HIERARCHICAL PLANNING OF OPERATIONAL DECISIONS IN FLEXIBLE MANUFACTURING SYSTEMS

by

Hande Savaş

B.S. in I.E., İstanbul Technical University, 1993

Submitted to the Institute for Graduate Studies in Science and Engineering in partial fulfillment of the requirements for the degree of Master of Science in Industrial Engineering

Boğaziçi University

1995

To my parents,
Zeki SAVAŞ
Müveddet SAVAŞ

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my thesis supervisor, Assoc. Prof. Gülay BARBAROSOĞLU for her invaluable guidance during this research. I am also thankful to her for her moral support, encouragement and motivation throughout all phases of my graduate study, for creating an opportunity for me to direct my carrier and for her advice which I will always consider in my life.

I would like to thank Assist. Prof. Ümit BİLGE and Assoc. Prof. Bülent DURMUŞOĞLU for their valuable comments and suggestions as members of my thesis committee.

My special thanks go to Assist. Prof. Tülin YAZGAÇ and Murat GÖKÇAY for their valuable helps and comments during the development of the computer software.

I wish to thank my brother Erol SAVAŞ and my cousins Özgül and Yasemin KURTULUŞ who were always with me throughout this study.

I would like to express my thanks to all of my friends, especially to Ayşe SEYHAN, Ayşe AKSOĞAN and Emre KABAŞ for their kindness and moral support during all stages of my graduate study.

Finally, I wish to express my deepest gratitude to my parents for their moral support, encouragement and patience not only during this research, but throughout all phases of my life.

ABSTRACT

This study proposes a hierarchical planning model for the operational problems in Flexible Manufacturing Systems (FMSs). The functional problems included in the model are part type selection (batching), loading and scheduling. The top level of the hierarchy deals with the batching decisions, the second level solves the machine loading and tool allocation problem, and the lowest level determines machine and material handling equipment scheduling. The batching and loading levels are formulated as linear mathematical models where the loading model considers various objective functions. A Lagrangian relaxation based solution approach is also applied to the loading level, and since it is not always possible to obtain feasible solutions by this method, a Lagrangian heuristic is developed to make the solutions feasible. On the other hand, the effect of aggregation/disaggregation of information between the batching and the loading levels upon the feasibility of top-down commands is of particular importance in the study. Finally in the scheduling level, a dispatching algorithm is developed, and various dispatching rules are tested under various performance criteria. In order to perform this analysis, some model parameters are varied, and certain conclusions are drawn about these dispatching rules.

ÖZET

Bu çalışmada, esnek üretim sistemlerinde söz konusu olan operasyonel kararları hiyerarşik olarak planlayan bir model geliştirilmiştir. Modele dahil edilen bu kararlar; parça seçimi, makina yükleme ve çizelgeleme problemlerini kapsamaktadır. Ortaya koyulan hiyerarşik yapının en üst seviyesi parça seçimi, ikinci seviyesi makina yükleme, en alt seviyesi ise makina ile malzeme taşıma sistemlerinin çizelgelenmesi konularını incelemektedir. Parça seçimi ve makina yükleme problemlerinin her biri için bir doğrusal matematik model geliştirilmiş, makina yükleme probleminde birden fazla amaç fonksiyonu göz önüne alınarak, model her bir amaç fonksiyonu için çözülmüştür. Ayrıca makina yükleme problemi için Lagrange çarpanlarının kullanıldığı alternatif bir çözüm yöntemi önerilmiş, fizibilite koşulunun sağlanmadığı durumlar için ise bir sezgisel geliştirilmiştir. Hiyerarşinin en üst ve ikinci seviyeleri arasındaki bilgi alışverişi de çalışmada özel olarak ele alınmış bir konudur. Hiyerarşik modelin en alt seviyesinde makina ve malzeme taşıma sistemlerinin çizelgelemesini yapan bir algoritma geliştirilmiş ve çeşitli performans kriterlerine göre çizelgeleme kuralları değerlendirilmiştir. Bu işlemi yürütebilmek amacı ile model parametrelerinde değişiklikler yapılmış ve son olarak da bu kurallarla ilgili birtakım sonuçlara varılmıştır.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

The main aim of this study is to reveal the high interaction between the functional decisions which are encountered in the planning of Flexible Manufacturing Systems (FMSs) and to analyze them in a hierarchical model in the form of information exchange.

In an FMS, design and operational problems are of particular importance [1]. Design problems include selection of an FMS production system, a material-handling system, fixtures and pallets, an appropriate computer system, and layout and integration of all the above systems. Operational problems which are the main focus of this study consist of batching, loading and scheduling subproblems. By means of the batching decisions, the subset of parts which can be manufactured together during a shift with a common tooling can be determined. Loading decisions assign all the operations of the parts included in a batch to the machine centers. Scheduling problem is solved in order to define the sequence of parts in the automated material handling system and in the machine centers. Since these problems are sequential in nature, the main characteristics of them are also considered in a hierarchy in this study.

The subproblems under consideration in this study tend to encompass all general features inherent in an FMS and be applicable in a real life application without loss of generality. In fact, a large number of test problems are generated and implemented in a general purpose FMS set-up in order to measure the performance of the information exchange process and the degree of interrelation between the levels.

The reason why FMSs among the production system types are considered in this study is that these systems and the functional decisions that must be made in their planning process have gained much importance since they are capable of integrating quality, speed and flexibility which are the major requirements of the success of industrial firms in the intense competition in the market.

In the 1940s, specific types of operations could be performed by the stand-alone machine tools such as lathes and milling machines. In the 1950s, numerically controlled machines which could carry out a variety of operations were introduced to industry, and the usage of these machines was disseminated in the 1960s. Later, by the development of computerized machine controllers in the 1970s, computer numerically controlled (CNC) machines were started being used. In the late 1970s, hierarchically controlled systems were

developed, and direct numerical control (DNC) cells with computers controlling groups of machine tools were introduced to manufacturing systems. Since 1980s, FMS and computer-integrated manufacturing (CIM) systems concepts have gained worldwide attention, and hundreds of these systems have been implemented around the world with Japan leading in terms of the number of applications and associated management and organizational success [1].

An FMS incorporates a set of flexible machine tools, automated material handling and storage systems and resources to be shared by part types such as tools, pallets, carriers and fixtures into a single production system under the control of a computer system. The material-handling system (usually automatic guided vehicles) provides the linkages between the work-stations by interfacing with load/unload stations, fixture and pallet staging areas and internal storage facilities. Most of the real time functions including down-loading the part programs to control the machining operations and controlling part movements and cutting-tool interchanges are coordinated by a central computer system which is interconnected by a hierarchical control structure [2].

FMSs can be classified into groups depending on the number of numerically controlled (NC) machines and their arrangement [1]. These groups are as follows:

(a) Flexible manufacturing module (FMM): It consists of an NC machine with a part buffer, a tool changer and a pallet changer. Changers can also be replaced by a robot. Figure 1.1 shows an FMM.

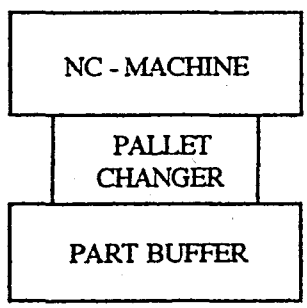

FIGURE 1.1. Flexible manufacturing module (FMM)

(b) Flexible manufacturing cell (FMC): It consists of several FMMs. It is possible to design these cells according to product or process type of manufacture . An FMC is illustrated in Figure 1.2.

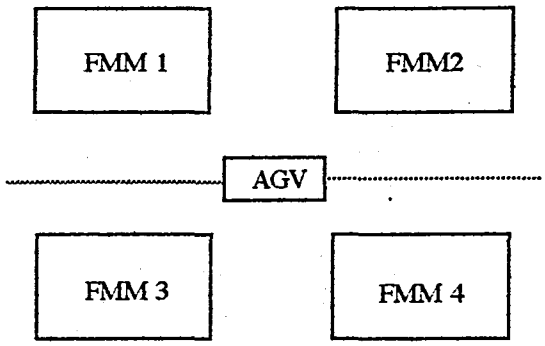

FIGURE 1.2. Flexible manufacturing cell (FMC)

(c) Flexible manufacturing group (FMG): A collection of FMMs and FMCs in the same manufacturing area connected by a material-handling system forms an FMG which is depicted in Figure 1.3. The manufacturing area may be fabrication, machining or assembly.

