X_{ijml} = Qty. of job j manufactured during stage i at machine m on section l

T_{iml} = Time available at machine m on section l during the stage i

R_{jml} = Processing time of job j on machine m at section l

S_{jml} = Setup time of job j on machine m at section l

r_{iml} = Machine availability index

$$r_{iml} = \begin{cases} 1 & \text{if machine } m \text{ at section } l \text{ is available during stage } i \\ 0 & \text{otherwise} \end{cases}$$

s_{jml} = Processing capability index

$$s_{jml} = \begin{cases} 1 & \text{if machine } m \text{ at section } l \text{ is capable of processing job } j \\ 0 & \text{otherwise} \end{cases}$$

Chapter 1

Introduction

1.1 General Remarks

Job allocation decision is a very important activity in the production planning function of a manufacturing environment. It has particular implications to the machine tool utilization, factory efficiency , product quality and ability to meet due dates.

Because of their sensitivity to various performance measures, job allocation decisions are quite complex. Several methods and procedures are being developed to meet this challenging problem. As a result, there are a variety of solutions suggested, each analysing the problem from different perspectives and with different degrees of complexity. These procedures involve substantial amount of mathematical calculations and the use of computers help to a large extent in the development and application of solutions. Yet job allocation is one major function which has not taken full advantage of computerization in a manufacturing system. The current research in this area includes not only optimizing the job allocation function, but also integrating the other components of the manufacturing system with this function.

Job distribution on machine tools is a short range capacity decision assigning jobs, activities or tasks to available resources (machine tools). The result of this process is a timetable which describes exactly where and how much of each task should be performed. In addition, the schedule details regarding the commencement of each task and its duration, serve to obtain a better knowledge of the flow of jobs on the shop floor. The process of job allocation helps to provide production control by determining the load on machine tools and hence to identify the bottlenecks on the shop floor.

In the production planning function, job allocation should be clearly differentiated from job sequencing as the term 'scheduling' is often used to represent both these activities. The purpose of job allocation is to distribute jobs on machine tools such that the available capacity (of the machine tools) is efficiently and effectively used. The purpose of sequencing is to determine the order in which the jobs or tasks have to be performed on a machine tool in order to best satisfy some criteria like minimum makespan of jobs, minimum tardiness, *etc.* The job distribution function being an allocation decision, uses the information made available by the aggregate planning and facilities planning. This involves ıning for all production line over approximately one year ahead. Thus, job-resource allocation is the most constrained decision in the hierarchy of capacity planning decisions.

Conflicting objectives are often the characteristics of job allocation decisions: high efficiency, low inventory and completion of all the tasks or jobs within the due dates to ensure prompt customer service. Efficiency is achieved by a job allocation schedule which maintains high utilization of labour, equipment (machine tools) and space. Of course the schedule should seek to maintain low inventories, which may lead to low efficiency due to lack of available material for processing, or high setup times. Thus a trade-off decision in job allocation function between high efficiency and low inventory level is required. The primary aim of job-resource allocation decision is, therefore, to make a balance between conflicting objectives so as to arrive at a satisfactory performance of the overall manufacturing system.

Job allocation is not only influenced by several factors, but also highly dependant upon the decisions made elsewhere in the plant. Apart from these decisions, various factors constrain it by adding complexity. As a result, existing job allocation problem solution procedures are sub-optimal. This is because (1) the factors which influence the decision process are simplified in order to reduce the complexity (*e.g.*, setup times being considered as a part of processing times) and (2) the dimensions of the problem are reduced so that a computational solution is feasible. These approximations often produce unrealistic, unsatisfactory solutions. Therefore the importance of developing a simple, computable but realistic general solution procedure must be emphasized.

Since the job allocation function also influences several other components of the manufacturing system, it is imperative that it provides the most favourable conditions for the other activities. In this context, an optimum solution for job allocation is desired for a specific performance criteria. In existing procedures the search for optimum job allocation is the main cause of difficulties in computational solution. Thus the increase in complexity of job allocation problems leads to sub-optimal solutions. However, reasonable approximation is acceptable because optimal job allocation is difficult, often impossible. In most circumstances, the optimization tools are combined with a set of rules. Such a procedure produces acceptable results while meeting the defined criteria.

1.2 Motivation for the Study

The job distribution function helps in several aspects of decision making. It aids in determining the type and amount of resource (machine tools) requirement, additional capacity needed and thereby to identify the bottlenecks. It is essential that the job distribution in a manufacturing system is carried out efficiently because in many circumstances the necessity to procure additional facilities can be postponed by efficient loading of existing machine tools.

While scheduling problems of varying degrees of importance occur in all types of systems, they are particularly complex in job shops. A job shop is a processfocused production system that employs general purpose machine tools. Sometimes, in an open shop, production is commenced based on customers' orders. At other times, in the case of closed shops, it is based on inventory replenishment decisions. In a job shop a large number of different products are processed, each in a specific lot sizes. First come first served, random selection and other priority rules to load jobs on machines may not be applicable for all situations and sometimes they will prove unacceptable. This will result in delayed deliveries and unbalanced utilization of resources. A clear understanding of the job allocation procedures at the most detailed level will help to alleviate these difficulties.

Job shops are characterized by the variety of jobs that has to be processed on limited resources and the set of tasks performed on each component differs widely. This leads to an increase in the dimension of the problem when optimization is attempted using tools like linear (L.P.), or other mathematical programming. Moreover in many situations the time and cost parameters governing the problem are assumed to be constant over the entire scheduling horizon. This is unrealistic especially when the horizon spans a long period. Hence it is imperative that a new procedure be developed to partition these problems such that the random changes in the variables are accommodated. In a job shop, setup times are significant due to the variety of jobs that are processed and the schedule is dynamically asynchronous because jobs have different arrival and completion times. An efficient job allocation procedure should address these issues apart from optimizing the assignment.

A computerized production management information system called Manufacturing Resource Planning (MRP II) has been used with varying degree of success by some of the large production plants. Since job shops usually schedule production for specific customers, on specific set of machines, on the basis of the orders received, MRP II was found to be less applicable. Moreover, the Master Production Schedule (MPS) generated by the MRP II system leaves the machine loading function to the foreman and this often results in the under utilization of the machine tools. Hence it is necessary that the job allocation function is not only optimized but also flexible enough to be able to integrate with other activities of the operations scheduling thus automating the entire production planning module. Such a link would enhance the system performance constituting an important step towards the integration of information flow in computer integrated manufacturing (CIM).

1.3 Thesis Outline

The main objective of the thesis is to develop a new procedure to optimize the allocation of jobs on machine tools. The effectiveness of this procedure is brought about through a decomposition scheme which partitions the planning horizon into divisions, based the scheduled flow of jobs. The decomposition scheme leads to the development of an algorithm to solve job allocation problems and brings to light the possibility of integrating this with other production planning functions, thus enhancing CIM.

The importance of the planning function in a manufacturing environment and the role of job allocation in the capacity decision function are explained in chapter 1. The motivation for the study of these problems, how it aids in various decision making processes are also discussed.

Chapter 2 provides a critical review of the work done on the problem of job allocation problems. The chapter concludes with an evaluation of the limitations imposed by several of the existing solution procedures. It appears that because of the constraints and limitations involved in selecting the parameters, existing procedures tend not to represent a realistic job shop environment. In this part of the thesis an attempt is made to improve the situation.

Chapter 3 discusses the need for partitioning the scheduling horizon and the principles underlying a proposed scheme. The time decomposition scheme based on the scheduled flow of jobs is described in detail. The chapter also presents the dynamic optimization principle which functions within each partition to allocate jobs on machine tools. The benefits of the partitioning scheme and dynamic optimization principle are described.

The algorithm based on the partition procedure and the dynamic optimization principle which were described in the previous sections, is outlined in Chapter 4. The L.P. model for the job allocation problem is formulated and explained in detail. The functions of the algorithm and its constituents, namely, the linear program and heuristics are presented.

Chapter 5 contains the demonstration of the algorithm application to single operation scheduling problems. The significance of single operation jobs, the possible applications of the problem is described. The numerical examples illustrate the application of algorithm for a case of multiple jobs. Cases where both the starting and completion times of jobs are known and when only the completion times are known, are analyzed. A sample control program used to solve the L.P. and the summary of computational results are provided in appendices A and B respectively. The chapter concludes with the discussion of results where these are compared with those of standard practices. Computational aspects are highlighted.

Chapter 6 deals with the multiple operation job shop which is an extension of its single operation counterpart. The L.P. formulation and the flow chart of the algorithm, modified to suit multiple operation job shops, are provided. A numerical example demonstrates the application of the algorithm to such a case. The results bring to light the benefits of the algorithm over the Shortest Processing Time (SPT) rule.

The possibilities of incorporating the partitioning scheme and the optimization algorithm into intelligent manufacturing system is discussed in chapter 7. The various factors that should be considered while designing such an incorporation are also described.

Chapter 8 presents conclusions that may be drawn from the study reported in the thesis, as well as recommendations for future research work.

Chapter 2

Job Allocation - State of the Art Review

2.1 Literature Review of Studies on Job Allocation

The job allocation problem falls into the general category of job-resource assignment. A variety of taxonomic procedures have been provided for dividing job allocation problems. Based on the processing complexity Graves [1] classifies the problems as:

1. One operation, one processor
2. One operation, multi-processor
3. Multi operation job shop, tasks performed with identical precedence.

4. Multi operation job shop, tasks performed without any precedence constraints.

In a multiple operation manufacturing environment the job allocation problem has the most complex form: there are no restrictions on the processing steps for a task and alternative routings are allowed. This may be due to the completely general nature of the multi-operation job shop category; however, for production lines which are limited by a bottleneck section, simpler categories like single operation problem are important.

A standard approach to job-resource allocation problem is through the application of linear programming. Typical for this approach are the works by Horn [2] and Powell [3]. The former formulates the parallel machine tools, single operation job shop scheduling problem as an assignment model. The linear programming formulation with minimum mean flow time criterion converts the model into an assignment problem. A similar approach is followed by Powell where a simple linear programming model to minimize the productive cost is formulated.

For a case of two identical machines Lawler and Martel [4] search for allocation to satisfy minimum late job requirement. In a three step algorithm the jobs which are subjected to deadlines are schedules within the fixed time interval besides minimizing the number of late jobs. Since the procedure involves listing all possible schedules, it is a tedious approach and this becomes increasingly complex as the number of jobs increases. When M identical machine tools are used to process N jobs, McNaghton [5] provides a solution procedure in which a loss function is defined for each job. The value of this function is proportional to the time between the deadline and the actual completion of the job: it is zero if the job is processed within the deadline. The jobs are assumed to be preemptive. However no examples were provided to demonstrate the solution procedure nor was the complexity of the procedure discussed.

Linear programming deals with a static matrix of jobs and machines, which of course, represents a fundamental constraint, resulting not only in the departure from realistic situations, but also leading to increase in the size of problem. In order to tackle such large problems several techniques have been proposed to partition the planning horizon which will result in sub-problems of smaller size.

Perhaps the pioneering work in decomposing a general multi-stage linear programming problems was carried out by Dantzig [6]. The scheme consists of dividing the planning horizon into stages of equal time period and a status vector defined at the beginning of each division such that this provides the only connection between the stages. While the local optimum solutions are obtained by linear programming, global solutions are obtained by a master program using parametric programming. However, as Dantzig indicates, when the decisions are tied together by more than one variable this approach becomes tedious. An improvement in this procedure is suggested by Lasdon [7] in which the large number of columns of the L.P. model are handled by column generation through sub-problems instead of formulating a master program. Although it is claimed that the number of iterations required to solve the problem is reduced by a factor of two in comparison with that of Dantzig, the computational complexity is quite significant in both these cases.

Although it is not confined to job assignment problems Wagner [8] addresses the issue of solving linear programming problems containing multiple but identical time periods. The variables are assigned unique subscripts for each time periods and hence for p periods there will be np terms in the objective function, where n is the number of actual variables. Unfortunately the variables of all time periods are combined, and this approach will therefore not provide stage sub-problems representing each time period.

A partitioning scheme based on the arrival and completion times of jobs is another approach to solve job allocation problems. Nasr and Elsayed [9] suggest a partitioning scheme based on the finish times of jobs. All jobs are assumed to be non-preemptive and integer programming is used to solve the assignment problem. Integer programming solutions, however, are computationally intensive. To deal with the machine loading problem Jain *et al.* [10] divide the planning horizon into partitions of equal time durations, say months. In an integrated scheduling system the machine loading is carried out by an L.P. with an objective to maximize production and then employing heuristics to construct a feasible schedule for sequencing jobs at various work centers. The monthly sub-problems are however, aggregated and the L.P. is solved for the whole planning horizon.

Network graph technique is yet another method to solve the job distribution problems. Dorsey *et al.* [11] addresses the problem of assigning the jobs to several identical machines over a definite time horizon. The optimal solution is arrived at by an initial integer programming formulation with a minimum production, inventory and backorder costs criteria. Subsequently the model is converted into a network graph problem and solved. However, from the computational view point solving integer programming problems is tedious. Iwata *et al.* [12] treats the problem of assigning jobs to machines using network graph and an algorithmic procedure consisting of two dispatching rules, namely: earliest finishing time (EFT) and its extension which considers alternate routings, EFTA. These proposed rules provide shorter makespan (makespan of a job is the time between its entry into the shop and its departure after processing) than other dispatching rules, the schedule is not however constrained by the due dates of jobs. As a result the setup times are added to the schedule after the assignment is carried out. When setup costs are considered in the schedule, mixed integer programming is used by Dilts [13] to solve the combined problem of lot sizing and job allocation.

2.2 The work of M. Alaee at McGill University

M. Alaee [14], in order to formulate a procedure to solve job allocation problems, developed a method to partition the scheduling horizon based on the flow of jobs (*i.e.*, input of jobs and their departure after processing). According to this decomposition scheme the partitions are defined when a job enters the shop or when it leaves after processing. Thus within the time duration of the partition the product-mix on the shop floor remains constant.

The jobs within each partition are denoted by unique subscripts. When the same job moves to the next partition, it assumes a different subscript. The job allocation is carried out using an linear programming model with an objective to minimize makespan. The constraints for this L.P. are the time available on the machine tools during each partition and the total number of jobs processed

Alternate routings are allowed *i.e.*, a job can be processed on different machines. However, the machines are not identical and hence the processing times of the same job on different machines are not the same. The setup times are accounted for in the schedule by subtracting the setup time of each job a priori from the available time on each machine. Thus irrespective of whether the job is loaded on the machine or not, setup times are deducted.