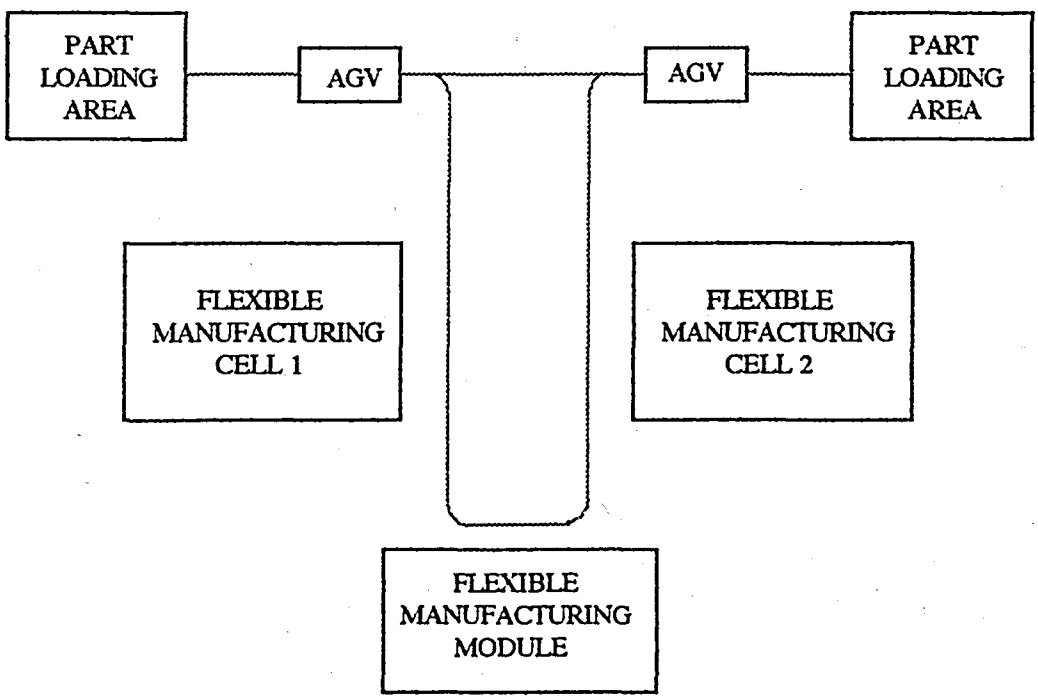

FIGURE 1.3. Flexible manufacturing group (FMG)

(d) Flexible production system (FPS): An FPS in Figure 1.4 is formed when several FMGs representing different manufacturing areas such as fabrication, machining and assembly come together.

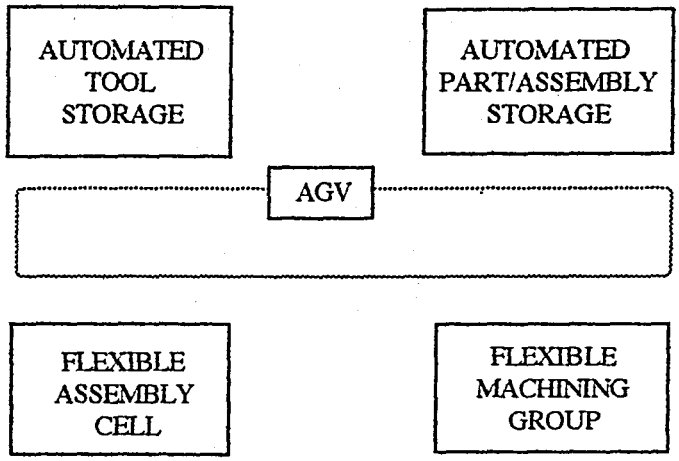

FIGURE 1.4. Flexible production system (FPS)

In this study, the overall decision process of FMSs consisting of the batching, loading and scheduling subproblems is analyzed in a hierarchical model. Hence, the effect of aggregation/disaggregation of information between the subproblems upon the feasibility of top-down commands is of particular importance. Capacity information including the machining and tool-magazine capacities is treated in an aggregate manner in the batching level before the loading decisions are made, and the capacity aggregation is done so as to ensure a feasible disaggregation in the loading level. These aggregation/disaggregation operations are performed in an instruction-reaction tandem process sometimes with imbedded anticipation preceding each instruction. Anticipation means that the top-level of a hierarchical model considers the relevant characteristics of the base-level and take the effects of these characteristics into account prior to making a decision. The top-level sends information to the base-level sometimes after this anticipation process, and the information which is a top-down influence of the top-level on the base-level is referred to as an instruction. One cycle of such an information exchange ends with the reaction of the base-level which is, in fact, a feedback to the top-level. In many cases, there are more than one cycle which is called a tandem until the top- and base-levels agree upon a decision. The sequence of all tandems build up a tandem process as the one between the batching and loading levels in this study.

The loading problem is formulated with different objectives as integer models with many variables so that the problem becomes hard-to-solve. Thus, Lagrangian relaxation method is applied to the model creating easier problems complicated by a relatively small set of constraints. Since it is not always possible to obtain feasible solutions by this method, a

Lagrangian heuristic is developed to make the solutions feasible. The overall applications are performed iteratively by improving the Lagrange multipliers by the subgradient optimization procedure at each iteration.

Scheduling decisions form the lowest level of the hierarchy developed in this study. This level obtains the necessary information from the loading stage which forms the upper level of the scheduling problem. A dispatching algorithm is developed for this level, and the effect of different dispatching rules on the performance measures of an FMS environment is analyzed. In order to perform this analysis, model parameters are varied to create an AGV bottleneck system.

This study is organized as follows: Literature survey about the subproblems stated above are presented in Section 2 of this study. Section 3 gives a general view about how the hierarchical structure of these subproblems are formed and about the characteristics of the FMS and of the material-handling system considered in this study. The top level of the hierarchy, batching, is explained in detail in Section 4. Section 5 gives the structure of the loading model. Lagrangian relaxation based solution approach to the loading problem is studied in Section 6. Capacity aggregation which is an important relationship between the batching and loading levels is fully examined in Section 7. Scheduling stage which constructs the lowest level of the hierarchy is explained in Section 8, where an algorithm developed for scheduling and several results obtained by the application of this algorithm are also presented. Section 9 states conclusions about this study and provides extensions about the subject for future study.

2. LITERATURE SURVEY

The hierarchical model analyzed in this study consists of batching, loading and scheduling subproblems. Each planning function has been extensively studied in literature with varying model formulations and solution approaches.

Co *et al.* [3] developed a mixed integer programming formulation which solved the batching, machine loading and the tool-magazine configuration problems simultaneously, and introduced a four-pass heuristic approach. Liang and Dutta [4] also presented an integrated approach to simultaneously solve part selection and machine loading problems in a class of FMSs. They developed a primary objective and secondary objectives and considered each objective in one of their models. A Lagrangian relaxation based solution method, incorporating the decomposition principle and column generation scheme, was also developed. Co *et al.* [3] and Liang and Dutta [4] introduced integrated approaches for the part selection and machine loading problems to avoid inconsistencies caused by the conflicts between the two sets of individually obtained solutions.

Hwan and Shogan [5] classified the FMS operations decisions as pre-release and post-release decisions. They considered part selection problem among the pre-release decisions and proposed a heuristic approach to model the realistic part selection problem. They designed two models under this approach. The basic model considered only the tool magazine capacity constraint while the second one considered both the capacity constraint and the due-date constraints. They also developed a model taking the machine balancing constraint into account. They introduced a solution structure with three methods, each one embedded within another. The outside layer of the proposed structure was a branch-and-bound procedure, where a number of subproblems would be generated and solved. In the middle layer, each problem was processed by a number of subgradient iterations using the Lagrangian relaxation method. In the innermost layer, a Lagrangian heuristic was called when needed to provide a feasible solution to each subgradient iteration.

Shanker and Tzen [6] considered the loading problem with the bi-criterion objective of balancing the work-load among the machining centers and meeting the due-dates of the jobs. A mathematical model was developed and heuristic procedures were suggested.

Sarin and Chen [7] developed a mathematical model to determine the routings of parts through the machines and to allocate appropriate cutting tools to each machine to

achieve minimum overall machining cost. They also stated that the model could be viewed as a quasi-generalized assignment problem which was considerably large in terms of variables and constraints and was often hard-to-solve. Hence, they applied Lagrangian relaxation to the model and discussed the computational refinements based on this approach.

Kirkavak and Dinçer [8] introduced several objectives for the loading problem. Maximizing the minimum target machine utilization was the first objective proposed, and four heuristic solution techniques were presented to solve that model. The primary formulation was later extended and two new models were developed. Minimizing the difference between maximum and minimum machine utilizations and minimizing the number of parts processed on different machines were the other objectives that were stated by the new models.

Chen and Askin [9] developed a multi-objective loading problem considering minimizing the maximum difference between machine utilizations, minimizing the inter-machine part movements, maximizing the routing flexibility, minimizing the tool investment and minimizing the maximum machine utilization. They also proposed loading heuristics requiring three basic decision steps as selection of machine type for each operation, grouping of machines within a machine type and assignment of tools and operations to machine groups.

Stecke [10] presented a set of five production planning problems to be solved for efficient use of an FMS, and addressed specifically the grouping and loading problems. The loading objectives were (a) balancing the assigned machine processing time, (b) minimizing the number of movements from machine to machine (the number of consecutive operations on each machine), (c) balancing work-load per machine for a system of groups of pooled machines of equal sizes, (d) unbalancing the work load per machine for a system of groups of pooled machines of unequal sizes, (e) filling the tool magazines as densely as possible, and (f) maximizing the sum of operation priorities. Also five linearization methods were introduced since almost all of the objective functions and some of the constraints were nonlinear.

Kim and Yano [11] tried to allocate operations and tools to machines or machine groups in such a way that work-loads of the machine groups were as close as possible to the ideal work-loads subject to tool magazine capacity constraints and machine capacity constraints. They also considered the fact that operations could share common tools, which they called tool commonality. In order to solve the problem, they developed heuristic

algorithms which were modifications of various single-pass and multiple-pass algorithms for multi-dimensional bin-packing problems.

Sodhi *et al.* [12] classified FMS operating policies into two main categories. The first category permitted dynamic tool changes while the second one consisted of static tool change policies. They developed four types of policies for the second category. The first and the second policy did not allow partial production, and the remaining two policies accepted some successive operations to be performed in later time periods of the planning horizon. In the first and the third policies, set-up costs were not considered while the others took these costs into account. They also proposed a heuristic to solve the model for the policy which did not allow partial production and which did not consider set-up costs.

Ram *et al.* [13] modeled the machine loading and tool allocation problem as a discrete generalized network with some simple side constraints, and they developed an algorithm to solve that model as a branch-and-bound procedure.

Leung *et al.* [14] addressed part assignment and tool allocation concurrently with explicit considerations given to material handling issues, and they provided a basic planning model formulated as a linear integer model with the objective of cost minimization. They also introduced several alternative formulations to minimize the sum of total processing time and material handling time and to balance the machine work-loads.

Mukhopadhyay *et al.* [15] suggested a heuristic solution to the loading problem by developing the concept of essentiality ratio for the objective of minimizing the system unbalance and thereby maximizing the throughput.

The lowest level of the hierarchy proposed in this study determines machine and material handling equipment scheduling, and there is a wide base of literature on this problem. The literature considers two main subjects related to this problem; (a) approaches for scheduling machines and material handling equipment and (b) dispatching rules that can be used in the scheduling approaches.

Sabuncuoğlu and Hommertzheim [16] proposed an on-line dispatching algorithm for the FMS scheduling problem. The algorithm they introduced used various priority schemes and relevant information concerning the load of the system and the status of jobs in the scheduling process. The scheduling decision process was hierarchical in the sense that different decision criteria were applied sequentially to identify the most appropriate part and machine to be served.

Mukhopadhyay *et al.* [17] stated that the scheduling problems in FMSs deal with (a) tool allocation, (b) parts scheduling, (c) pallets scheduling, (d) machines scheduling and (e) material handling equipment scheduling, and presented an approach to determine an optimal schedule of parts integrating all these scheduling criteria. The problem was formulated as a hierarchical process and solved through eigenvector analysis of priority ordering.

Ulusoy and Bilge [18] decomposed the scheduling problem into two subproblems and solved it by an iterative heuristic procedure. At each iteration, a new machine schedule, generated by a heuristic procedure, was investigated for its feasibility to the vehicle scheduling problem. The operation completion times obtained from the machine schedule were used to construct time windows for each material handling trip, and the second supbroblem was handled as a sliding time window problem.

Moreno and Ding [19] dealt with the concurrent solution of the loading and scheduling problems in an FMS environment. A heuristic approach using a constructive scheduling method was developed to solve these problems concurrently.

Hirabayashi *et al.* [20] proposed a "decomposition scheduling method" for operating an FMS with small buffer capacity. In this method, a set of unscheduled jobs were decomposed into several sets of jobs, and the optimum subschedules were derived independently for each set of jobs. A near-optimum schedule was obtained by connecting these sub-schedules arbitrarily.

Kim [21] considered the flowshop scheduling problem for the cases in which job sequences on all machines were the same and in which they might be different. For the former case, various methods that had been devised for minimizing the makespan were modified according to their objective, and the list scheduling algorithm was used for the latter case.

Aanen *et al.* [22] introduced a branch-and-bound algorithm based on an active schedule strategy for a two-machine scheduling problem. In an active scheduling approach, the set of all possible schedules is reduced to the subset of active schedules. They also described a procedure to find lower bounds for active schedules.

In Stecke and Solberg [23] and Sabuncuoğlu and Hommertzheim [24], a list of scheduling rules which were tested and the results of the experimental conditions can be found. AGV dispatching rules which were classified into work center-initiated rules and

vehicle-initiated rules can be investigated in Yim and Linn [25]. Also, Denzler and Boe [26] investigated the scheduling decision rules for a dedicated FMS.

Egbelu and Tanchoco [27] classified the dispatching decisions into two categories as "work center-initiated task assignment (dispatching) rules" and "vehicle-initiated task assignment (dispatching) rules". They put forward the likely effects of these rules on the performance of a job-shop.

The hierarchical structures in the organizations which is of particular importance in this study was discussed in Schneeweiss [28] in detail.

Bitran *et al.* [29] presented a hierarchical approach to plan and schedule production in a manufacturing environment that can be modeled as a single stage process. Special attention is given to aggregation/disaggregation procedures, and several improvements in the methodology related to hierarchical production planning are suggested.

Barbarosoğlu [30] also applied the hierarchical decision-making in production planning by designing a bi-level hierarchical structure to make tactical decisions at the top-level and operational decisions at the lower-level. The lower-level decision process is further decomposed into a hierarchy of subproblems. The hierarchical production planning model proposed in that study allowed aggregation/disaggregation between levels.

Axsäter [31] studied under what conditions a perfect aggregation of product data in a hierarchical structure can be obtained. A mathematical formulation of an approximate model when a perfect aggregation is not possible is also proposed in the study.

Axsäter and Jönsson [32] presented a simulation study of hierarchical planning procedures, which can support a material requirements planning system. An aggregate plan in terms of product groups and machine groups is derived, and then the plan is disaggregated by changing order release times obtained from material requirements planning and by distributing extra capacity among individual machines.

Axsäter [33] described including constraints upon the aggregate level product groups that will guarantee the existence of a feasible disaggregation at the lower detailed level in a hierarchical planning system with two levels applied to a multistage production process.

The Lagrangian relaxation method which is used in the loading level of the hierarchy in this study was fully explained in Fisher [34], and this method and its applicability in integer programming are discussed in Geoffrion [35].

Shapiro [36] combined the theory of generalized Lagrange multipliers with a reformulation of the integer programming problem due to group theory.

Fisher [37] presented an algorithm for solving resource-constrained network scheduling problems by using Lagrange multipliers to dualize the resource constraints, forming a Lagrangian problem in which the network constraints appear explicitly, while the resource constraints appear only in the Lagrangian function.

Held *et al.* [38] studied the computational performance and theoretical convergence properties of the subgradient method.

It must also be stated that a broad classification of studies on FMSs was provided in Gunasekaran et al. [39] and Hedin et al. [40].

3. PROBLEM DEFINITION

The purpose of this study is to analyze the FMS batching, loading and scheduling decision problems by employing different models and solution procedures within a hierarchical structure and to compare the effectiveness of these approaches with respect to different performance criteria.

The basic objective of the flexible manufacturing concept is to achieve the efficiency and utilization levels of mass production while retaining the flexibility of manually operated job-shops [23]. In an FMS, machines are equipped with sensors, tool magazines and automatic tool changing mechanisms so that a wide variety of different parts can be machined automatically with minimum changeover time. Furthermore the material handling system is designed to be flexible and connects various work centers of the system over a common computer control infrastructure. The powerful innovation in such a production environment addresses a set of complex operational problems which require an integrated solution framework of several combinatorial optimization models.

The operational problems in an FMS encompass different levels of management responsibility and different degree of detail of the required information. They also differ in terms of the length of the planning horizons and objectives. While planning production which addresses such decision problems, one possibility is to treat the whole planning system as a single model which can be solved in one step with a "monolithic approach" [30]. However, not only the formulation of an appropriate mathematical model is an impractical effort, but also the solution of such a complex problem and the interpretation of results is almost impossible. Thus, one way of avoiding these difficulties is to decompose the FMS operational decision-making into subproblems and to develop a decentralized and hierarchical design. Within the hierarchical approach, decisions are made in sequence with varying extent of information content.