The solution procedure aggregates all the partitions: the L.P. contains the terms for all the partitions, both in the objective function and in the constraints. This precludes full exploitation of the advantages of the partitioning procedure fully. The scheme presented in this thesis is based on this concept; in addition setup times are taken into account. An algorithm is developed subsequently to solve the sub-problems sequentially.

2.3 Concluding Remarks

In the proposed allocation models, one or more of the following drawbacks are noticed:

1. In some of the works it is assumed that setup times are part of processing times. However, job shops are characterized by a variety of jobs which consume a considerable amount of setup time. It is difficult or sometimes impossible to incorporate setup times/cost along with processing time/cost.

2. When several jobs are considered together, they are assumed to be available simultaneously and have a common completion time.

3. Alternate routings are not permitted and hence there is no chance of preemption. The jobs have to wait until the appropriate resource is free to load. However, in reality the resources in a job shop are flexible enough to handle a variety of jobs and hence there are alternate ways to process a job.

4. Most of the work done in the field of job shop scheduling, both for single and multiple operations, assume that processors are identical.

5. In cases where quantity processed is the optimization criterion, obtaining integer solutions was found to be difficult. Some authors uses integer programming models for which the computational effort required is high, particularly when the size of the problem is large.

6. Scheduling problems are solved for definite time horizon. So all the parameters within that horizon are assumed to be constant.

7. When network graph technique is used in conjunction with heuristics to solve scheduling problems, it was noticed that the computational effort increases for large scale problems. This is because the technique involves listing of all possible schedules and selecting the best among them, for a given criterion.

   Similar computational difficulty is experienced when dynamic programming approach is used.

Chapter 3

Time Decomposition and Dynamic Optimization

3.1 Necessity for Decomposition

The problem that is often encountered in linear programming (L.P.) formulations of job allocation is that the jobs are assumed to be available at time zero and all jobs have common completion times. Since job shops are characterized by parts that arrive at different times and have different due dates, this assumption may not hold good on most of the occasions. When L. P. formulation is attempted accounting for such job arrivals and departures, the number of variables increase, making the computational procedure difficult. The new partitioning scheme should address these drawbacks and also include the new jobs in the schedule as they arrive into the shop.

Maintenance for machine tools in a job shop can be scheduled at specific time periods within the planning horizon. The machines will not be available for processing the jobs during maintenance periods and a job allocation schedule should accommodate this by taking into account the starting times and duration of scheduled maintenance. When the planning horizon is large, it is found to be difficult to incorporate time periods during which the machines are not available.

For practical job allocation problems, computer storage may be an important consideration. These problems may sometimes require matrices of a size that tax even large, modern computers. In general, all these job allocation problems span a definite horizon and if such a problem could be divided into smaller problems, then the computational effort can be reduced. However, while partitioning the scheduling horizon in this fashion, the sub-problems should be linked in such a manner that there exists an effective coordination between them.

It should be noted that the parameters such as processing time, setup time and the associated cost, tend to decline as the learning curve progresses. Another reason for this decline could be the technological advancement of operations methodology. A linear programming formulation must however assume that these parameters are constant for a given time horizon. In the short run these may not affect the performance of the job allocation solution, as the time elements can be assumed to be fairly constant. However, if the time elements and the costs associated are variable or discontinuous over the full range of scheduling horizon, then the assumption of linearity will not provide a realistic solution. Hence, it is often practical to segment an element of the time horizon and assign different time and cost values to each of the segments thereby approximating the behaviour of variables which are not constant. The immediate effect of such decomposition will be the increase in the size of the variable matrix and hence the computing time required to solve the problem.

Some progress has been made in developing methods for solving integer variable problems. While it can be stated that theoretically these methods will ultimately yield integer solutions, even small problems have been found to require great many iterations and huge computer memories. Real life problems demanding integer solutions should be solved using approximations such that the solution procedure is economical.

3.2 Time Decomposition Scheme

The jobs (new materials and components) arrive in the shop for processing on a suitable and available machine tool according to their respective process plan and schedule defined by the order releases and due dates. When considering the allocation of jobs on the machines, one should notice that their arrival and due dates are spread throughout the planning horizon. This means, the job mix on the shop floor is changing with time. As mentioned earlier, this situation complicates the L.P. formulation, which is a simple way to find the optimum solution for job-resource allocation problems. There are, however, time periods within the planning horizon, when the job mix on the shop floor remains constant Such periods occur when no new jobs arrive or a finished one leaves the shop floor. It is therefore possible to decompose the time of the planning horizon into constant job-mix stages. Any job arrival or departure means a change in the product mix on the floor, and therefore such an event marks the beginning or end of the stage. A job arrival to the shop floor or departure of a finished component is an *event*. A *stage* is defined as the time interval between two events, during which the product mix is constant. A stage can also be formed by consecutive arrivals or consecutive departures of any jobs. Thus the planning horizon can be divided into constant job-mix stages (CMS) based on the scheduled flow of jobs. It is apparent that with N jobs considered, there could be upto $2N - 1$ stages. The concept of constant job-mix stages (CMS) formed by the time decomposition method is illustrated in Fig 3.1.

The time and cost factors are assumed to be constant within a stage. This is valid as it provides a reasonable approximation for any existing variables whose values are changing with time, and also allows for a step change at an event occurrence.

The division of the scheduling problem into stages (time periods when the job-mix is constant) has its advantages:

1. By partitioning the scheduling horizon based on the flow of jobs, each division has a constant mix of jobs. The dimension of the L.P. formulation is reduced because of this partitioning scheme.

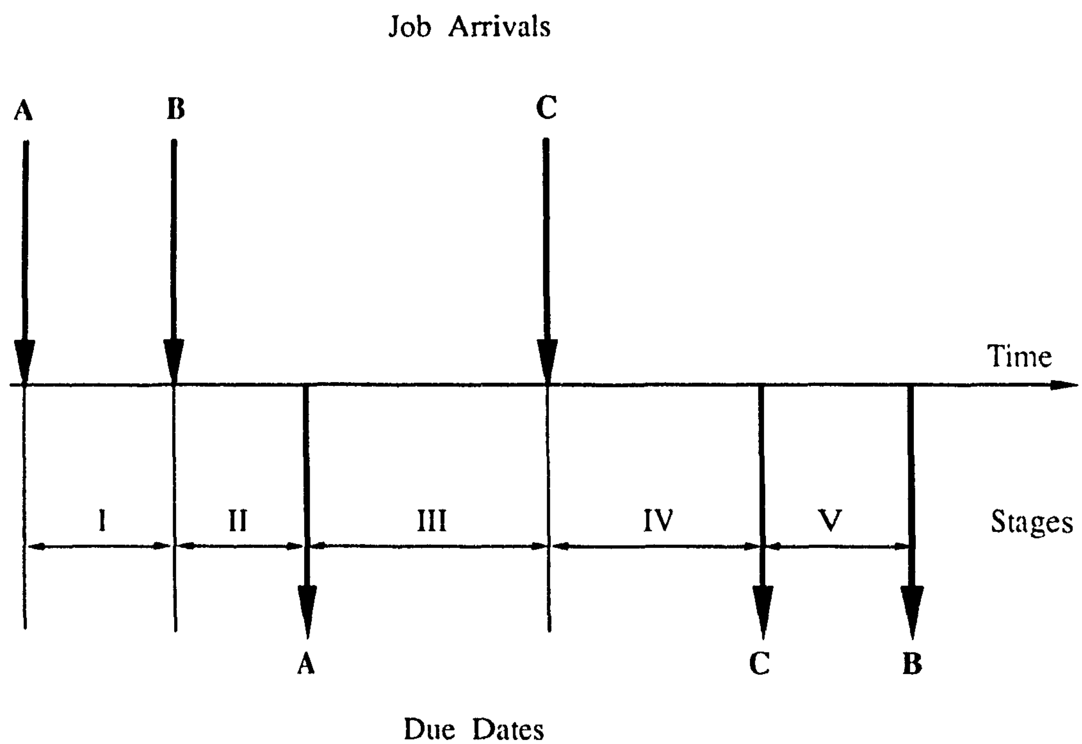

Fig 3.1 Schematic representation of constant job-mix stages

2. Factors like machine availability and alternate routings for jobs can be incorporated accurately as these are defined within each partition.

3. Since allocation of jobs on machine tools is carried out as soon as the components arrive on the shop floor, it is easy to incorporate setup times, and this provides a realistic schedule.

4. From a computational viewpoint, it is profitable to partition the scheduling horizon and solve the sub-problems, since the computational effort and the time required to solve an L.P. is proportional to the size of the problem

3.3 Dynamic Optimization Principle

The decomposition of the planning horizon into constant job-mix stages, opens the possibility of using L.P. to optimize the job allocation on machine tools following the time-based schedule. The optimization can be carried out within each stage, with the output of one stage forming the input for the subsequent stage, in addition to the basic positive or negative input defining the beginning of this new stage. The job allocation solution of the L.P. indicates the quantity of jobs to be processed on the available machine tools. This is deducted from the total demand quantity in order to obtain the remaining quantity to be processed during the subsequent stages and this forms the input for the next stage L.P. Considering that such a link between the stages continues throughout the planning horizon, from stage to stage, the dynamic nature of both the problem and the optimization procedure is apparent, and hence the procedure can indeed be called as *dynamic optimization.*

As will be seen in the following chapters, the optimization of job allocation within each stage provides a more compact schedule. While this leads to compressing the schedule towards the due dates with backward decomposition, it provides a schedule compressed towards the starting dates when forward decomposition is carried out. The former corresponds to the just-in-time philosophy with reduction of slack and work-in-process (WIP) inventory as its effects. The latter procedure provides early shipment dates for orders and permits loading additional jobs on machine tools during the spare time.

Chapter 4

The Job Allocation Algorithm

4.1 Introduction

Based on the time-partitioning scheme and dynamic optimization principle, an algorithm is developed to optimize the allocation of jobs on machine tools. The algorithm consists of two parts; the first part is the allocation problem which is formulated as a linear programming model and the second, a heuristic which will account for various functions, like: incorporation of setup times, obtaining integer solutions *etc.* The job allocation procedure can be viewed as a decision making process, in which the quantity of each job to be loaded on a particular machine tool is to be determined. Consequently, the decision making process is exercised over a discrete set of events in the scheduling horizon. In each sub-problem, the set of jobs (operations) available for allocation is considered and the assignment for those jobs to machines is obtained with consideration given to local constraints like machine availability, capability and time available on machines.

4.2 L.P. Formulation

The job allocation problem within each stage is formulated as a linear allocation model, with an objective to maximize the production rate of all the machines. The constraints for the L.P. are:

1. The production quantity should match the demand,
2. The time of processing of all the components on every machine cannot exceed the stage time period.

Some machines might not be available throughout the stage duration. When a machine tool is not capable of processing a particular job and/or when a machine is not available, the machine availability index r_{im} and processing capability index s_{jm} guarantee that the objective function will contain only the valid non-zero terms. In all the equations found in the L.P. formulation which follows, the subscript i denoting the stage is unique; hence there will be P stages representing the entire problem. The L.P. formulation for a single stage job allocation problem is as follows:

$$Max \quad \{\sum_{m=1}^{M}(\sum_{j=A}^{N} r_{im}s_{jm}X_{ijm})\}$$

subject to:

$$\sum_{m=1}^{M} r_{im}s_{jm}X_{ijm} \leq Q_j \qquad j = A, B, ..., N$$

$$\sum_{j=A}^{N} r_{im}s_{jm}R_{jm}X_{ijm} \leq T_{im} \qquad m = 1, 2, ..., M$$

$$X_{ijm} \geq 0 \qquad j = A, B, ..., N$$

$$m = 1, 2, ..., M$$

The first set of constraints state that the quantity of any job processed should not exceed the demand quantity. For components which are due by the end of the stage the demand is met by equating the quantity processed to the remaining quantity *i.e.*, in the demand constraint the $\leq$ operator is changed to $=$. The second constraint set requires that the machine hours utilized in any stage should not exceed the capacity available on each machine tool. The last one is a non-negativity constraint for the quantity of any job processed in a stage.

The linear programming formulation of the allocation model is somewhat similar to the generalized transportation problem. The constraints are not equations but are inequalities which must be augmented by slack variables. Although the solution of a transportation model yields integer solutions directly, the allocation tableau is large and sparse. Storage and/or data accessing problems may arise.

The allocation model belongs to a special class of transportation problem called the multi-divisional problems which exhibit a special block angular structure (Hillier and Lieberman [16]). Each smaller block contains coefficients of the constraints for one sub-problem (one sub-problem represents one stage). Since the divisions operate with considerable autonomy, the problem is decomposable into separate problems, where each division is concerned with optimizing its own operation. The decomposition scheme makes use of this special structure in order to solve the allocation problem. Moreover, the scheme also provides a schedule which is not bounded by time. Such an open ended schedule permits loading of jobs on a continuous basis, thus enabling realistic and short term scheduling.

4.3 Setup Time Incorporation

Setup times are incurred whenever a machine tool starts to process a new component. If the lot is split among many machines, then all those machine tools undergo a setup. The job allocation problem is more difficult to solve when setup times are considered. For a given problem and ignoring the setup times, it is easy to test whether or not a feasible solution exists by comparing the cumulative demands to the cumulative capacity for all time periods and facilities. With setup times taken into account, the scheduling problem under consideration is found to be NP-complete *i.e.,* cannot be solved in polynomial time (Garey *et al.* [15]).

Setup times are significant parameters in a job shop manufacturing environment. This is due to the fact that a variety of components are processed in a job-shop. Setup times in job shops take more time than in other kinds of manufacturing environment. This is because job shops are flexible in processing a variety of complex jobs. Obviously, this will have an impact on meeting the due dates, especially in cases where the setup times are more than 500 times the processing times, which is often the case.

The setup times are incorporated after the initial allocation of jobs on machine tools is carried out. This is because, in order to assign setups, it is essential to know which jobs are loaded on which machines. After assigning the appropriate setup times on machines, the job allocation schedule is generated once again to obtain the final solution.

4.4 Other Functions of the Algorithm

The other functions of the algorithm, apart from setup time incorporation are: providing integer solutions, updating demand quantity, reducing slack on machines, enforcing due dates for jobs and providing a link between the different stages. Since the job allocation is solved using linear programming, the solution contains real variables and this is unsuitable if only integers are admissible. In order to obtain integer values, the variables that enter the L.P. solution are rounded off to the nearest smaller integer. In other words, the variable is truncated. A cumulative effect of this might build slack on the machine tools. However, the CMS algorithm also accounts for slack reduction by reassigning jobs on machines where slack exists. The necessary condition for this, of course is, that the required setup exists on these machines for a particular job.