The most frequently referenced framework among the hierarchical decision frameworks which have been developed for FMSs is the hierarchical scheme in [10]. The set of subproblems with high interaction is solved sequentially as shown in Figure 3.1 as the short-term production planning process. In such a multi-level decision system, the decisions at a high planning level lead to some restrictions on the detailed decisions at a lower level to serve as a link between the aggregate and detailed decisions. A hierarchical production

planning system in an FMS is a top-down planning approach with bottom-up feedback between any two planning levels.

The hierarchical structure developed in this study covers only the part type selection, loading and scheduling problems. The part type selection problem forms the aggregate level of the hierarchy. The aggregate decisions determine which parts should be processed on the shop-floor considering aggregate capacities for a predetermined period of time. The lower-level problems are solved for each subset of part types generated in the batching level using detailed information. A more detailed planning level which is referred to as the loading problem assigns all the operations and the required tooling for the selected part types to the machine centers. The implication is that the tooling required to support each operation must be assigned to its associated machine center. Scheduling decisions which are placed at the bottom of the hierarchy determines which part in a queue of available parts is dispatched to an idle machine center by being transported by an idle AGV.

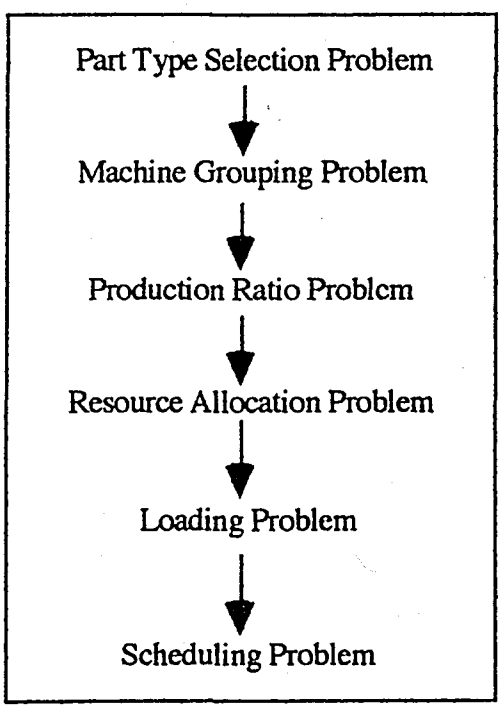

Figure 3.1. The hierarchical scheme in [10].

3.1. Description of the Hierarchical Structure

The hierarchy proposed in this study has three levels. The top level deals with batching decisions, the second level solves part loading and tool allocation problem, and the lowest level determines machine and material handling equipment scheduling.

Given a set of parts to be processed during a certain planning period,

(a) *batching* determines the subset of parts with a common tooling to be manufactured together during a shift,

(b) *loading* assigns all the operations and the required tooling for the parts in the batch to the machine centers,

(c) *scheduling* defines the sequence of parts dispatched to the machine centers in an automated material handling system consisting of automatic guided vehicles (AGVs) and the sequence of parts processed in the machine centers.

These planning functions are sequential in nature, and the related decision subprocesses are hierarchical with a strong interaction. In fact, once an upper level function is executed, it communicates its instructions to and monitors information from the lower level. Figure 3.1.1 shows the conceptual hierarchical model composed of three main modules and interface programs, which will be discussed in detail in the following sections.

In this structure, the batching model which is developed as an integer program is solved by executing HYPER LINDO so that the subset of parts to be manufactured together during a shift is determined. Then, by means of an interface program, this information is converted to data necessary to form the loading model. The integer loading model is also solved by first using HYPER LINDO and then using the Lagrangian relaxation approach developed in this study. At this point, it must be noted that since the batching model considers the machine and tool magazine capacities in an aggregate manner as a whole system capacity, the loading model may turn out to be infeasible, and the batching model has to be solved again. Hence, an extensive information exchange may take place between the batching and loading levels in an iterative manner to obtain a batch of parts feasible from the loading point of view. This information exchange is studied by three different methods: Pure top-down hierarchy, explicit anticipation approach and the heuristic anticipation process developed in this study. In the pure top-down hierarchy, the batching model is solved to yield a factual instruction without any formal anticipation and then the loading model is solved using this instruction. In the explicit anticipation approach, an anticipation function

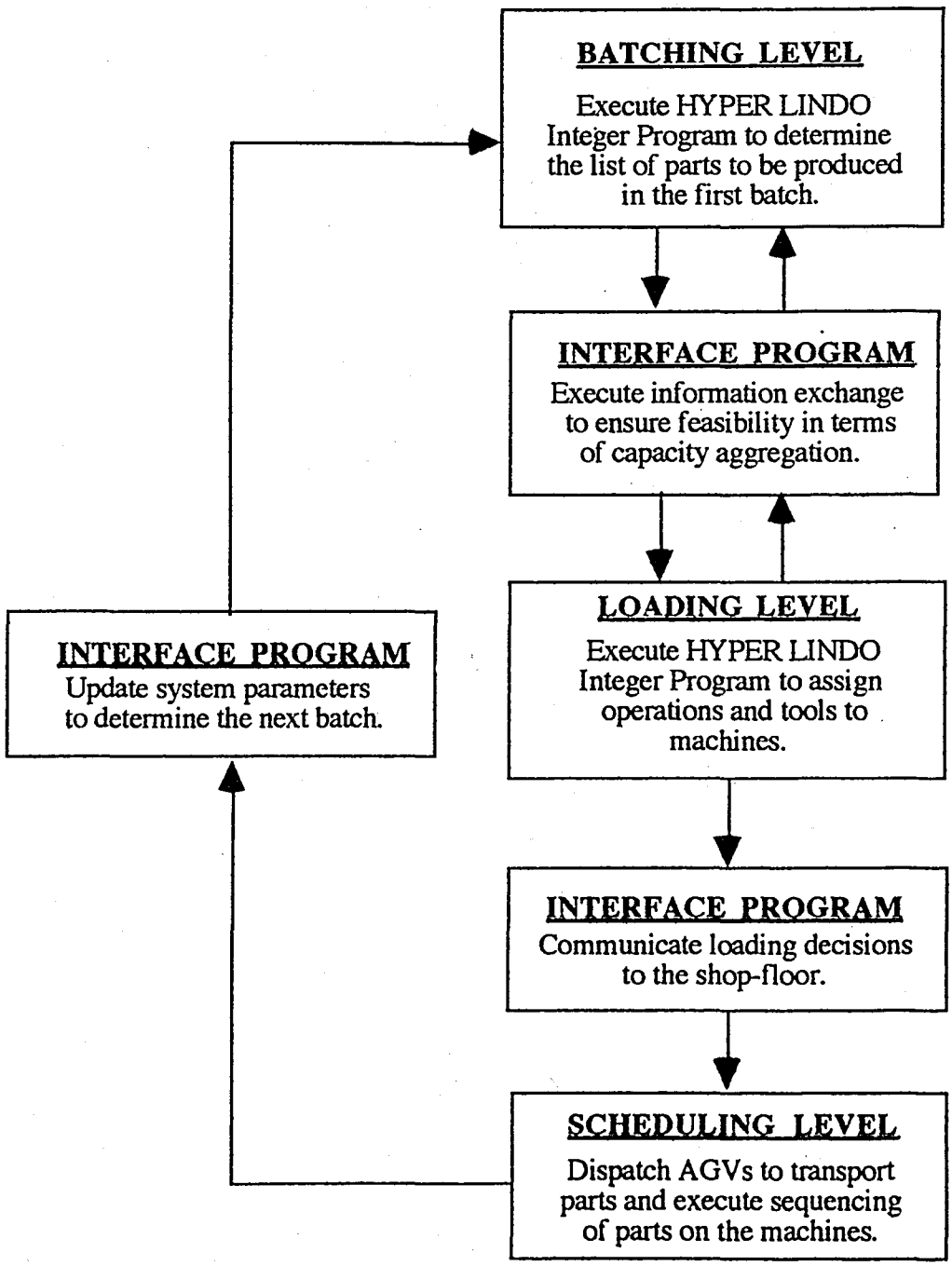

FIGURE 3.1.1. The flow-chart of the conceptual hierarchical model

based upon the reaction of the base level as the information content of the top level is also considered. The anticipation process developed in this study is, on the other hand, a heuristic process which expands the anticipation content about the base level iteratively and studied in Section 7.3 in detail. When the operations of the parts in the batch and the corresponding

tools are assigned to certain machines in the loading level, the information is transformed to the shop-floor, where scheduling operations take place. In the scheduling level, the AGVs are dispatched to transport parts and the parts are sequenced on machines by means of a scheduling algorithm, which includes various decision points where certain rules are required. Since the system performance is sensitive to the joint-effect of all rules imbedded in the algorithm, all relevant combinations of these rules are tried in this study. The relative performance of rules is also compared against different measures. At the end of this procedure, according to the current status of the shop-floor, the system parameters are updated in order to be able to determine the next batch. It must be stated that the interface programs which enables the information exchange in this procedure are written in Pascal.