Since the stage time durations are considerably less than that of the planning horizon, the total demand quantity may not be processed within a stage. The CMS algorithm also updates the demand quantity of jobs at the end of every stage by deducting the quantity that has been processed during the stage. The remaining demand quantity to be processed in the subsequent stages, is the parameter which provides the link between the stages. The CMS algorithm also enforces the due dates for jobs which are due by the end of the following stage, by changing the $\leq$ operator in the quantity constraint to $=$.

4.5 Algorithm Procedure

Once the starting dates and due dates are known, the stages can be determined. The L.P. model of the job allocation problem for the first stage is then formed and solved. Based on the job allocation to machines in this stage, the setup times are deducted from the assigned time for each machine and the L.P. model is iterated. If there is no change in job allocation, then the quantities processed on each machine are truncated to integers. Slack reduction on machines which have idle time and the due date enforcement is then carried out by reallocating the jobs to machines. The demand quantity is then decreased by the quantity processed during the stage and constitutes the quantity to be processed in the subsequent stages. The quantity of all components processed during the next stage is equated to the demand quantity, for all jobs which are due by the end this stage. These procedures are iterated until the job allocation in all the stages is completed. The following algorithm provides the steps involved in the solution procedure.

Step 0: Formulate the Linear Programming (L.P.) model for the first stage .

Step 1: Solve the L.P. model.

Step 2: Based on the assignment of jobs, deduct setup times from the assigned times. If setup already exists on the machine, then do not deduct setup time.

if $X_{ijm} > 0$ and $X_{i-1jm} > 0$, then $S_{jm} = 0$

for job j having setup on machine m. Otherwise,

$(T_{im})_{new} = (T_{im})_{old} - \sum_{j=A}^{N} S_{jm}$

Here, $(T_{im})_{old}$ refers to the available time on machine m during stage i, before setup time incorporation.

Step 3: Solve the modified linear program. If the solution obtained indicates the same job allocation pattern as observed in step 1, the solution is chosen.

Otherwise go to step 2[1].

Step 4: The quantity processed during the stage is integer truncated.

Step 5: If the current stage is the last stage for component j, and if the quantity processed is less than the demand, then distribute job j on machines which accept loads other than j.

If after rounding off, the following conditions exist:

1. $\sum_{m=1}^{M} X_{ijm} \leq Q_j$
2. i is the last stage for component j.
3. $T_{im} - \sum_{j=A}^{N}(R_{jm} X_{ijm}) \leq R_{jm}$

then,

$$(T_{im})_{new} = (T_{im})_{old} - (\alpha \times R_{jm})$$

$$(X_{ijm})_{new} = (X_{ijm})_{old} + \alpha$$

for machines which accept load components other than j.

where $\alpha = (Q_j - \sum_{m=1}^{M} X_{ijm})$.

[1]Note that the steps 2 and 3 in the algorithm will not loop Once the setup time is incorporated, the available time on machines is reduced, thus fewer jobs are allocated on the machines during the next iteration The pattern of allocation often remains the same causing the algorithm to go to step 4. Looping can happen only when the stage duration is less than the setup time. But when this happens, either the job processing is postponed to subsequent stage or the entire stage duration is consumed for setup on the machine.

Step 6: If there is slack on any machines, load job for which setup already exists on that machine; *i.e.*, if the following conditions exist:

$$T_{ij} - \sum_{j=A}^{N} X_{ijm} R_{jm} \geq R_{jm} \text{ and,}$$

$$\sum_{m=1}^{M} X_{ijm} \leq Q_A$$

then,

$$(X_{ijm_{new}} = (X_{ijm})_{old} + \alpha$$

$$\text{provided,} \quad (X_{ijm})_{new} \times R_{jm} \leq T_{im},$$

$$\text{where} \quad 0 < \alpha < Q_A - \sum_{m=1}^{M} X_{ijm}$$

Step 7: If this is the last stage, then terminate, else proceed to step 8.

Step 8: For all X_{ijm} in the solution, subtract the quantity processed from the demand quantity to obtain the remaining quantity to be processed in the next stage[2].

Step 9: For any component which has due date during the end of next stage the $\leq$ operator in the time availability constraint is changed to $=$.

Step 10: Form the L.P. model based on step 8 and step 9. Go to step 1.

[2]The steps 9 and 10 belong to the next stage.

Chapter 5

Application of Algorithm to Single Operation Job Shops

5.1 Analysis of the Problem

The single operation, multiple job scheduling problem is an important generalization of a job shop. The task or operation requires one processing step which may be performed on any of a number of similar machine tools. In a production line, the different operations performed on a particular job need not consume the same amount of time. A slower operation in an unbalanced line constitutes a bottleneck. The output of a bottleneck (slowest) operation decides the output of the entire production line. In such situations, it may be sufficient to consider only the bottleneck operation and analyze it If the cause for bottleneck can be attributed to inefficient loading patterns, job allocation can be optimized on this operation using the CMS algorithm. Besides increasing the throughput, this exposes other existing problems in the production line. This aids in solving the problems, thus enhancing the throughput and efficiency of the overall shop.

The theoretical insight obtained by analyzing the single operation job allocation problem is helpful in tackling more complex problems. These serve as building blocks for understanding the decision processes involved in multiple operation job assignment problems. Individual optimization of single operation jobs may provide a global sub-optimal solution.

5.2 Problem Definition

The classic multi-item, single operation, multi-machine tool job allocation problem can be stated as follows.

1. There are N items to be scheduled.
2. There are M machines which are not identical. These machines may or may not be capable of processing all the N jobs.
3. Unit processing time R_{jm} is incurred when the machine m ($m=1, 2, 3, \ldots, M$) processes job j ($j=A, B, C, \ldots, N$).
4. Setup time S_{jm} is incurred whenever a machine m starts processing a new job j. Setup times are unique for a particular job on a particular machine.
5. Due dates are specified for each type of job.

6. The demand quantities $Q_j (j = A, B, C, ..., N)$ for all jobs are known.

7. All machines may or may not be available throughout the planning horizon.

5.3 Scheduling from the Starting Times

In situations where job arrival times are known, scheduling jobs from the start dates is prudent as it determines the actual completion date of the operations. The jobs are processed as soon as they arrive on the shop floor. This provides a loading pattern in which the schedule is compacted towards the start dates. Early shipment dates for all the orders is an obvious result of such schedule.

5.3.1 Numerical Example

A case of four jobs to be processed on four machines is considered to demonstrate the application of the algorithm described in Chapter 4. Each job undergoes only one operation and there are alternate machines to perform each operation. All jobs are to be processed in specific quantity and their arrival and due dates are known in advance. The processing time of different jobs on machines is provided in Table 5.1. The demand quantity of jobs and their due dates are given in Table 5.2 while the availability of different machine tools during the planning horizon is shown in Table 5.3. The constant job-mix stages for this problem is illustrated in Fig 5.1. The problem is to allocate jobs on machine tools in such a way that the output is maximized.

Table 5.1: Processing times of jobs.

Machine	*Proc. time (min)*			
m	A	B	C	D
1	3	-	2	6
2	-	8	3	7
3	5	10	-	12
4	10	-	4	9

The algorithm provided in the previous chapter is applicable for a general case. Moreover in the current example the setup times are not considered. Hence, some of the steps may not be applicable for the case under consideration and hence are omitted.

Table 5.2: Demand quantities, arrival and due dates.

Job j	*Quantity* Q_A	*Arrival* *(min)*	*Due date* *(min)*
A	420	0	3600
B	240	900	2500
C	1800	1200	4400
D	400	3000	6000

Table 5.3: Availability of machine tools.

Machine m	*Available* *(min)*	*Not available* *(min)*
1	0 - 6000	0
2	0 - 6000	0
3	0 - 2000 3000 - 6000	2000 - 3000
4	2400 - 6000	0 - 2400

Stage 1

Step 0: First stage L.P. is formulated as follows:

$$Max \quad (X_{A11} + X_{A13})$$

Subject to:

$$X_{A11} + X_{A13} \leq 420$$

$$3X_{A11} \leq 900$$

$$5X_{A13} \leq 900$$

Step 1: Solution for this L.P. is $\{X_{A11} = 300, X_{A13} = 120\}$

Step 10: The next stage L.P. is formulated as follows:

$$Max \quad (X_{B22} + X_{B23})$$

Subject to:

$$X_{B22} + X_{B23} \leq 240$$

$$8X_{B22} \leq 300$$

$$10X_{B23} \leq 300$$

Stage 2

Step 1: Solution $\{X_{B22} = 37.5, X_{B23} = 30\}$

Step 4: $\{X_{B22} = 37, X_{B23} = 30\}$

Step 8: $Q_B - \{X_{B22} + X_{B23}\} \Rightarrow 240 - 67 = 173$

Step 10: The next stage L.P. is formulated as follows:

$$Max \quad (X_{B32} + X_{B33} + X_{C31} + X_{C32} + X_{C34})$$

Subject to:

$$X_{B32} + X_{B33} = 173$$

$$X_{C31} + X_{C32} + X_{C34} \leq 1800$$

$$2X_{C31} \leq 1300$$

$$8X_{B32} + 3X_{C32} \leq 1300$$

$$10X_{B33} \leq 1300$$

$$4X_{C34} \leq 1300$$

Stage 3

Step 1: Solution $\{X_{B32} = 93, X_{B33} = 80,$

$X_{C31} = 650, X_{C32} = 185.33, X_{C34} = 25\}$

Step 4: $\{X_{B32} = 93, X_{B33} = 80,$

$X_{C31} = 650, X_{C32} = 185, X_{C34} = 25\}$

Step 8: $Q_C - \{X_{C31} + X_{C32} + X_{C34}\} \Rightarrow 1800 - 860 = 940$

Step 10: The next stage L.P. is formulated as follows:

$$Max \quad (X_{C41} + X_{C42} + X_{C44})$$

Subject to:

$$X_{C41} + X_{C42} + X_{C44} \leq 940$$

$$2X_{C41} \leq 500$$

$$3X_{C42} \leq 500$$

$$4X_{C44} \leq 500$$

The problem is solved for the remaining stages in the similar fashion.

5.3.2 Results and Discussion

The linear programming problem for all the stages was solved using IBM's MPSX software on a IBM 3090 mainframe. The final results obtained by the application of CMS algorithm are provided in Table 5.4 and this shows the assignment of various jobs during each stage. As the problem could be solved with all stages aggregated together using a linear programming formulation, it is considered as a basis for result comparison. The objective function and the time availability constraints of such an aggregated formulation are obtained by combining those of the stages. Table 5.5 and 5.6, and Figure 5.1, show a comparison between the CMS algorithm and the aggregated approach.

As can be seen from Fig 5.1, the job distribution on machine tools is compressed towards the starting times when CMS algorithm is used, whereas in aggregated approach the jobs are processed in rather a discontinuous manner.

The decrease in makespan of jobs, as in Table 5.5., is the effect of such schedule compression: a maximum of 68% reduction in makespan is indicated for individual jobs. The obvious result of this is shorter completion time of jobs (Table 5.5), due to the fact that the jobs are loaded on any available machine tool as soon as they arrive on the shop floor.

The important benefit of using the decomposition scheme and optimization is an increase in utilization of the machine tools. Machine tool utilization is defined as the ratio of the time when the machine is busy to the time between the entry of first job and completion of the last job on the machine. Table 5.6 shows a comparison of utilization between CMS algorithm and aggregated approach. The maximum output objective of the linear program ensures that within each stage the jobs are loaded on the available machine tools and no machine is left idle if any job which the machine is capable of processing, is present on the shop floor. Thus the machine tool utilization is maximized in every partition, providing a cumulative effect of overall maximization of machine tool utilization.

Since the schedule is compacted towards the starting dates and makespan of jobs is reduced, free time is made available on machines and this can be used to process other jobs. This also helps to provide early shipment of orders.

From a computation time viewpoint, the partitioning of the time horizon and optimizing the sub-problems, proves to be profitable. For a linear program the computational time roughly equals the third power of the number of functional constraints (Hillier and Lieberman [16]). When a problem is decomposed, the sub-problems contain far fewer constraints than the large problem. Thus the computational time of all the sub-problems added is substantially reduced in comparison with that of the large problem. For a general N job M machine problem, when all the P stages are aggregated the number of constraints present will be $(N + \sum_{i=1}^{P} M_i)$, while that of single stage is $M_i + N_i$. For P stages, the number of constraints present will be $\sum_{i=1}^{P}(M_i + N_i)$. Hence, the savings in computational time when the problem is solved using CMS algorithm is

$$R_T = \frac{(N + \sum_{i=1}^{P} M_i)^3}{\sum_{i=1}^{P}(M_i + N_i)^3}$$

For the example case the number of constraints present in the aggregated LP is 28 while that of the 6 stages are $\{3, 3, 6, 4, 6, 5\}$. The savings in computational time will be:

$$R_T = \frac{28^3}{3^3 + 3^3 + 6^3 + 4^3 + 6^3 + 5^3} \Rightarrow 32.5$$

Hence the algorithm approach is 32 times less time consuming than the aggregated approach. This shows the implication of partitioning on the computational time effort required, which will be even more pronounced when problems of larger size are considered.