3.2. Description of the FMS

The FMS designed in this study is assumed to consist of M machining centers and a central buffer integrated through an automated material handling system and T different types of machining tools. Each machine consists of an input buffer where it receives the part to be processed and an output buffer where the processed part is released upon the completion of processing. Manufacturing each part requires a sequence of operations which can be performed on different machines with different tool requirements where machines are assumed to be perfectly flexible. Thus, the operation-machine-tool compatibilities and the corresponding processing times are assumed to be known. Each tool, on the other hand, occupies a given number of tool slots in the tool magazines of the machines, and the tool magazine capacity cannot be exceeded. Machining capacities and tool magazine limitations are assumed to be known.

In the system, there is a production order to manufacture a given set of N different parts, each of which requires $|O(i)|$ operations to be performed. First the parts are partitioned into batches according to technological similarity in the batching level, and the parts in the first batch are simultaneously released to the central buffer. The machines are loaded and the tools are allocated to the machines according to the results of the loading model. Once the machine loading and tool allocation is done, the parts are scheduled among the machines according to a scheduling algorithm. The part returns to the central buffer only after all its operations are performed.

3.3. Description of the AGV System

One of the key factors in the total productivity of an FMS is material movement. Without controlled material movement within the manufacturing shop, wasted utilization occurs in manpower, floor space and machine tools. Material handling systems (MHSs) must integrate a variety of equipment in the FMS and must also be reliable, fast and easy to maintain. Six types of MHS have been identified for use in FMSs: *Belt conveyors* are often used to transport parts down the line. When the appropriate part reaches the machine, a robot arm can be used to pull it off. *Roller conveyors* (chain or chain-driven) are similar to a belt conveyor with different mechanical aspects. *Power-and-free conveyors* have trolleys located in a free rail and are powered by a continuously moving chain. Free trolleys engage and disengage at the appropriate machine tools. *Monorails* and *monotractors* are similar to the power-and-free conveyors but usually handle higher volumes and lower unit weights of material. *AGV systems* are the most flexible MHS type. The MHS type that is considered in this study is an AGV system.

The design of an AGV system addresses the determination of the layout of the AGV tracks, the traffic flow pattern along the AGV tracks and the location of buffers. Basically there are three types of material handling systems: single line, loop (single / multi) and network type. Figure 3.3.1 shows the single-loop layout of the hypothetical model developed in this study for the 6-machine case. There is a central buffer where initially all unprocessed parts wait until the first operation is initiated and return back only upon the completion of the processing of all operations. There is no capacity restriction upon the central buffer. AGVs transport the parts between the machines along a predetermined path, which is assumed to be unidirectional. When an AGV completes a part transfer, it stays there at the same machine waiting for the next task assignment. As far as the link between the machines and the AGVs is concerned, "a direct access part retrieval design" is assumed which means that any part from the output queue can be retrieved regardless of its position in the queue. The AGVs in the system are assumed to be of the same type and of "unit load" type of vehicles, and the number of AGVs is one of the variable model parameters.

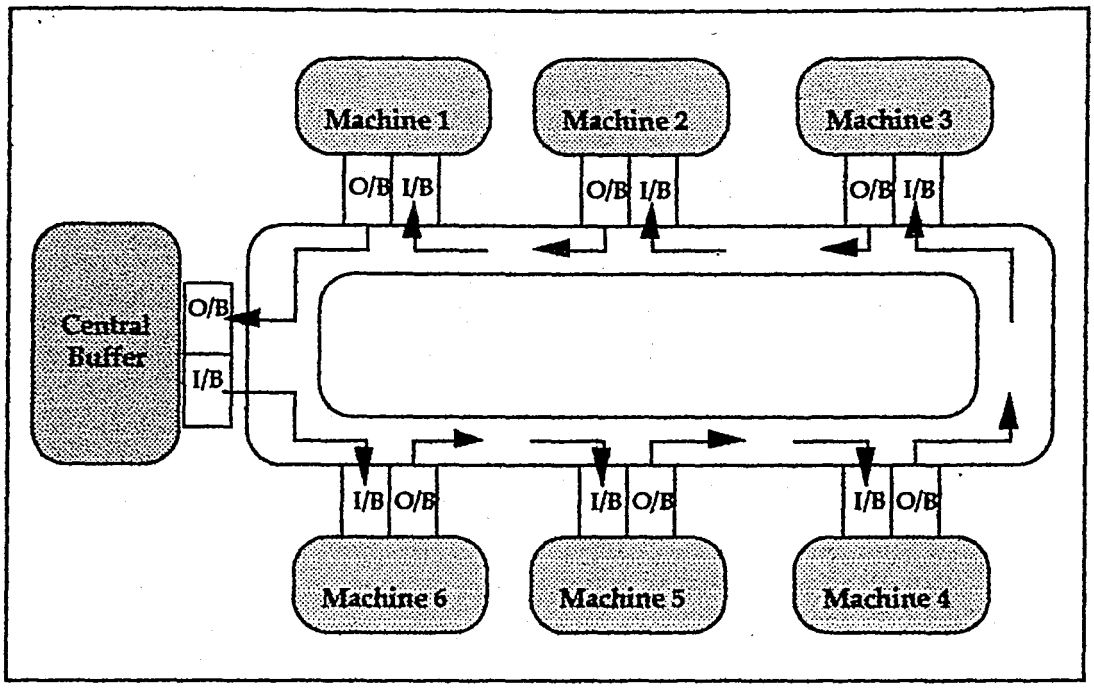

FIGURE 3.3.1. Layout of the hypothetical FMS (6 - machine case)

3.4. Design of Test Problems

Various test problems are generated to analyze the performance of the 3-level hierarchical approach proposed in this study by varying the number of parts, the number of operations each part possesses and the number of machining centers. The number of parts is varied between four and 30; the number of operations per part is varied between four and eight and the number of flexible machines is varied between four and eight. The number of tool types is taken to be 20 in all test problems. The machine hour availabilities are varied between 50 and 500. Although in many experiments machine hour availabilities are assumed to be the same for all the machines in the system, there are several test problems where machine capacities differ from each other. The tool magazine capacities are taken to be the same for each machining center in all test problems and varied in a range of 60 and 80 slots.

These system parameters together with processing times, machine-tool compatibilities and cost factors are generated from Uniform Distribution within stated limits

by using SLAM II. A typical data set is given for a problem with four parts, four operations, 20 tools and four machines in Table A.3.4.1.

4. BATCHING LEVEL

FMS operations consist of pre-release decisions and post-release decisions. Pre-release decisions consider the pre-arrangement of parts and tools before the FMS begins to process. Post-release decisions, on the other hand, deal with sequencing and routing parts when the system is in operation. According to these definitions, the batching problem covers pre-release decisions which determine the pool of parts eligible for release to the system.

Given a set of jobs to be processed over a short-range horizon, the jobs in an FMS may have to be partitioned into batches for two major reasons:

(a) Maintaining proper shop-floor control may dictate that smaller variety of jobs be loaded concurrently on the manufacturing facility.

(b) The FMS resources are finite and limited. It is not often possible to set up the equipment with the capabilities of all required operations at once. Such limited resources as pallets and fixtures must be changed or reassigned, and tool magazines must be reconfigured after one batch is completed so as to process another batch.

By means of batching, fewer classes of jobs flowing in the system are obtained for a shift, and this means that less non-identical tooling is required by the system for that horizon. Less non-identical tooling leads to the fact that machines can have more commonality in tooling. Pooling of machines is thus possible, and productivity is enhanced. Furthermore, less variety of jobs means diverse routing and, therefore, few complications in scheduling and less incidence of blocking.

The most critical limitation of FMS is usually the tool magazine capacity. A part type has several operations each of which may require different kinds of tools. Some of these tools may occupy several slots on the machine's tool magazine while the others may need just one. When each machine has a limited number of slots, the total number of slots in the FMS is fixed so that it is not possible to mount all of these tools on the magazines of the FMS concurrently. This limitation is called the system's magazine capacity. The magazine capacity constrains the number of part types that can be processed simultaneously. Batching is not needed if the system's magazine capability is large enough to process all the part types. It must be noted that if an FMS has automated tool exchange devices which ship tools around the machines and the central storage during the time the system is in operation, there is no need to include the batching problem. However, most of the FMSs do not have automated tool exchange devices, and the batching decisions are still important.

When a batch of parts is selected, the set of tools needed to process them is computed by using the part-tool-machine compatibility data in a loading model. Tool changes within a batch are not allowed. When a batch finishes processing, the tools are unloaded from the tool magazines, new tools are loaded, the working area is cleaned, the machines are maintained, and a new batch can start processing. This process, called a set-up, usually requires a fixed amount of time and arises from the critical limitation of tool magazine capacity. Thus, the objective of the batching model is chosen to minimize the total set-up time.