5.4 Scheduling from the Completion Times

Scheduling from completion times presents two important advantages:

1. It provides a schedule which is compacted towards the due dates.

Table 5.4: Final allocation of Jobs

Machine	*Stage 1*	*Stage 2*	*Stage 3*	*Stage 4*	*Stage 5*	*Stage 6*
1	$X_{A11} = 300$	†	$X_{C31} = 650$	$X_{C41} = 250$	$X_{C51} = 300$	$X_{D61} = 133$
2	†	$X_{B22} = 37$	$X_{B32} = 93$ $X_{C32} = 185$	$X_{C42} = 166$	$X_{C52} = 99$ $X_{D52} = 43$	$X_{D62} = 108$
3	$X_{A13} = 120$	$X_{B23} = 30$	$X_{B33} = 80$	‡	$X_{D53} = 50$	
4	‡	‡	$X_{C34} = 25$	$X_{C44} = 125$	$X_{D54} = 66$	

† = Machine not capable of processing a particular job

‡ = Machine not available

Table 5.5: Comparison of completion time and makespan

Jobs	*Compl. time (min)*		*Makespan (min)*		
	Stage	*Aggr.*	*Stage*	*Aggr.*	*reduction %*
A	900	2860	900	2860	68.5
B	2000	2500	1100	1600	31.3
C	3600	4400	2400	3200	25.0
D	4100	6000	1400	3000	53.3

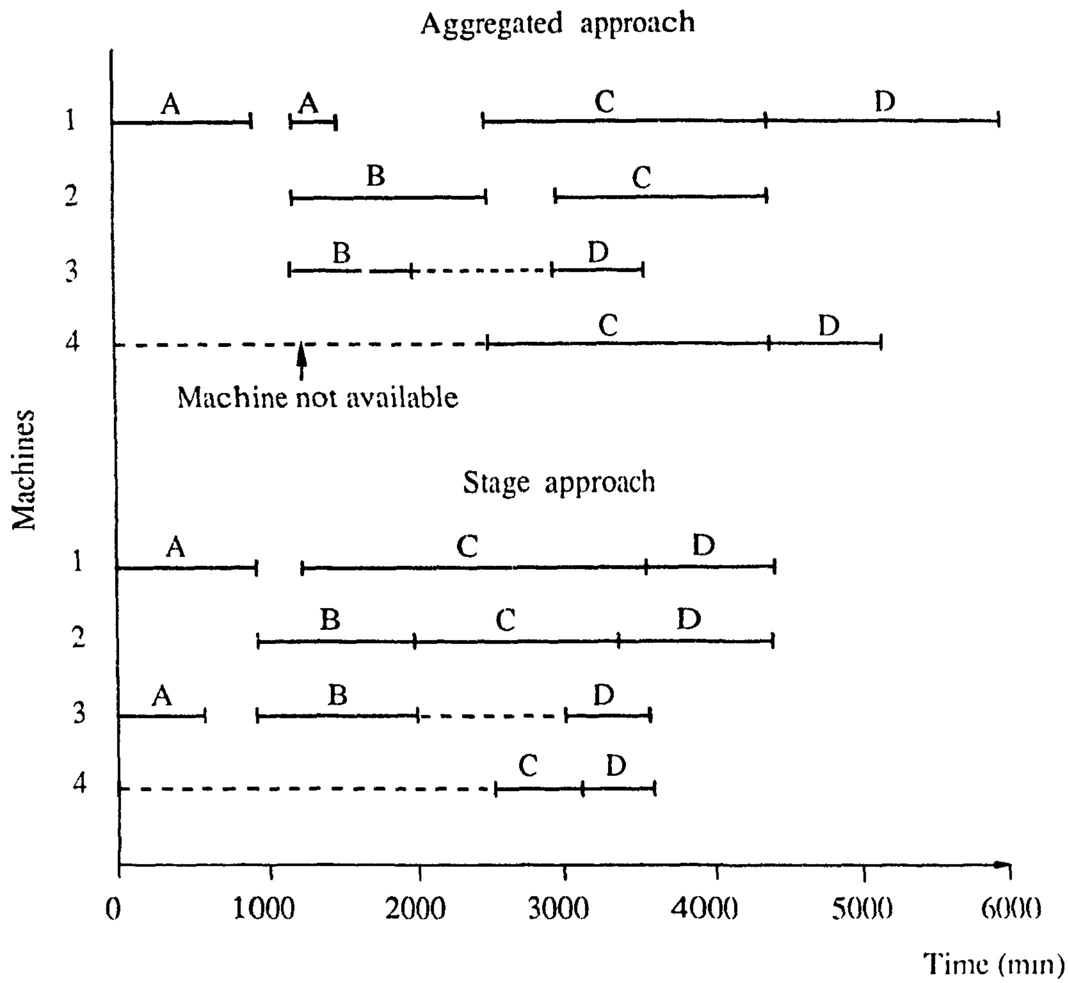

Fig 5.1. Allocation of jobs on machine tools based on Aggregated approach and Stage approach

Table 5.6: Comparison of utilisation

Machines	*Utilisation %*		*Increase*
m	*Stage*	*Aggr.*	*Stage/Aggr.*
1	93.1	77.5	1.2
2	78.3	44.9	1.7
3	88.5	53.1	1.6
4	99.5	96.2	1.0

2. Late start dates is obtained for all jobs. This corresponds to just-in-time (JIT) philosophy.

As a result of JIT schedule work-in-process (WIP) inventory is reduced and clearly this leads to a lot of saving especially for components which have high raw material value and for those components for which the value increases rapidly as the operations progress. Since the schedule is compacted, the production process can be organized such that inventories are strategically placed throughout the process. Then, by carefully reducing these inventories certain production problems are exposed. Solving these reduces costs and lead times and improves quality. The following example demonstrates how the partitioning scheme can be applied to cases where the starting dates are unknown. Comparison of results with that obtained by applying SPT rule is also provided.

5.4.1 Numerical Example

Let us consider a problem of allocating 8 jobs on 5 machines. The processing times and the setup times of various jobs on different machines are provided in the Table 5.7 and Table 5.8. The '†' symbols in these tables indicate the machine is *not capable* of processing the particular job. For the problem at hand, the machines are assumed to be *available* throughout the scheduling horizon i.e., there are no downtimes. Table 5.9 presents the demand quantities of different jobs and their stipulated due dates.

In this example, it is assumed that only the due dates are known, while the starting times are *not known* initially. Hence the following backward scheduling procedure is adopted to determine them. Starting from the due date of the last component, jobs are assigned to machines using SPT rule without preemption. If two or more jobs compete for the same machine, the job which has the least processing time is assigned to the machine. The remaining jobs are distributed to other available machines (based on SPT rule). When this procedure is carried out for all the jobs, it generates a loading pattern with the late start dates which serve as a reference to define the stages (Fig 5.4). The SPT rule was selected because, this rule ensures high machine utilization and performs better than other priority rules in situations where jobs arrive in the shop at different times. This provides the least favourable conditions to which the results of the CMS method are compared.

Table 5.7: Processing times of jobs.

Machine	Processing time (min)							
	A	B	C	D	E	F	G	H
1	2.0	†	3.5	4.0	2.0	3.0	†	3.5
2	1.7	3.0	†	†	†	2.8	2.7	3.2
3	†	3.5	2.7	4.3	2.4	†	2.5	†
4	2.1	3.8	3.9	3.7	†	3.1	3.0	†
5	1.5	†	3.0	4.2	2.3	3.5	†	3.1

† = Machine not capable of processing the job

Table 5.8: Setup times of jobs on machines.

Machine	Setup time (min)							
	A	B	C	D	E	F	G	H
1	200	†	195	145	150	155	†	110
2	235	240	†	†	†	160	175	140
3	†	230	210	100	120	†	200	†
4	200	220	190	180	†	150	160	†
5	210	†	200	100	145	120	†	145

† = Machine not capable of processing the job

Table 5.9: Demand quantities and due dates.

Job	*Quantity*	*Due date (min)*
A	1500	5000
B	2000	7000
C	1500	9000
D	1000	9000
E	5000	12000
F	3000	15000
G	3000	16000
H	4000	17000

5.4.2 Results and Discussion

Results of the example are given in Figures 5.2 to 5.4 and Tables 5.10 to 5.13. As can be seen from the figures, the method of dynamic optimization (CMS) using decomposition into constant job-mix stages generates a considerably more compressed schedule than normal backward scheduling based on SPT rule. The late starts are pulled towards the due dates, corresponding to the just-in-time philosophy.

The pattern of job allocation on machines using the CMS method is shown in Figure 5.2. It provides a good illustration of the effect of pull logistics. The corresponding numerical data for the late starts are given in relation to the due dates, in Table 5.10.

The inherent feature of job allocation which is visible in Figure 5.3 is the distribution of setup times. While in the common practice of assigning a job to machine with SPT, only one setup is needed for the full processing time of this job (Figure 5.3 a), this is not the case with the CMS method. In the new method setups are assigned to various machines according to the optimized solution, in which setup times are taken into account. In this procedure the setups may be repeated on the same machine in various stages if another job has to be processed in the meantime to meet the due date. Improvement in sequencing is however, possible.

The compressed schedule, in which the distributed setups form an integral part, permits an increase machine tool utilization (Table 5.11) and provides considerable free capacity concentrated early in the schedule; this free capacity can be used to process additional jobs. The total free capacity of the shop is increased by nearly 50 % (Table 5.12). The utilization of machines is calculated as the ratio of time a machine is busy (including the setup times) to the total time-span in the shop between the start of the first job to the last due date and expressed as percentage. This time- span is different for SPT and CMS schedule. As can be seen from Figure 5.3 and Table 5.11, machine # 2 is the busiest one in both the cases; it has 100% utilization in CMS case, in which case its busy time is equal to the total shop time-span. The utilization of remaining machines would have been higher than indicated 78% to 86.3%, if not for the out of range earlier start of machine # 2, which defines the denominator of the utilization ratio. Table 5.11 indicates that the increase of utilization in the case of CMS over SPT, while varying from machine to machine, reaches 1.87 (*i.e.*, 87% higher) for machine # 4. The increase of the machine utilization reflects the compression of the schedule.

To verify the increase of production rate of the shop due to the CMS optimization of job allocation, production output (total number of pieces produced on all machines) is divided by the total shop time-span, as in the case of utilization. Thus,

for SPT: $21000/(17000 - 200) = 1.25 pcs/min = 75 pcs/hr$

for CMS: $21000/(17000 - 2120) = 1.41 pcs/min = 84.7 pcs/hr$

This constitutes 12.9% production rate increase with CMS. However, the present example puts the CMS method at disadvantage due to the fact that machine # 2 precedes the starts of all other machines by $4000 - 2120 = 1880min$. If the shop time-span was counted from the start of these all other machines at $4000min$ then, with the production output reduced by the amount produced by machine # 2 during these $1880min$, then the production rate would be:

for modified CMS: $(21000 - 636)/(17000 - 4000) = 1.57pcs/min = 94pcs/hr$

i.e., 25% production rate increase.

The increase in production rate as well as the compression of the schedule due to CMS optimization is reflected in the reduction of makespan. As Figure 5.4 and Table 5.13 show, individual jobs experience up to 71% makespan reduction in comparison with SPT case.

This example permits assessment of the advantages of job allocation using constant job-mix stages. Shorter lead times, lower inventory (WIP) and additional capacity have been recognized by Goldratt [17] as having a major effect on the throughput. Of course, the new capacity has an effect only when it is utilized.

An important benefit of the decomposition of the problem is the reduction of computational time. The expression derived in section 5.3.2 can be applied here also. For the example studied, the non-decomposed problem contains 58 constraints, while the number of constraints for each stage sub-problem is $\{4, 7, 8, 8, 7, 9, 7, 4, 7, 7, 4\}$. Hence the savings in computational time will be

$$R_T = \frac{58^3}{4^3 + 7^3 + 8^3 + 8^3 + 7^3 + 9^3 + 7^3 + 4^3 + 7^3 + 7^3 + 4^3} \Rightarrow 53.31$$

Table 5.10: Comparison of late start times.

Jobs	*Late start times (min)*		*Due Dates (min)*
	SPT	*CMS*	
A	1680	4040	5000
B	200	2120	7000
C	4040	6440	9000
D	5120	5120	9000
E	1850	7990	12000
F	6440	9000	15000
G	8300	12000	16000
H	4455	6600	17000

Thus it will take 53 times less computational time to solve the problem using CMS algorithm than when all stages are aggregated together.

Table 5.11: Comparison of utilisation.

Machines	*Utilization %*		*Increase factor*
	SPT	*CMS*	*CMS/SPT*
1	60.4	86.3	1.43
2	88.1	98.6	1.14
3	71.2	80.4	1.13
4	42.9	80.4	1.87
5	74.7	78.0	1.04

Table 5.12: Comparison of free capacity.

Machines	*Free capacity (min)*		*Increase factor*
	SPT	*CMS*	*CMS/SPT*
1	1850	4000	2.16
2	200	2120	10.60
3	4010	4000	0.99
4	1680	4000	2.38
5	4455	4000	0.90
Total	12225	18120	1.48

Table 5.13: Comparison of makespan

Jobs	*Makespan (min)*		*Reduction*
	SPT	*CMS*	*%*
A	3320	960	71
B	6800	4880	28
C	4260	1860	56
D	3880	3880	0
E	10150	4010	60
F	8560	6000	30
G	7700	4000	48
H	12545	10400	17