Since each set-up requires a fixed amount of time as stated above, fewer number of set-ups means less total set-up time. So, the aim of minimizing total set-up time may be viewed as minimizing the number of set-ups, and the number of set-ups can be minimized only by minimizing the number of batches since a set-up takes place between the processing of two respective batches. The minimization of the number of batches is, on the other hand, equivalent to the maximization of the number of parts included in each batch. Hence, the objective of the batching model used in this level is to maximize the number of parts in a given batch, which in turn will tend to fully utilize the tool magazine capacity.

Batching decisions require the consideration of two important limitations in an FMS. The first one is the tool magazine capacity of the system which is discussed above. The other one is the machine-hour availability of the FMS. As far as machine capacities are considered, there exist two approaches: variable period and fixed period. Once a fixed-period approach is assumed, the length of the batching planning horizon is assumed to be fixed, and the machine capabilities during that period are known and should be included in the model. However, when the variable-period approach is preferred, it is not necessary to consider the system capacity as a constraint in the batching problem because once a set-up is done, the batch will be processed continuously without interruption, so that the batch size will determine the length of the batching planning period. In this model, the fixed-period approach is employed to determine the batch of parts to be manufactured within one set-up during a shift or a day, and thus, it is necessary to include the machine capacities over the fixed-period. However, since the loading problem has not been solved yet, the exact part-machine-tool assignments are not known and capacity limitations should be considered in an aggregate manner; in fact capacity aggregation implies that machines are assumed to be identical in terms of their ability to accept different tools and the whole system is treated as one machine. Thus, the aggregation should be designed for both machining capacity and tool-slot capacity so as to guarantee a feasible link between the batching and loading stages and will be discussed later in detail.

The batching problem is developed as an integer 0-1 mathematical model to determine the first batch in a greedy manner. The mathematical model is given by **(M1)**:

(M1)

Maximize $$V = \sum_i x_i \qquad (4.1)$$

subject to $$x_i \le \frac{\sum_{j \in O(i)} y_{ij}}{|O(i)|} \quad \forall i \qquad (4.2)$$

$$a_{ijt} \cdot y_{ij} \le z_t \quad \forall t, \forall i, \forall j \in O(i) \qquad (4.3)$$

$$\sum_t N_t \cdot z_t \le \hat{S} \qquad (4.4)$$

$$\sum_{j \in O(i)} \sum_i \hat{p}_{ij} y_{ij} \le CAP \qquad (4.5)$$

$$y_{ij} \le x_i \quad \forall i, \forall j \in O(i) \qquad (4.6)$$

$$x_i = (0,1) \quad \forall i \qquad (4.7)$$

$$y_{ij} = (0,1) \quad \forall i, \forall j \in O(i) \qquad (4.8)$$

$$z_t = (0,1) \quad \forall t \qquad (4.9)$$

Here the decision variables are defined to be:

i : part i, $i = 1,\ldots,N$

j : operation j, $j \in O(i)$

t : tool type t, $t = 1,\ldots,T$

m : machine m, $m = 1,\ldots,M$

$$x_i = \begin{cases} 1 & \text{if job i is included in the batch} \\ 0 & \text{otherwise} \end{cases}$$

$$y_{ij} = \begin{cases} 1 & \text{if operation j of part i is processed in the batch} \\ 0 & \text{otherwise} \end{cases}$$

$$z_t = \begin{cases} 1 & \text{if tool t is required in the batch} \\ 0 & \text{otherwise} \end{cases}$$

The parameters are:

$O(i)$: set of operations that are needed to manufacture job i.

$|O(i)|$: number of operations that are needed to manufacture job i.

a_{ijt} : a binary variable indicating if the operation j of part i requires tool t on any machine.

N_t : number of tool slots occupied by tool t.

$\hat{P}_{ij}$: aggregate processing time of operation j of part i.

$\hat{S}$: aggregate tool magazine capacity.

CAP : aggregate system capacity.

The objective function given by (4.1) aims at maximizing the number of parts in each batch as it is discussed previously. The constraint set (4.2) ensures that a job is included in a batch only if all its required operations can be performed in the given batch configuration. The fact that an operation in a batch can be performed only if the required tools for the operation are installed is expressed by (4.3). The aggregate tool magazine capacity is expressed by (4.4) while the aggregate system capacity is given by (4.5). The technical constraint (4.6) guarantees that an operation will be processed in a given batch only if the associated part is included in the batch.

A Heuristic Approach to Solve the Batching Model:

Although there are various approaches to solve the batching model , a greedy approach is recommended as a realistic solution procedure. In the proposed heuristic, the entire order is not divided into batches all at once. Instead, a first batch of maximum size from the whole order is selected as the solution of (M1). After it is produced, a second batch of maximum size is then selected from the current set of job orders, and so on, until the order is exhausted. This dynamic heuristic called Procedure BTC can shortly be described as follows:

Procedure BTC:

Step 0: Initialize $b = 1$.

Step 1: $J_b^c = J_o + \bigcup J_k^n - \bigcup_{k=1}^{b-1} J_k^a$

If $J_b^c = 0$, $B = b - 1$, stop.

Step 2: Solve the batching model (M1) for x_j for all $j \in J_b^c$.

Step 3: Set $J_b = \{j \mid x_j = 1\}$. Produce the batch.

Step 4: Update J_k^n and J_k^a.

Step 5: Set $b = b + 1$, go to Step 1.

where

J_o: set of parts under consideration.

J_k^c: set of parts that are candidates for assignment to batch k.

J_k: set of parts actually assigned to batch k.

J_k^a: set of parts actually produced in batch k.

J_k^n: set of parts that arrive during the processing of batch k.

This is a greedy approximation, which results in a sequential procedure. In this fashion, it is possible to keep enough flexibility to cope with incoming orders while the current order is being processed, with machine breakdowns or with other dynamic and unpredictable conditions. The batching submodel given in Step 2 is solved by using HYPER LINDO.

5. LOADING LEVEL

The top level of the hierarchy proposed in this study selects a subset of parts which is called a batch to be produced simultaneously within a single set-up of the system. However, the exact part-tool-machine assignment decisions are not made in this level, and the tool magazine capacities and the machine availabilities are considered in aggregate terms while determining the batching decisions. Thus, the problem in the loading level is to allocate the operations of the parts in the selected batch and the tools required by these operations among a set of machines so as to achieve certain specified objectives while meeting technological and capacity constraints. So, the problem can also be viewed as a machine loading and tool allocation problem.

At the loading stage, more detailed information in terms of the parts to be processed, the tools which are available to perform the operations of these parts, machine-tool compatibilities and capacity limitations are known to the decision-maker. Hence, the aggregate capacities used in the batching level are considered separately for each machine and for each tool in the loading level, and the assignment of each operation and of the required tools to the machines according to the compatibilities is done accordingly. Once these decisions are made, the tools stay on the machines they are assigned to for the planning period. The parts are then routed through the machines where the needed tooling and NC programs are already loaded.

It must be emphasized that there may be several desirable objectives that loading decisions may possess and each of these objectives can lead to different solutions. Hence, it is possible to generate several alternative solutions to the loading problem by giving greater or less emphasis to one versus the others. So, by taking this fact into account, the loading model developed in this study is not formulated with a unique objective but with different objectives.

The loading model (LM) developed consists of the following technological and capacity constraints :

(LM)

$$\sum_i \sum_j \sum_t p_{ijtm} x_{ijtm} \leq \alpha_m . M_m \qquad \forall m \qquad (5.1)$$

$$\sum_t N_t . y_{tm} \leq S_m \qquad \forall m \qquad (5.2)$$

$$\sum_i \sum_j \sum_m p_{ijtm} x_{ijtm} \leq \beta_t . T_t \qquad \forall t \qquad (5.3)$$

$$\sum_m y_{tm} \leq A_t \qquad \forall t \qquad (5.4)$$

$$\sum_i \sum_j x_{ijtm} \leq M . y_{tm} \qquad \forall t, m \qquad (5.5)$$

$$\sum_t \sum_m x_{ijtm} = 1 \qquad \forall i, j \qquad (5.6)$$

$$x_{ijtm} = (0,1) \qquad \forall i, j, t, m \qquad (5.7)$$

$$y_{tm} = (0,1) \qquad \forall t, m \qquad (5.8)$$

Here the decision variables are given by

$$x_{ijtm} = \begin{cases} 1, \text{ if operation j of part i is assigned to machine m containing tool t.} \\ 0, \text{ otherwise.} \end{cases}$$

$$y_{tm} = \begin{cases} 1, \text{ if tool t is assigned to machine m.} \\ 0, \text{ otherwise.} \end{cases}$$

where t indicates the tools the operations in the batch may require, and m shows the machines which are compatible with these tools.

The parameters are defined by

P_{ijtm} : processing time of operation j of part i on machine m using tool t.
M_m : total number of machining hours available on machine m.
α_m : efficiency of machine m.
N_t : number of tool slots occupied by tool t.
S_m : tool magazine capacity of machine m.
T_t : service life of tool t in hours.
β_t : reliability of tool t.
A_t : number of available tool t.

In the above formulation the machine capacity constraints are given by (5.1) and the tool magazine capacity limitations are given by (5.2). Since the tools wear out very rapidly, the tool life-time considerations are expressed by (5.3). In most FMSs the number of tools is limited and should be included as a constraint similar to (5.4). The technical constraint (5.5) ensures that the necessary tools should be installed on a machine to perform a particular operation. The technical constraint (5.6), on the other hand, guarantees that all of the operations of all parts in a given batch be performed.

As it is mentioned before, the loading problem can be formulated with different objectives. The following objective functions are used in this study:

(a) Minimizing cost:

Here it is necessary to define c_{ijtm} as the machining cost of performing operation j of part i on machine m by tool t. This cost may be proportional to the processing times for the (i,j,t,m) combinations or may be related to the maintenance costs of the machines and/or tools in the system. Then, the objective functions is

$$\text{Min } Z_1 = \sum_i \sum_j \sum_t \sum_m c_{ijtm} x_{ijtm} \qquad (5.9)$$

(b) Balancing the workload:

Balancing the workload can be done by minimizing the unbalances of total processing time assigned to each machine. Then it is necessary to replace constraint set (5.1) by

$$\sum_i \sum_j \sum_t p_{ijtm} \cdot x_{ijtm} + U_m - O_m = \alpha_m \cdot M_m \qquad \forall m \qquad (5.1')$$

where U_m is the underload and O_m is the overload on machine m. Then the objective function is

$$\text{Min } Z_2 = \sum_m (U_m + O_m) \qquad (5.10)$$

(c) Minimizing the maximum workload:

It is necessary to include another constraint set to (5.1) - (5.8).

$$\sum_i \sum_j \sum_t p_{ijtm} \cdot x_{ijtm} \leq Z \qquad \forall m \qquad (5.11)$$

where Z will give the maximum workload in the system and the related objective function is

$$\text{Min } Z_3 = Z \qquad (5.12)$$

(d) Minimizing the differences between the workloads of the machines:

It is necessary to determine the workload of each machine by

$$\sum_i \sum_j \sum_t p_{ijtm} \cdot x_{ijtm} = WL_m \qquad \forall m \qquad (5.13)$$

and to express the positive and negative deviations between the workloads by

$$WL_k - WL_h - R_s + L_s = 0 \qquad \text{for } k \in (1,\ldots,m-1),\ h \in (k+1,\ldots,m),\ s \in \left(1,\ldots\binom{m}{2}\right) \tag{5.14}$$

Then constraints (5.13) and (5.14) should be included to (5.1) - (5.8), and the objective function becomes

$$\text{Min } Z_4 = \sum_{s=1}^{\binom{m}{2}} (R_s + L_s) \tag{5.15}$$

To facilitate the excessive number of runs, an interface program is prepared to link the batching and loading models as shown in Figure 3.1.1.

Data for a 15-part, 8-operation, 6-machine problem are presented in Tables A.5.1 and A.5.2 in detail. Since the machine and tool-magazine capacities are aggregate in the batching level, the loading processing times are aggregated for batching decisions and given in Table A.5.3.

As it is stated above, the machine and tool-magazine capacities form the main information link between batching and loading levels, and an algorithm is developed to perform capacity/processing time/tool slot aggregation as an information exchange mechanism which will be discussed in Section 6. The above problem is solved in the batching level with CAP = 2000 minutes and the parts x_1, x_2, x_3 and x_{11} are included in the batch. To find the machine loading and tool allocation for this batch, the exact machine capacities are taken as 330 minutes per machine. The magazine capacity is assumed to be 60 tool slots, and the tool life is considered as 150 minutes for each tool type .It is solved under the four objective functions and the loading results given in Table 5.1 are obtained.

The loading results for the 30-part, 4-operation, 4-machine problem are also given in Table 5.2. Machine capacities are assumed to be 150 minutes per machine, while the first batch includes x_7, x_{12}, x_{15}, x_{20}, x_{21}, x_{24}, x_{27}, x_{29} and x_{30} with CAP = 600 minutes in the batching problem.

TABLE 5.1. Loading Results for the 15-part, 8-operation, 6-machine Problem

Objective Function	Work - Loads					
	M1	M2	M3	M4	M5	M6
Z_1	182 min	166 min	102 min	104 min	137 min	78 min
Z_2	168 min	133 min	147 min	105 min	236 min	138 min
Z_3	97 min	101 min	101 min	100 min	97 min	100 min
Z_4	138 min	139 min	143 min	139 min	139 min	134 min

TABLE 5.2. Loading Results for the 30-part, 4-operation, 4-machine Problem

Objective Function	Work - Loads			
	M1	M2	M3	M4
Z_1	149 min	142 min	142 min	149 min
Z_2	150 min	150 min	150 min	150 min
Z_3	130 min	130 min	131 min	132 min
Z_4	150 min	150 min	150 min	150 min

As far as the work-loads on each machine are concerned, it is observed that the minimum balanced work-loads can be obtained by the objective function Z_3, and Z_2 and Z_4 lead to high machine utilizations. Since they try to minimize the under- or over-loads on the machines, the higher processing times are first assigned to the machines.

6. LAGRANGIAN RELAXATION APPROACH TO THE SOLUTION OF THE LOADING PROBLEM

The loading problem in this study is a combinatorial optimization problem formulated as an integer programming problem, and it is considerably large in terms of variables and constraints and is often hard-to-solve. One of the most computationally useful ideas of the 1970s is the observation that many hard problems can be viewed as easy problems complicated by a relatively small set of side constraints. Dualizing the side constraints produces a Lagrangian problem that is easy to solve and whose optimal value is a lower bound (for minimization problems) on the optimal value of the original problem [34]. Over the last decades Lagrangian relaxation has turned out to be the best algorithm for the solution of famous combinatorial optimization problems of considerable size Hence, a solution approach based on the Lagrangian relaxation and subgradient method is proposed for the loading model to facilitate the computational effort, and an extensive Lagrangian design is developed with a hierarchical Lagrangian heuristic.

6.1. Basic Structure of the Lagrangian Relaxation Method

Let us consider the following integer program [34] :

$$
\begin{aligned}
&(IP) \qquad && Z = \min cx \\
&\text{subject to} && Ax = b \\
& && Dx \le e \qquad (6.1.1) \\
& && x \in \{0,1\},
\end{aligned}
$$

where x is n x 1, b is m x 1, e is k x 1 and all other matrices have conformable dimensions.

The constraints of (IP) is assumed to be partitioned into two sets $Ax = b$ and $Dx \leq e$ so as to make it easy to solve the Lagrangian problem

$$\textit{(LR)} \qquad Z(u) = \min \left[cx + u(Ax - b)\right]$$

$$\text{subject to} \qquad \begin{aligned} Dx &\leq e \\ x &\in \{0,1\}, \end{aligned} \qquad (6.1.2)$$

where $u = (u_1,\ldots,u_m)$ is a vector of Lagrange multipliers. By means of (LR), the problem becomes easy-to-solve relative to (IP).

It is clear that $Z(u) \leq Z$ and can be shown by assuming an optimal solution x^* to (IP) and observing that

$$Z(u) \leq cx^* + u\,(\,Ax^* - b) = Z \qquad (6.1.3)$$

The inequality in this relation follows from the definition of $Z_D(u)$ and the equality from $Z = cx^*$ and $Ax^* - b = 0$. If $Ax = b$ is replaced by $Ax \leq b$ in (IP), then u is required to be positive. In the same fashion, $u \leq 0$ is required for $Ax \geq b$ for $Z(u) \leq Z$ to hold.

The fact that $Z(u)$ is less than or equal to Z allows (LR) to be used to provide lower bounds for (IP). Good feasible solutions to (IP) can frequently be obtained by perturbing nearly feasible solutions to (LR).

In Lagrangian relaxation applications, obtaining values for Lagrange multipliers is an important task. It is clear that the best choice for u would be an optimal solution to the dual problem (LD):

$$\textit{(LD)} \qquad Z_D = \max_u Z_D(u) \qquad (6.1.4)$$

Although $Z_D(u)$ which is the envelope of a finite family of linear functions satisfies the continuity and convexity properties, it is non-differentiable at an optimal point. The best

known approaches for non-differentiable optimization include the subgradient optimization, simplex methods using column generation techniques and multiplier adjustment methods. In this study, the subgradient method in which gradients are replaced by subgradients is employed to obtain the u values as near-optimal solutions to (LD).