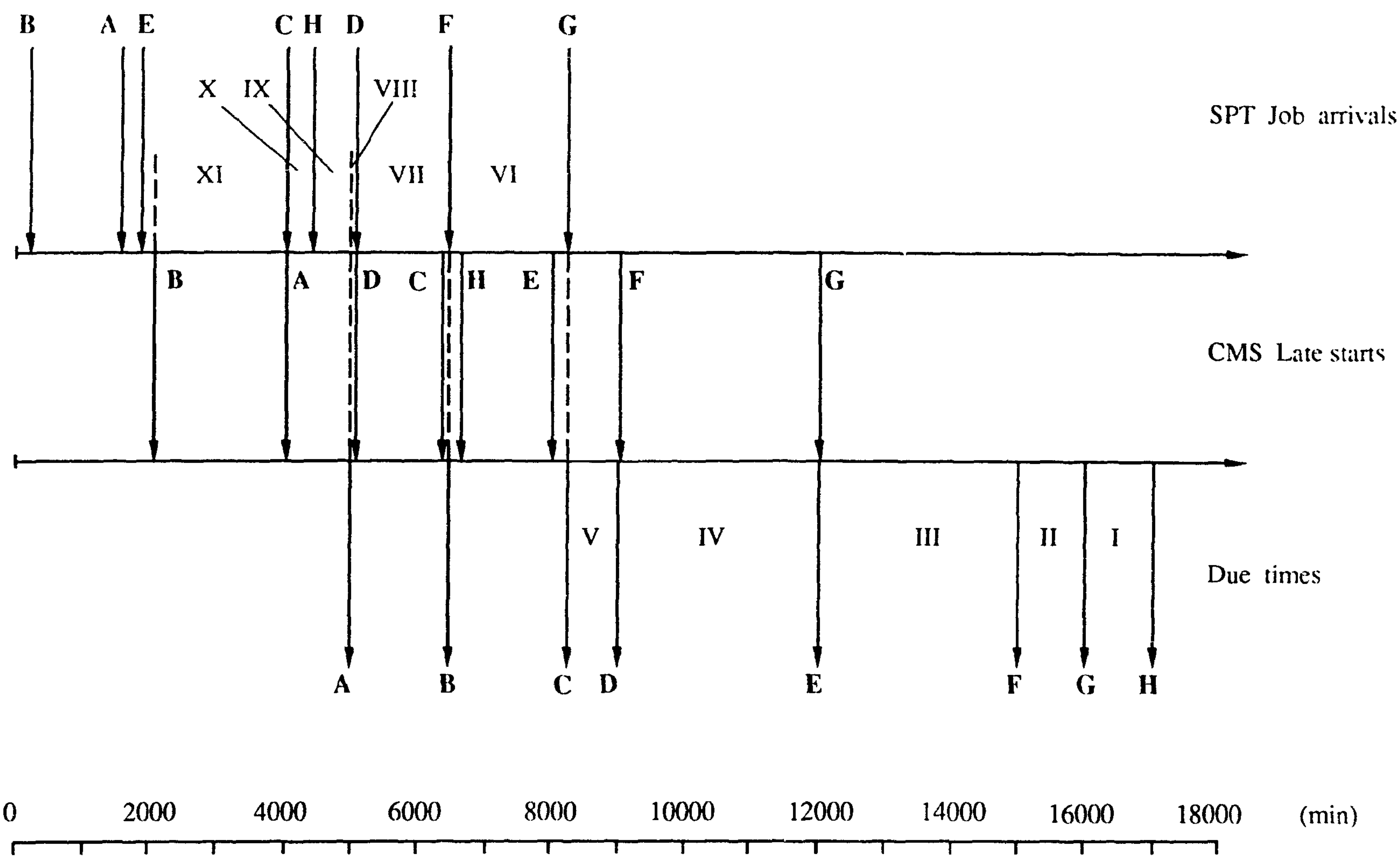

Fig 5.2 Job arrivals (SPT), late starts (CMS) due times and stages

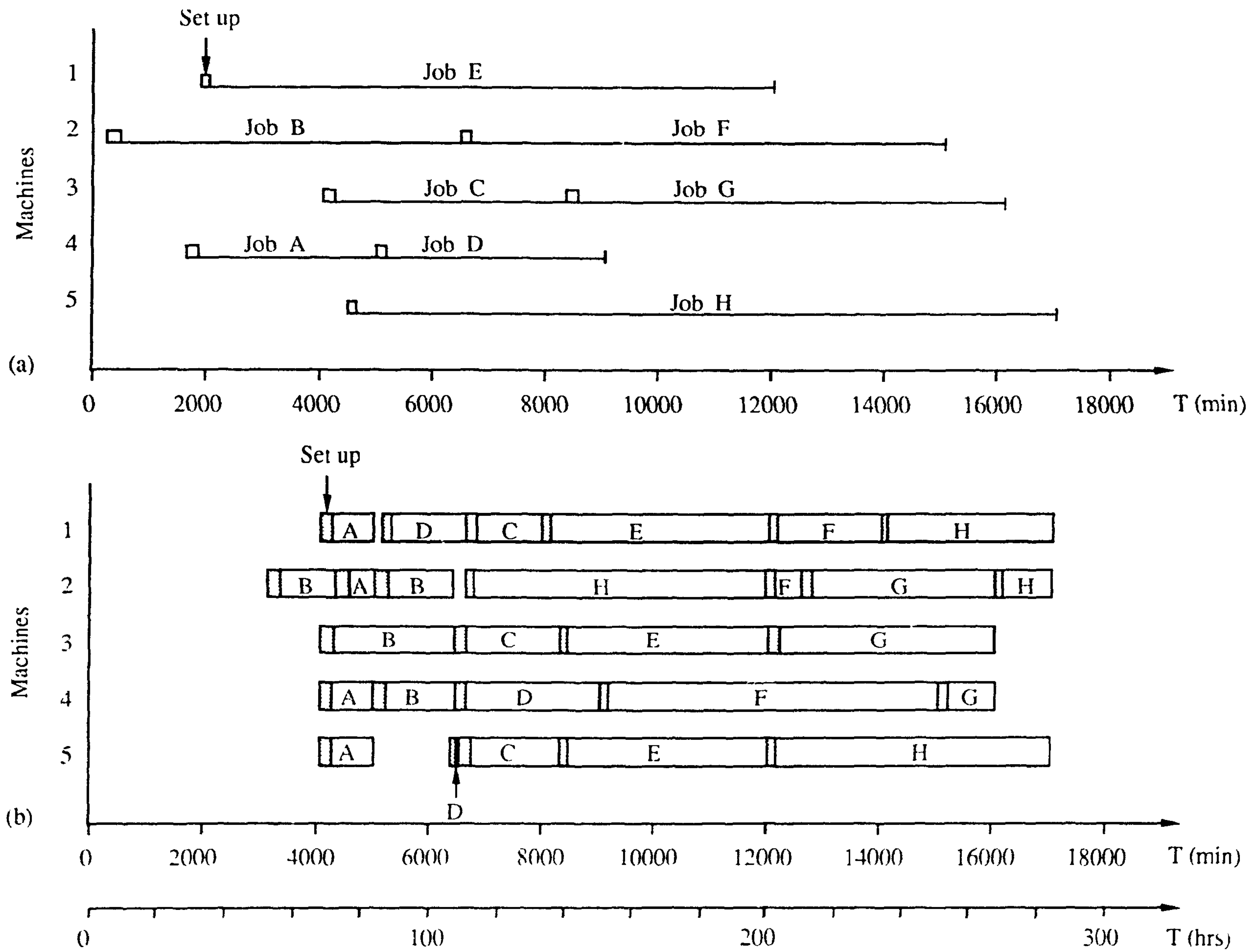

Fig 5.3 Job allocation pattern determined by: (a) SPT rule, and (b) CMS optimization

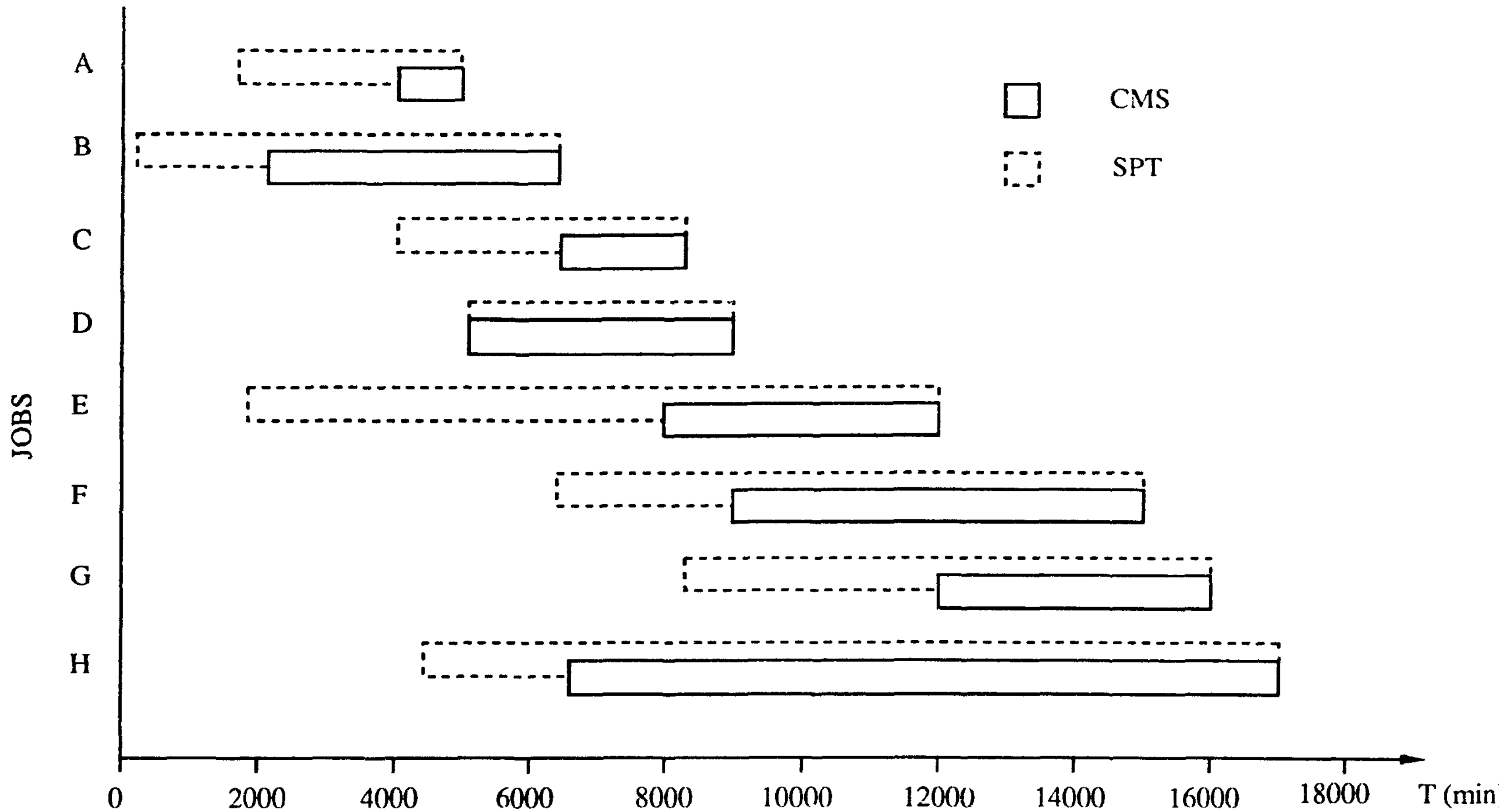

Fig 5.4 Starts and makespans of jobs determined by SPT rule and CMS algorithm

Chapter 6

Application of Algorithm to Multiple Operation Job Shops

6.1 Analysis of the Problem

In this chapter, the job allocation process involved in multiple operation job shops will be considered. This is an extension of the single operation problem. The principles, L.P. formulation and the solution procedure can be adopted for multiple operation jobs with minor modifications.

The machine tools in the factory are assumed to be grouped according to the nature of processes *e.g.*, turning, milling, grinding *etc*. All jobs are assumed to go through the different sections (operations) in a predetermined sequence. There are alternative machine tools in each section. The processing times, setup times, due dates, quantities, machine availability and capability are known a priori. Job allocation problems of this nature are typical for factories where the layout of machine tools is according to the processes. By combining multiple operations the problem becomes more realistic and hence the applicability of solutions to job shops is appropriate. The optimization is carried out within each section, thus providing a cumulative effect on the entire shop operations.

6.2 L.P. Formulation

The linear programming model provided in section 4.2 is modified to suit the multiple operation problem. In addition to the nomenclature already provided, the subscript l acts as the index for sections (operations). The linear programming model of the job allocation problem within each partition i (stage) is formulated with an objective to maximize the number of pieces X_{ijml} of each job j ($j = A, B, C, ..., N$) produced on a machine tool m ($m = 1, 2..., M$) at the operating sections l ($l = 1, 2, ..., L$). While the machines in each section are not identical, a job can be processed on more than one machine. Processing time R_{jml} and setup time S_{jml} are incurred when one unit of job j is loaded on machine m at section l. The L.P. formulation for the job allocation problem is as follows:

$$Max \qquad \{\sum_{m=1}^{M}(\sum_{j=A}^{N} r_{iml}s_{jml}X_{ijml})\}$$

subject to:

$$\sum_{m=1}^{M} r_{iml}s_{jml}X_{ijml} \le Q_j \qquad j = A, B, \ldots, N$$

$$\sum_{j=A}^{N} r_{iml}s_{jml}R_{jml}X_{ijml} \le T_{iml} \qquad m = 1, 2, \ldots, M$$

$$X_{ijml} \ge 0 \qquad j = A, B, \ldots, N$$

$$m = 1, 2, \ldots, M$$

$$r_{iml} = \begin{cases} 1 & \text{if machine } m \text{ at section } l \text{ is available at stage } i \\ 0 & \text{otherwise} \end{cases}$$

$$s_{jml} = \begin{cases} 1 & \text{if machine } m \text{ at section } l \text{ is capable of processing job } j \\ 0 & \text{otherwise} \end{cases}$$

The machine availability index r_{iml} and the machine capability index s_{jml} are responsible for eliminating all the terms that reduce to zero, both in the objective function and constraints. The first set of constraints ensures that the quantity produced at each stage is not more than the demand quantity. The second set of constraints restricts the total processing time on each machine tool not to exceed the available time on that machine. The last set is a non-negativity constraint for the quantity of jobs produced in the stage.

6.3 Solution Procedure

The solution procedure described in section 4.3 is used here. An additional step is provided to link the different sections (operations). The solution procedure encompassing both the linear program and the heuristics is shown in Fig 6.1.

6.4 Numerical Example

Let us consider a case of 6 jobs that has to undergo 3 operations (*e.g.,* turning, milling and grinding). Each section (operation) has 4 machines and the jobs can be processed in any of the alternative machine tools. The processing times and the setup times at different sections are given in Tables 1 and 2, while the demand quantities and the due dates are shown in the Table 3. The Tables indicate also the machine tools that are not capable of processing a particular job.

The optimization problem can be stated as follows. For a case of $N = 6$ jobs, $M = 4$ machines in each section, the optimization problem based on CMS algorithm is defined for every stage with an objective to maximize the number of pieces of each job processed within a stage. Such an objective leads to maximum production rate and machine utilization. Each job has to be processed in sections 1, 2 and 3, in that order. Production is in lots equivalent to the demand quantity within the due dates specified. The starting times are calculated by applying SPT rule backwards from the due dates given and this provides a reference to define the stages. In this procedure the jobs are assumed to be non-preemptive and

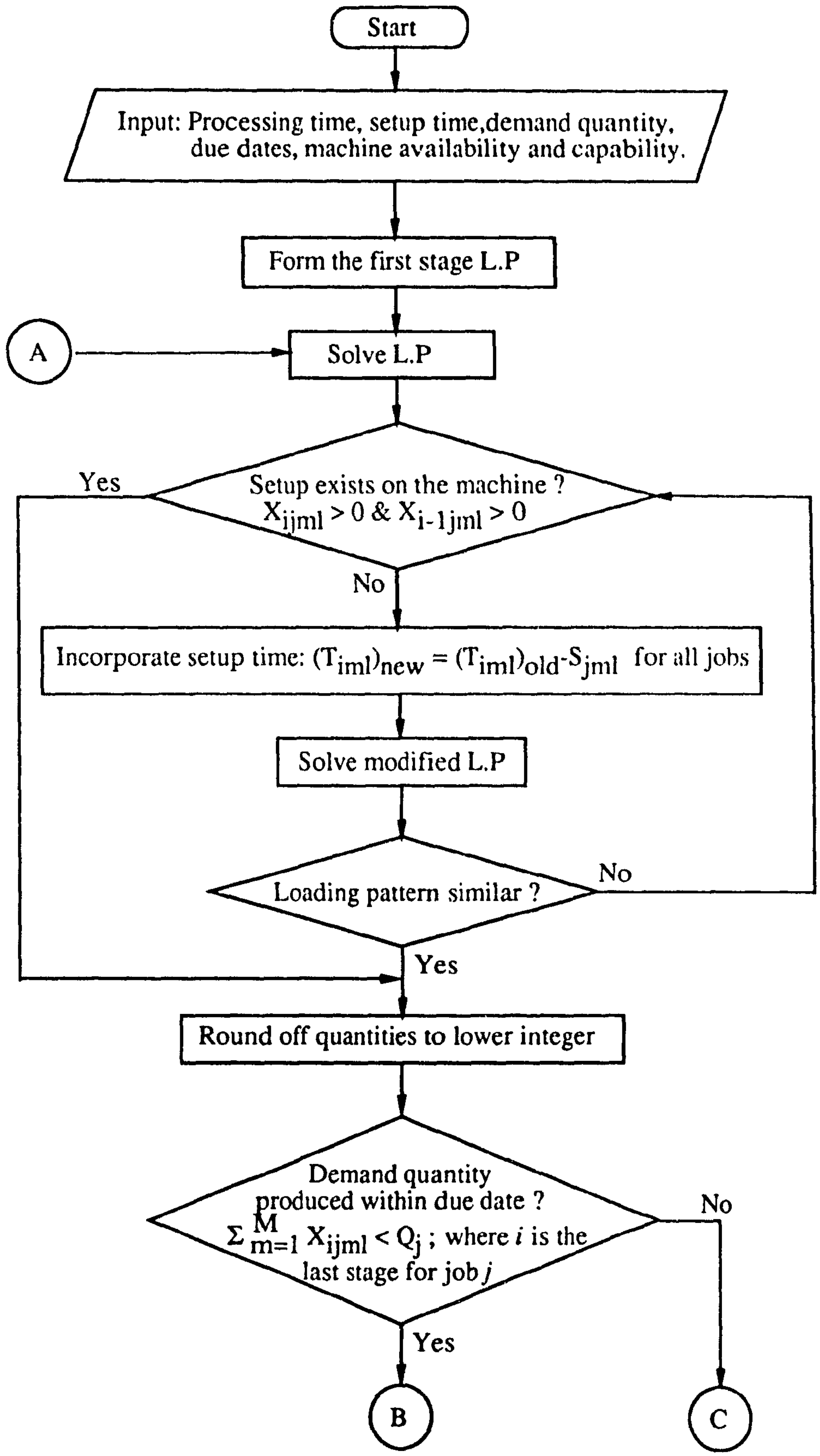

Fig 6.1 Flow chart for the solution procedure

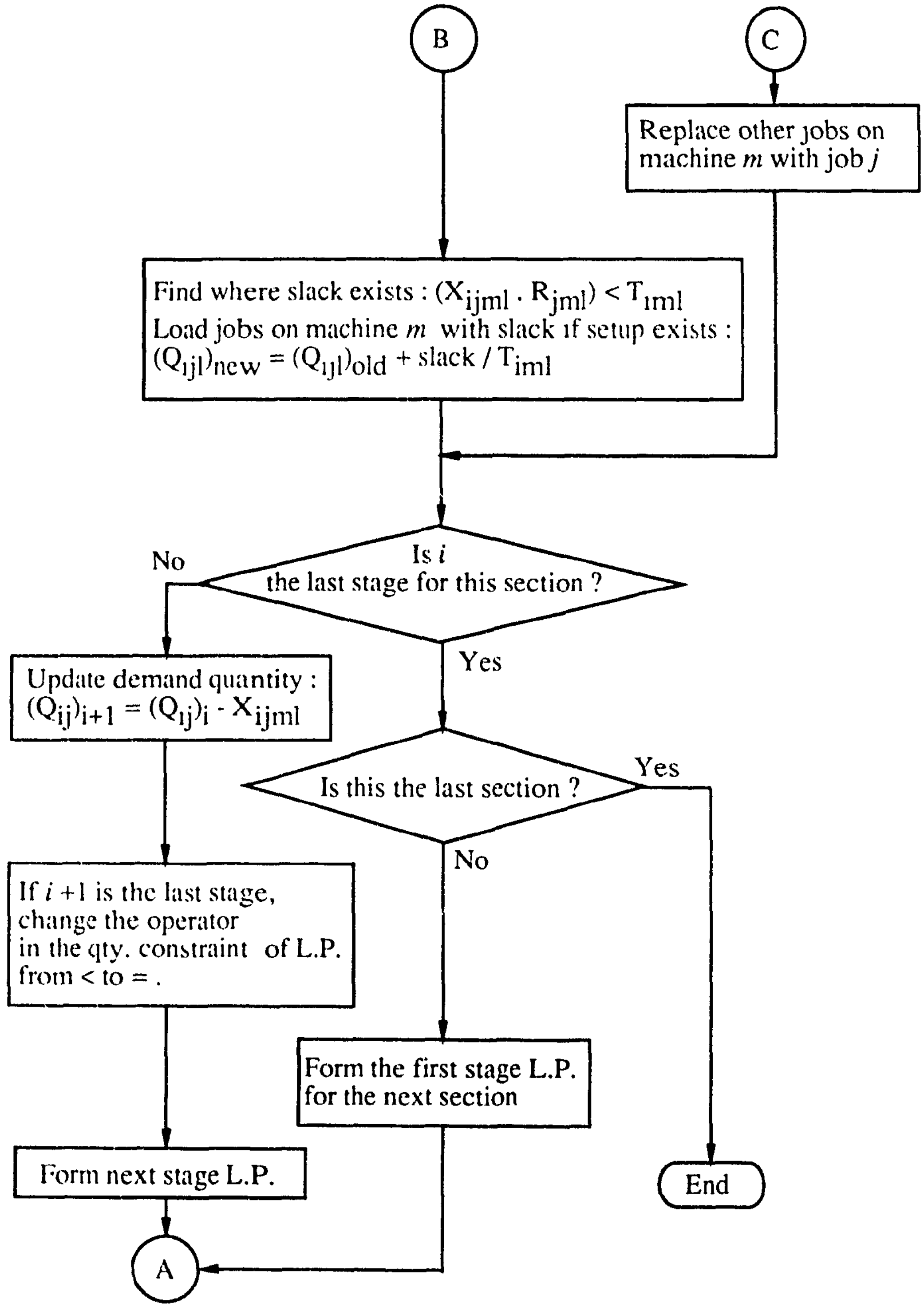

Fig 6.1 Flow chart of for the solution procedure (contd.)

when two or more jobs compete for the same machine the job having the least processing time is assigned and the rest of the jobs is allocated to the remaining machines.

6.5 Results and Discussion

The results obtained by applying CMS algorithm are presented in Tables 6.1 to 6.6 and Figures 6.2 to 6.4. As an example of the resultant job allocation, Fig 6.2 shows the distribution of jobs on machine tools in the first section. The SPT allocates each job to one machine only, causing unbalanced machine loading. By distributing jobs on alternate machine tools, the CMS algorithm achieves better utilization of machines. Table 6.4 shows utilization of machine tools for all three sections. The utilization was calculated as the ratio of time the machine is busy, to the time between the entry of the first job and departure of the last job from the section in percentage. According to Table 6.4, an average of 43% increase in utilization of machine tools for all three sections was achieved when CMS algorithm is applied (ave. 87.3%), in comparison with SPT rule (ave. 61%).

Another positive effect of the application of CMS algorithm is the reduction of completion dates of every job, in the present example. Fig. 6.3 provides the Gantt diagram for jobs being processed in all three sections. For clarity of presentation only the first and the last job are plotted, A and F respectively. Processing times for jobs allocated according to both SPT rule and CMS algorithm

Table 6.1: Processing times of jobs on machine tools

Section	Machine *m*	A	B	C	D	E	F
		Processing Times (min)					
section 1	1	1.0	0.98	1.8	3.0	3.2	3.92
	2	1.5	1.0	1.85	†	3.0	3.90
	3	0.95	1.5	2.0	3.3	3.4	4.05
	4	†	0.95	1.9	2.95	3.25	3.95
section 2	1	2.1	1.1	2.7	4.1	†	2.7
	2	2.25	1.0	2.8	4.2	3.5	2.5
	3	2.2	†	2.75	4.25	3.35	2.8
	4	2.0	1.15	2.85	4.3	3.2	2.6
section 3	1	7.4	6.5	9.2	7.6	8.75	7.35
	2	7.3	6.3	9.1	7.5	8.8	†
	3	7.2	6.2	†	7.3	8.5	7.3
	4	7.35	6.35	9.15	7.55	8.7	7.4

†-the machine is not capable of processing the job

Table 6.2: Setup times of jobs on machines

Section	Machine m	A	B	C	D	E	F
		Setup Times (min)					
section 1	1	60	75	105	80	55	75
	2	50	70	100	†	60	70
	3	65	65	90	55	50	80
	4	†	100	90	95	65	65
section 2	1	80	105	85	100	†	100
	2	85	120	90	90	120	110
	3	83	†	95	95	115	100
	4	90	110	80	95	110	105
section 3	1	60	45	90	70	65	100
	2	65	50	100	75	60	†
	3	70	65	†	80	70	105
	4	65	60	100	70	65	95

†-the machine is not capable of processing the job

Table 6.3: Demand quantities and due dates

Jobs	A	B	C	D	E	F
Quantity	600	800	1200	700	850	900
Due Date (min)	12000	15000	23000	19000	16000	16000

are shown for comparison. Since each section is solved sequentially, reduction in completion time has a cumulative effect and the due dates are significantly shortened. This amounts to additional avaialble free capacity and could be used to process other jobs. The reduction in completion dates is linked to shorter makespans which can be achieved when CMS optimization algorithm is applied. This is illustrated in Fig. 6.4 and by the data in Table 6.5. It can be seen from this Table, that a reduction in makespan between 5.7% and 73.8% was achieved for individual jobs. The reduction in makespa· of course, the result of the distribution of jobs on machines.

As expected, the maximum output objective results in increased production rate for all sections as indicated in Table 6.6. The maximum increase of 43.6% was achieved in section # 3. One should note, that although in the present example each job passes through all the sections, in general a job may be scheduled to bypass certain operations and to be processed in the remaining sections only.

The reduction in computational time which is derived as a result of decomposing the scheduling horizon into constant job-mix stages, can be calculated for every section (operation), using the expression derived in 5.3.2. The number of constraints present in the L.P., when all stages are aggregated, for sections 1, 2 and 3 are 36, 31 and 38 respectively. The number of constraints present in the stages:

for section # 1: $\{3, 5, 6, 7, 7, 8, 5, 5\}$

for section # 2: $\{5, 6, 5, 5, 5, 6, 7\}$

for section # 3: $\{5, 6, 6, 7, 7, 8, 8, 8\}$.

Hence the computational savings for each section is:

$$R_{T1} = \frac{36^3}{3^3 + 5^3 + 6^3 + 7^3 + 7^3 + 8^3 + 5^3 + 5^3} \Rightarrow 25.7 \approx 26$$

$$R_{T2} = \frac{31^3}{5^3 + 6^3 + 5^3 + 5^3 + 5^3 + 6^3 + 7^3} \Rightarrow 30.8 \approx 31$$

$$R_{T3} = \frac{38^3}{5^3 + 6^3 + 6^3 + 7^3 + 7^3 + 8^3 + 8^3 + 8^3} \Rightarrow 19.75 \approx 20$$

Table 6.4: Comparison of machine utilization

Machine		Machine Utilization %	
	m	*SPT*	*CMS*
section 1	1	46.6	97.9
	2	92.1	89.3
	3	48.9	95.3
	4	71.3	89.3
section 2	1	50.7	94.2
	2	49.4	57.2
	3	46.7	46.7
	4	62.5	97.0
section 3	1	35.6	100.0
	2	72.8	99.7
	3	77.2	85.9
	4	78.8	94.6

Table 6.5: Comparison of completion dates and makespans

Jobs		A	B	C	D	E	F
Completion Dates (min)	*SPT*	9691	10682	23000	16000	19000	19000
	CMS	6583	5570	15990	15227	15912	11131
Makespan (min)	*SPT*	10253	12780	20690	13176	16336	15710
	CMS	4836	3350	13680	12573	13249	10841
Reduction	%	52.8	73.8	33.8	5.7	18.9	30.9

Table 6.6: Comparison of production rate

Op.Section	*Production Rate (pcs/min)*		
	SPT	*CMS*	*Increase %*
1	0.838	1.139	35.9
2	0.620	0.744	20.1
3	0.267	0.383	43.6

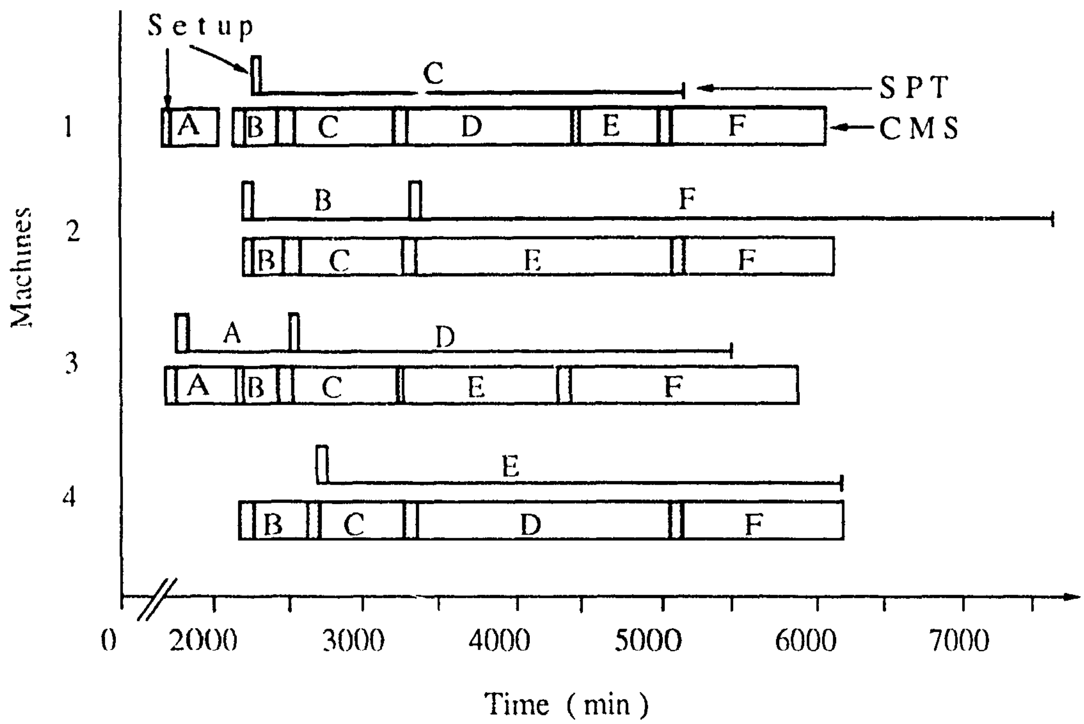

Fig 6.2 Allocation of jobs at section # 1 determined by SPT rule and CMS algorithm

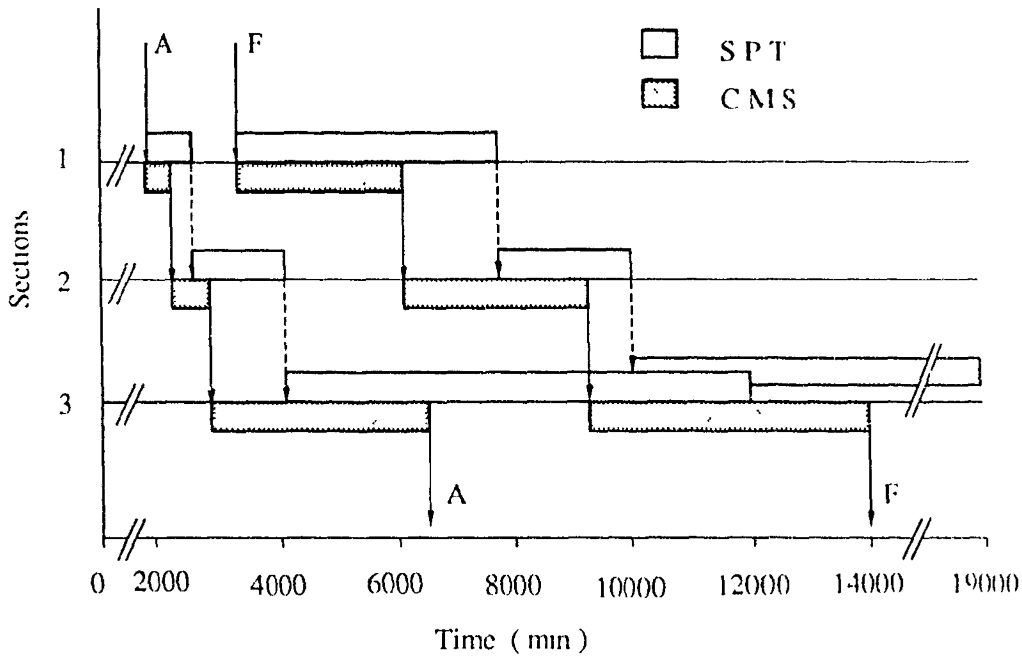

Fig 6 3 Gantt chart representing the flow of job A and job F for SPT rule and CMS algorithm

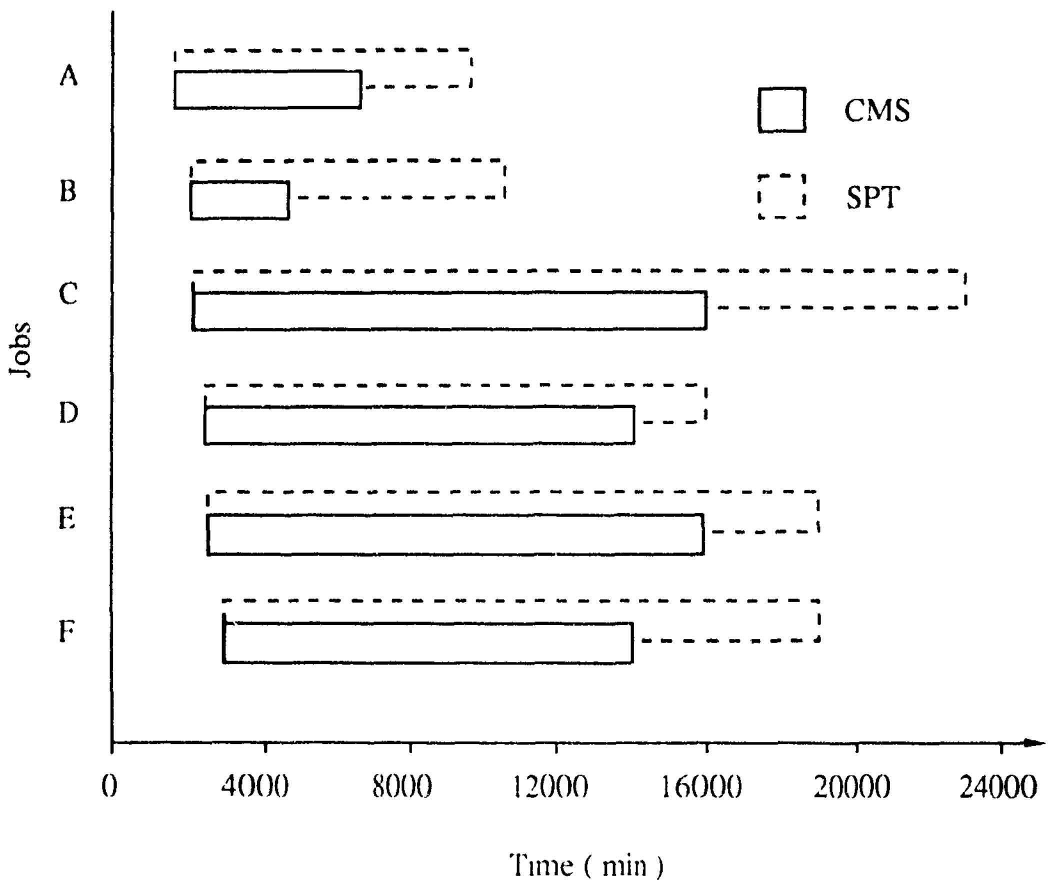

Fig 6.4 Completion dates and makespans of jobs resulting from SPT rule and CMS algorithm

Chapter 7

Integration in Intelligent Manufacturing Systems

Within the many components of manufacturing decision processes, operation scheduling can be perceived not only as an important link in the chain, but even as a hub of the information system linking the activities associated with the production processes. The information flow concerns both the decision processes and the material flow *i.e.*, physical execution of orders. However, the vast amount of raw data existing in present day management information systems is not used for optimization of production management, a particularly difficult task considering that this is a dynamically changing environment.

The usefulness of integrating the capability of intelligent systems with traditional information systems has been recognized already by Tseng and O'Connor [18], while Lu [19] proposed a framework for integration of heuristic and deterministic knowledge for engineering decisions.

The present procedure for dynamic optimal job allocation using the CMS algorithm links existing computerized information system with combined deterministic and heuristic solution. Thus, the procedure extends the functions covered by the information system to operational level decision.

The link to the existing management information system is accomplished through the manufacturing resource planning system (MRP II). It contains, as one of its five function, the master production schedule (MPS), the others being: requirements planning, capacity planning, shop floor control and cost control (production accounting) (Flapper *et al.* [20]).

The MPS module provides a timetable specifying what components, in what quantities are needed and when. However, MPS does not extend its functions to operations scheduling, *i.e.*, job sequencing and job-resource allocation. In practice, while sequencing is done using some priority rules, the decision concerning job allocation on machine tools is left to the foreman on the shop floor. And yet, these functions are of critical importance for shop efficiency which affects its actual capacity and subsequently the throughput.

The CMS algorithm can be incorporated into the MPS module, and then, linked to other functions of MRP II system. Such an integration would function in the following way: the MPS module generates the master production schedule, which triggers the release of production orders for all components which have to be processed in the shop. Based on the production orders and the data available from the manufacturing database, the CMS algorithm generates the job allocation schedule

An important factor in the integration of CMS algorithm with the MPS/MRP II system is the type of information and the compatibility of data. To generate a schedule, MPS uses information on demand quantities and delivery dates The CMS algorithm uses the same information, in addition to which it uses also the machine availability and capability data from the manufacturing database. While the algorithm is reasonably independent from the MRP II modules, the structure of the data, the access areas of the algorithm and the hierarchy of the algorithm function should be taken into account while integrating.

The integration of the CMS job allocation algorithm with the MPS module (which also contains process planning information) could be, and indeed should be, extended to other components of the manufacturing decision system, such as: scheduling and allocation of maintenance, personnel and equipment on machines, scheduling (and in some cases also allocation) of components processed on various machines to quality control stations and/or assigning quality control inspectors to machine tools. Further extension would also interface scheduling of material release and material handling (physical) to be synchronized with the job-machine allocation and sequencing.

All these links would have the feedback loops equipped with learning – self correcting adaptive capability for the use of CMS algorithm for possible rescheduling and reallocation of jobs. Such a design would make full use of the dynamic optimization capabilities of the CMS algorithm. These areas are the obvious topics for future research and development in an effort towards developing computer integrated intelligent manufacturing systems – an intelligent CIM.

Chapter 8

Conclusions and Recommendations for Further Study

8.1 Conclusions

A new method which has been developed to optimize job allocation on machine tools in a time-dependent environment, enhances the efficiency of production planning systems by generating more realistic schedules. Besides, since the job allocation function is closely related to other components of a manufacturing system, the schedules obtained by using the Constant Job-Mix Stage (CMS) algorithm aids in the efficient functioning of the related departments like maintenance, quality control *etc.* As the CMS algorithm targeted the key parameters of a machine shop (*e.g.*, production rate, makespan) for improvement, the performance of the overall factory is ameliorated.

The decomposition scheme partitions the scheduling horizon into stages of constant job-mix. The CMS algorithm based on this scheme uses linear programming within each stage to allocate jobs on machine tools, and associated heuristics to account for setup times, due date enforcement, integer solutions and slack reduction while optimizing job distribution. The procedure proceeds from stage to stage, the output of one stage being the input of the next. This forms a dynamic scheme of optimization which links the whole problem.

Three different examples were dealt with in detail to demonstrate the application of CMS algorithm. All of them concerned multiple jobs to be allocated on a number of machine tools, each with different capacities and capabilities. Such situations are quite typical of many job shops in which machine tools are laid out according to their processes. The following conclusions can be drawn from these applications:

1. The results of the CMS algorithm, when compared with those of the aggregated approach and SPT rule indicate that better utilization of machine tools can be achieved.

2. The compressed schedule along with high utilization of machine tools brings about a significant reduction in the makespan of individual jobs. As a result the jobs spend less time on the shop floor leading to greatly reduced work-

in-process (WIP) inventory.

3. Improved production rate (throughput) is an important benefit derived from the application of the CMS algorithm. In the case of single operation jobs the increase in production rate is measured for all the jobs and machines together. For the multiple operation case, improvement in production rate is noted for each section thus contributing to the productivity of the overall shop.

4. The CMS algorithm provides a compressed schedule; while scheduling is carried out from the starting times, early shipment dates for jobs (or orders) are obtained. When scheduled from the due dates the CMS algorithm provides late start dates. In both cases the spare time which is made available because of the compressed schedule, can be used to process additional jobs.

5. It is possible to apply the CMS algorithm to both single and multiple operation job shops. In the latter case the machine tools are grouped in sections according to their operations.

6. In comparison with the aggregated approach, the CMS algorithm provided a significant savings in computational time required to solve the job allocation problems. Using the expression developed to calculate such savings in time, it can also be concluded that, the savings in computational time increases when the stages contain fewer jobs.

7. Job allocation problems solved using the CMS algorithm, open the possibility of combining deterministic and heuristic procedure with management information systems, specifically the master production schedule of MRP II, to arrive at a solution for shop floor control.

8. As the CMS algorithm aids the job allocation decision process in a production planning function, it provides an efficient control when integrated with the other components of manufacturing system like maintenance, quality control and material release. Such an integrated system enhances the automation of various decision making processes.

8.2 Recommendations for Further Study

One of the important feature of job allocation using the CMS algorithm is the distribution of setup times. While in the common practice of assigning a job to machine with SPT, only one setup is needed for the full processing time of a job, this is not the case with the CMS method. In the new method setups are assigned to various machines according to the optimized solution, where setup times are taken into account. Since the optimization is carried out within each stage without considering the allocation in the previous stage, this may result in multiple setups for a single product on a particular machine at different periods. If such a situation is encountered, allocation of jobs on each machine can be shuffled so that fewer setups are needed. However, due to this shuffling the arrival and due times of jobs can change leading to the redefinition of stages. In such circumstances, a comparison between CMS algorithm allocation and the shuffled schedule can be carried out with a suitable performance criterion.

In the CMS algorithm jobs are assumed to be preemptive. Because of the multiple setups problem mentioned earlier, it may be worthwhile to conduct a comparison between preemptive and non-preemptive scheduling with makespan criteria. The non-preemptive schedule is obtained by assigning jobs on only machines which are free to load while allocated to subsequent stages.

In the job allocation schedule, the setup times are accounted for, independently of the processing times. That is, the setup times are assigned based on the initial solution and often the job allocation before and after the setup time assignment are the same. This is because, once the setup time is deducted from the total available time, optimization is carried out for the remaining time. In order to consider collectively both setup and processing times in the schedule, break even analysis can be used. The equation for each machine will consist of a fixed setup time and a variable processing time which is a direct function of the quantity processed. Choice of machine to process a particular job is arrived at by comparing these equations and obtaining the break even quantity.

Another interesting topic for future study is the transfer quantity of the jobs between the different operations in a job shop. This means one can produce a process batch (which equals the demand quantity, as is in all our cases) and move to the next operation, or in smaller lots called transfer batches. Transfer batch is used by JIT concept, where the objective is to achieve a one-unit processing batch. Such transfer batch movement from one operating section to another affects the time-partitioning and the duration of partitions because the stages are defined by the job arrivals and departures. This concept of changing a transfer batch can be applicable to jobs of high demand quantity and processing times.

Since the linear programming formulation of the job allocation problem is similar to the transportation model, the possibility of devising a solution procedure using the triangularity structure can be studied. Some of the steps in the CMS algorithm are carried out manually. The scheduling algorithm can be fully computerized such that it can either form a part of an expert system on scheduling, or be integrated with other functions of the Production System. While the former will be a dedicated scheduling system to advise on the allocation function in different circumstances, the latter will be a part of planning and control function enhancing the automation of the production management system

Bibliography

[1] GRAVES, S. C., 1981, A review of Production Scheduling.*Operations Research* **29** (4), 646-675.

[2] HORN, W. A., 1973, Minimizing Average Flow Times with Parallel Machines. *Operations research*, **21**(3), 846-847.

[3] POWELL, N., 1970, *Production Planning, Scheduling and Inventory Control* (London: The Macmillan Company).

[4] LAWLER, E. L., and MARTEL, C. U., 1989, Preemptive scheduling of two uniform machines to minimize the number of late jobs. *Operations Research*, **37** (2), 311-318.

[5] McNAUGHTON, R., 1959, Scheduling with Deadlines and Loss Functions. *Management Science*, **6**, 1-12.

[6] DANTZIG, G. B., 1959, On the Status of Multi- Stage Linear Programming Problems. *Management Science*,**6** (1), 53-71.

[7] LEON S. LASDON., 1970, *Optimization Theory of Large Systems* (New York: Macmillan).

[8] WAGNER, H. M., 1975, *Principles of Management Science* (New Jersey. Prentice-Hall, Second Edition).

[9] NASR. N., and ELSAYED, E. A., 1990, Job Shop Scheduling with Alternative Machines. *International Journal of Production Research*,**28** (9), 1595-1609.

[10] JAIN. S. K., SCOTT. K. L., and VASOLD. E. G., 1978, Order Book Balancing using Combination of Linear Programming and Heuristic Techniques *Interfaces*, 55 67.

[11] DORSEY. R. C., HODGSON. T. J., and RATLIFF. H. D., 1974, A Production Scheduling Problem with Batch Processing. *Operations Research* **22** (6), 1271-1279.

[12] IWATA, K., MUROTSU, Y., OBA, F., and OKAMURA, K., 1980, Solution of Large-Scale Scheduling Problems for Job Shop Type Machining Systems with Alternative Machine Tools. *Annals of CIRP*, **29** (1), 335-338.

[13] DILTS, D. M., 1991, The Sensitivity of Set-up Cost Estimation on the problem of Joint Lot Sizing and Scheduling. *International Journal Production Research*, **29** (1), 77-93.

[14] ALAEE, M., 1978, Efficient Loading of Machine Tools in Intermittent Manufacturing, *M.Eng. Project Report*, McGill University, Canada.

[15] GAREY, M. R., JOHNSON, D. S., and SETHI, R., 1976, The complexity of flowshop and jobshop scheduling. *Mathematical Operations Research*, **1**, 117-129.

[16] HILLIER, F. S., and LIEBERMAN, G. J., 1990, *Introduction to Operations research* (McGraw-Hill, 5^{th} ed.)

[17] GOLDRATT, E. M., 1990, *The Theory of Constraints*. (North River Press Inc., Croton-on-Hudson, New York).

[18] TSENG, M., and O'CONNOR, D., December 1986, Augmenting CIM With Intelligent Systems, *Knowledge-Based Expert Systems for Manufacturing*, ASME, PED-Vol. **24**, 3-10.

[19] LU, S. C-Y., Decemeber 1986, Knowledge-Based Expert Systems: A New Horizon of Manufacturing Automation, *Knowledge-Based Expert Systems for Manufacturing*, ASME, PED-Vol. **24**, 11-23.

[20] FLAPPER, S. D. P., MILTENBURG, G. J., and WIJNGAARD, J., 1991, Embedding JIT into MRP. *International journal of production research*, **29**(2), 329-341.

[21] BAHL, H. C., TAJ, SHARAM., and CORCORAN, W., 1991, A linear-programming model formulation for optimal product-mix decisions in

material-requirements planning environments. *International Journal of Production Research*. **29**(5), 1025-1031.

[22] BAKER, K. R., 1971, *Introduction to Sequencing and Scheduling* (New York John Wiley).

[23] BUFFA, E. S., and SARIN, R. K., 1987, *Modern Production Operations Management*, (New York: John Wiley & Sons, 8^{th} ed)

[24] BAUDIN, M., 1990. *Manufacturing Systems Analysis with Application to Production Scheduling*, (New Jersey: Yourdon Press)

[25] BURN, D., 1990. Computers come to Job-Shop Scheduling. *Canadian Machinery and Metal Working*. 20-21.

[26] CONWAY, R. W., MAXWELL, W. L., and MILLER, L. W., 1967, *Theory of Scheduling* (Reading, Mass.: Addison-Wesley) pp 74 79.

[27] FARN, C. K., and MUHLEMANN, A. P., 1979. The Dynamic Aspects of a Production Scheduling Problem. *International Journal of Production Research*, **17** (1), 15-21.

[28] GROOVER, M. P., 1987, *Automation, Production Systems, and Computer Integrated Manufacturing* (New Jersey: Prentice-Hall)

[29] GROOVER, M. P., and ZIMMERS, E. W., 1984, *CAD CAM*, (New Jersey: Prentice Hall).

[30] LYNWOOD A. JOHNSON., DOUGLAS C. MONTGOMERY., 1974, *Operations Research in Production Planning, Scheduling and Inventory Control* (New York. John Wiley & Sons) pp 315-349.

[31] MACBETH, D. K., 1989, *Advanced manufacturing: startegy and management*, (IFS publ.).

[32] MILTENBURG, J., and WIJNGAARD, J., 1991, Designing and phasing in just-in-time production systems. *International journal of production research* **29**(1), 115-131.

[33] PHILLIPS, D. T., RAVINDRAN, A., and SOLBERG, J. J., 1976, *Operations Research* (New York: John Wiley & Sons).

[34] PRABHU, P., and WANG, H.P., 1991, Algorithms for Computer Aided Generative Process Planning. *International Journal of Advanced Manufacturing Technology*, **6** (1), 3-15.

[35] RONEN, B., and STARR, M. K., 1990, Synchronised Manufacturing as in OPT: From practice to theory. *Computers Industrial Engineering*, **18** (4), 585-600.

[36] SCHROEDER, R. G., 1981, *Operations Management* (New York: McGraw-Hill Book Company).

[37] SIMON FRENCH., 1982, *Sequencing and Scheduling: An Introduction to the Mathematics of the Job-Shop* (New York: John Wiley & Sons) pp 15-17.

[38] WAGNER, H., and WHITIN, T. M., 1958, Dynamic version of the economic lot size model. *Management Science*, 5(1), 89–96.

[39] *Mathematical Programming System Extended* (MPSX) *Programmers Reference Manual* IBM Manual, October 1988.

Appendix A

Sample of Job Control and Data Cards for MPSX

This section shows an example of job control data cards used for the purpose of solving a linear programming problem. The model under consideration is example 5.4 (stage 6). The linear programming formulation of this problem is as follows:

$$Max \quad \{X_{6H1} + X_{6H2} + X_{6H5} + X_{6E1} + X_{6E3} + X_{6E5} + X_{6D1} +$$
$$+X_{6D3} + X_{6D4} + X_{6D5} + X_{6C1} + X_{6C3} + X_{6C4} + X_{6C5}\}$$

subject to:

$$X_{6H1} + X_{6H2} + X_{6H5} \leq 531$$
$$X_{6E1} + X_{6E3} + X_{6E5} \leq 155$$
$$X_{6D1} + X_{6D3} + X_{6D4} + X_{6D5} \leq 860$$
$$X_{6C1} + X_{6C3} + X_{6C4} + X_{6C5} \leq 1500$$
$$3.5X_{6H1} + 2X_{6E1} + 4X_{6D1} + 3.5X_{6C1} \leq 1520$$
$$3.2X_{6H2} \leq 1860$$
$$2.1X_{6E3} + 4.3X_{6D3} + 2.7X_{6C3} \leq 1650$$
$$3.7X_{6D4} + 3.9X_{6C4} \leq 1860$$
$$3.1X_{6H5} + 2.25X_{6E5} + 4.2X_{6D5} + 3X_{6C5} \leq 1660$$

The McGill's MUSIC (Multi User System for Interactive Computing) job control cards followed by MPSX (Mathematical Programming and System Extended) data cards are provided in the following two pages.

```
/INFO MVS R(MUSIC) CL(40)
// EXEC MPSX
//MPSGO.SYSIN DD *
        PROGRAM
        INITIALZ
        MOVE(XPBNAME,'LINEAR')
        MOVE (XDATA,'JIT6')
        CONVERT('SUMMARY')
        BCDOUT
        SETUP('MAX')
        MOVE(XOBJ,'QUANTITY')
        MOVE(XRHS,'BVECTOR')
        PICTURE
        PRIMAL
        SOLUTION
        PEND
//MPSGOX.SYSIN DD *
NAME        JIT6
ROWS
 N  QUANTITY
 L  ROW1
 L  ROW2
 L  ROW3
 L  ROW4
 L  ROW5
 L  ROW6
 L  ROW7
 L  ROW8
 L  ROW9
COLUMNS
    X6H1      QUANTITY   1.          ROW1        1.
    X6H1      ROW5       3.5
    X6H2      QUANTITY   1.          ROW1        1.
    X6H2      ROW6       3.2
    X6H5      QUANTITY   1.          ROW1        1.
    X6H5      ROW9       3.1
    X6E1      QUANTITY   1.          ROW2        1.
    X6E1      ROW5       2.
    X6E3      QUANTITY   1.          ROW2        1.
    X6E3      ROW7       2.4
    X6E5      QUANTITY   1.          ROW2        1.
    X6E5      ROW9       2.25
    X6D1      QUANTITY   1.          ROW3        1.
    X6D1      ROW5       4.
```

```
    X6D3     QUANTITY   1.        ROW3     1.
    X6D3     ROW7       4.3
    X6D4     QUANTITY   1.        ROW3     1.
    X6D4     ROW8       3.7
    X6D5     QUANTITY   1.        ROW3     1.
    X6D5     ROW9       4.2
    X6C1     QUANTITY   1.        ROW4     1.
    X6C1     ROW5       3.5
    X6C3     QUANTITY   1.        ROW4     1.
    X6C3     ROW7       2.7
    X6C4     QUANTITY   1.        ROW4     1.
    X6C4     ROW8       3.9
    XKC5     QUANTITY   1.        ROW4     1.
    XKC5     ROW9       3.0
RHS
    BVECTOR  ROW1      531.       ROW2     155.
    BVECTOR  ROW3      860.       ROW4     1500.
    BVECTOR  ROW5      1520.      ROW6     1860.
    BVECTOR  ROW7      1650.      ROW8     1860.
    BVECTOR  ROW9      1660.
ENDATA
```

Appendix B

Sample of Computational Results

The summary of computational result for the example provided in appendix A is given in the following two pages. Section 1 of the results deal with the sensitivity analysis of the variables. The main result, namely, the quantity of components loaded on different machines is obtained from section 2. The variables which enter the basic solution (denoted by BS in section 2 of the results) are the final solution for the example under consideration.

```
1MPSX/370 R 2.1.0      MPSCL EXECUTION
            PAGE    4  91/266
0NAME          JIT6
 ROWS
  N  QUANTITY
  L  ROW1
  L  ROW2
  L  ROW3
  L  ROW4
  L  ROW5
  L  ROW6
  L  ROW7
  L  ROW8
  L  ROW9
 COLUMS     X6H1      QUANTITY       1.00000   ROW1          1.00000
     X6H1     ROW5          3.50000
     X6H2     QUANTITY       1.00000   ROW1          1.00000
     X6H2     ROW6          3.20000
     X6H5     QUANTITY       1.00000   ROW1          1.00000
     X6H5     ROW9          3.10000
     X6E1     QUANTITY       1.00000   ROW2          1.00000
     X6E1     ROW5          2.00000
     X6E3     QUANTITY       1.00000   ROW2          1.00000
     X6E3     ROW7          2.40000
     X6E5     QUANTITY       1.00000   ROW2          1.00000
     X6E5     ROW9          2.25000
     X6D1     QUANTITY       1.00000   ROW3          1.00000
     X6D1     ROW5          4.00000
     X6D3     QUANTITY       1.00000   ROW3          1.00000
     X6D3     ROW7          4.30000
     X6D4     QUANTITY       1.00000   ROW3          1.00000
     X6D4     ROW8          3.70000
     X6D5     QUANTITY       1.00000   ROW3          1.00000
     X6D5     ROW9          4.20000
     X6C1     QUANTITY       1.00000   ROW4          1.00000
     X6C1     ROW5          3.50000
     X6C3     QUANTITY       1.00000   ROW4          1.00000
     X6C3     ROW7          2.70000
     X6C4     QUANTITY       1.00000   ROW4          1.00000
     X6C4     ROW8          3.90000
     X6C5     QUANTITY       1.00000   ROW4          1.00000
     X6C5     ROW9          3.00000
```

RHS

BVECTOR	ROW1	531.00000	ROW2	155.00000
BVECTOR	ROW3	860.00000	ROW4	1500.00000
BVECTOR	ROW5	1860.00000	ROW6	1860.00000
BVECTOR	ROW7	1860.00000	ROW8	1860.00000
BVECTOR	ROW9	1860.00000		

ENDATA

0SECTION 1 - ROWS

- NUMBER	...ROW..	AT	...ACTIVITY...	SLACK ACTIVITY	..LOWER LIMIT.	..UPPER LIMIT.	.DUAL ACTIVITY
0 1	QUANTITY	BS	2908.98048	2908.98048-	NONE	NONE	1.00000
2	ROW1	UL	531.00000	.	NONE	531.00000	1.00000-
3	ROW2	UL	155.00000	.	NONE	155.00000	.50000-
4	ROW3	BS	722.98048	137.01952	NONE	860.00000	.
5	ROW4	UL	1500.00000	.	NONE	1500.00000	.12500-
6	ROW5	UL	1860.00000	.	NONE	1860.00000	.25000-
7	ROW6	BS	1699.20000	160.80000	NONE	1860.00000	.
8	ROW7	UL	1860.00000	.	NONE	1860.00000	.32407-
9	ROW8	UL	1860.00000	.	NONE	1860.00000	.27027-
10	ROW9	UL	1860.00000	.	NONE	1860.00000	.29167-

1MPSX/370 R 2.1.0 MPSCL EXECUTION PAGE
10 91/266

0SECTION 2 - COLUMNS

- NUMBER	.COLUMNS	AT	...ACTIVITY...	..INPUT COST..	..LOWER LIMIT.	..UPPER LIMIT.	.REDUCED COST.
0 11	X6H1	LL	.	1.00000	.	NONE	.87500-
12	X6H2	BS	531.00000	1.00000	.	NONE	.
13	X6H5	LL	.	1.00000	.	NONE	.90417-
14	X6E1	BS	155.00000	1.00000	.	NONE	.
15	X6E3	LL	.	1.00000	.	NONE	.27778-
16	X6E5	LL	.	1.00000	.	NONE	.15625-
17	X6D1	BS	220.27778	1.00000	.	NONE	.
18	X6D3	LL	.	1.00000	.	NONE	.39352-
19	X6D4	BS	502.70270	1.00000	.	NONE	.
20	X6D5	LL	.	1.00000	.	NONE	.22500-
21	X6C1	BS	191.11111	1.00000	.	NONE	.
22	X6C3	BS	688.88889	1.00000	.	NONE	.
23	X6C4	LL	.	1.00000	.	NONE	.17905-
24	X6C5	BS	620.00000	1.00000	.	NONE	.

1MPSX/370 R 2.1.0 MPSCL EXECUTION PAGE
11 91/266

0EXIT - TIME = 0.00