Given an initial value u^0, a sequence $\{u^k\}$ is generated by the rule

$$u^{k+1} = u^k + \theta^k (Ax^k - b) \qquad (6.1.5)$$

where x^k is an optimal solution to (LR^k), θ^k is a positive scalar step size, and $(Ax^k - b)$ which corresponds to the dualized constraints is the subgradient for which x^k solves LR.

6.2. Relaxed Loading Problem

The main issues considered in the development of a procedure based on Lagrangian relaxation include

(a) selection of constraint sets to be relaxed,

(b) solution of the Lagrangian problem with fixed multiplier values,

(c) use of an appropriate Lagrangian heuristic to obtain a feasible solution to the original problem.

While developing a Lagrangian problem, it is possible to choose different sets of constraints to be relaxed. However, each constraint set requires different lengths of time for the solution of the problem with varying sharpness of bounds. Hence, facilitating the computational effort highly depends on the choice of the constraint set to be relaxed, and the most appropriate set must be selected in order to handle the problems which are hard-to-solve as easy problems. In this study, constraints (5.1), (5.2), (5.3) and (5.5) are dualized so that the best Lagrangian problem is developed in terms of the amount of computation. The model obtained is as follows:

(RLM)

$$Z_D(v) = \text{Min} \sum_i \sum_j \sum_t \sum_m c_{ijtm}x_{ijtm} + \sum_m v^1_m \left(\sum_i \sum_j \sum_t p_{ijtm}x_{ijtm} - \alpha_m M_m \right)$$
$$+ \sum_m v^2_m \left(\sum_t N_t y_{tm} - S_m \right) + \sum_t v^3_t \left(\sum_i \sum_j \sum_m p_{ijtm}x_{ijtm} - \beta_t T_t \right)$$
$$+ \sum_t \sum_m v^4_{tm} \left(\sum_i \sum_j x_{ijtm} - M y_{tm} \right) \qquad (6.2.1)$$

subject to constraints (5.4), (5.6), (5.7) and (5.8). Then the model is further simplified as:

(RLM)

$$Z_D(v) = \text{Min} \sum_i \sum_j \sum_t \sum_m \left(c_{ijtm} + v^1_m p_{ijtm} + v^3_t p_{ijtm} + v^4_{tm} \right) x_{ijtm}$$
$$\sum_t \sum_m \left(v^2_m N_t - v^4_{tm} M \right) y_{tm} - \sum_m v^1_m \alpha_m M_m - \sum_m v^2_m C_m - \sum_t v^3_t \beta_t T_t \qquad (6.2.2)$$

subject to constraints (5.4), (5.6), (5.7) and (5.8) which is designated as relaxed loading model (RLM). This Lagrangian problem (RLM) is defined for all $v \geq 0$. It must be noted that this is a necessary condition for $Z_D(v) \leq Z$ to hold.

The second issue that must be considered in the development of a Lagrangian relaxation based procedure is the solution methodology of the problem with fixed multiplier values as stated previously. In this study, since the Lagrangian problem (RLM) is a perfectly decomposable model, it can be divided into two subproblems (SP1) and (SP2) which are given below, and these subproblems can be solved separately.

(SP1) $$\text{Min} \sum_i \sum_j \sum_t \sum_m \left(c_{ijtm} + v^1_m p_{ijtm} + v^3_t p_{ijtm} + v^4_{tm} \right) x_{ijtm} \qquad (6.2.3)$$

subject to constraints (5.6) and (5.7).

(SP2) $$\text{Min} \sum_{t}\sum_{m}\left(v_m^2 N_t + M v_{tm}^4\right) y_{tm} \qquad (6.2.4)$$

subject to constraints (5.4) and (5.8).

(SP1) can be solved by Procedure RLM1 which is given below.

Procedure RLM1:

Step 1: Calculate $\left(c_{ijtm} + v_m^1 p_{ijtm} + v_t^3 p_{ijtm} + v_{tm}^4\right)$ which is the coefficient of variable x_{ijtm}, for each i,j,t and m.

Step 2: Determine $\min_{t,m}\left(\text{coefficient}_{ijtm}\right)$, for each i and j.

Step 3: Set variable x_{ijtm} with the minimum coefficient to one and the remaining x_{ijtm} variables to zero, for each i and j, so that each job is assigned to only one tool-machine combination.

(SP2) is a knapsack problem, and such knapsack problems can be attempted by dynamic programming. The following algorithm which is called Procedure RLM2 shows how (SP2) is solved by this method.

Procedure RLM2:

Step 1: Since the only technological constraint for (SP2) is the number of tools, decompose (SP2) into individual subproblems for each tool and consider them independently from each other.

Step 2: Check if there are subproblems which have not been solved yet. If there are, start by one of them. Otherwise, go to step 7.

Step 3: Calculate $\left(v_m^2 N_t - M v_{tm}^4\right)$ which is the coefficient of variable y_{tm} for each m.

Step 4: Determine the number of machines compatible with the tool under consideration (q_t).

Step 5: Order y_{tm} variables ascending by m, and rename them as y_{tmj} where $j = 1,2,\ldots, q_t$.

Step 6: Apply dynamic programming method to the subproblem according to the following definitions:

Stage j is represented by variable y_{tmj} where j shows the order of m; j = 1,2,..., q_t.

State w_j at stage j is the total number of machines a certain tool is assigned to in stages j, j+1,..., q_t.; $w_j = 0,1,...,\min(A_t,q_t)$.

Alternative y_{tmj} at stage j indicates whether the tool is assigned to the machine; $y_{tmj} = 0$ or 1.

$f_j(w_j)$: optimal value of stages j, j+1,..., q_t given the state w_j.

Hence, the backward recursive equation is

$$f_{q_t}(w_{q_t}) = \max_{\substack{y_{tmq_t}=0,1 \\ w_{q_t}=0,1,...,\min(A_t,q_t)}} \left(\text{coefficient}_{tmq_t}\right) * y_{tmq_t} \qquad (6.2.5)$$

$$f_j(w_j) = \max_{\substack{y_{tmj}=0,1 \\ w_j=0,1,...,\min(A_t,q_t)}} \left[\text{coefficient}_{tmj} . y_{tmj} + f_{j+1}(w_j - y_{tmj})\right]$$

$$j = 1, 2, ..., (q_t - 1) \qquad (6.2.6)$$

Step 6: Go to Step 2.

Step 7: Stop.

Another consideration in the development of a Lagrangian relaxation based procedure is the design of an appropriate Lagrangian heuristic which will generate a feasible solution out of a near-feasible solution to (LM). In this study, the following Lagrangian heuristic which is referred to as Procedure RLM3 is developed to obtain a feasible solution to the original problem.

Procedure RLM3:

Elimination of unnecessary tool assignments:

Step 1: Check if a tool which is assigned to a certain machine is required by any operation. If no, then cancel that tool assignment to free up slots in the magazine of the machine.

Solving the infeasibility problem of constraint set (5.1):

Step 2: Check whether there exists machines for which the machine-hour availabilities are exceeded. If no, then go to Step 7. Otherwise, continue.

Step 3: If there are machines for which the tool magazine capacities are also exceeded, then take those machines under consideration.

Step 4: Choose the machine with maximum v_m^1.

Step 5: Perform 'Procedure RLM4' to determine the job to be assigned to another machine.

Step 6: Perform 'Procedure RLM5' to determine another tool-machine pair to assign the selected job. Then, go to Step 2.

Solving the infeasibility problem of constraint set (5.3):

Step 7: Check whether there exists tools the lives of which are exceeded. If no, then go to Step 11. Otherwise, continue.

Step 8: Choose the tool with maximum v_t^3.

Step 9: Perform 'Procedure RLM4' to determine the job to be assigned to another machine.

Step 10: Perform 'Procedure RLM5' to determine another tool-machine pair the selected job shall be assigned to. Then, go to Step 7.

Elimination of parts from the batch if necessary:

Step 11: Check if there are still constraints from sets (5.1) and (5.3) which are not satisfied. If yes, then eliminate the part(s) which will create minimum slack on these constraints.

Solving the infeasibility problem of constraint set (5.2):

Step 12: Check whether there exist machines for which the tool magazine capacities are exceeded. If no, then go to Step 19. Otherwise, continue.

Step 13: Choose the machine with maximum v_m^2.

Step 14: Determine the tools which are assigned to the selected machine m.

Step 15: Find the operations which require these tools and which can be performed by other tool-machine combinations.

Step 16: Determine the number of other tool-machine pairs capable of processing each of these operations (num_{ij}).

Step 17: Compute $total_{tm}$ by

$$total_{tm} = \sum_i \sum_j num_{ij} \qquad \forall t \qquad (6.2.7)$$

where m is the selected machine, t stands for the tools allocated to machine m and (i,j) pair indicates the jobs assigned to that tool-machine pair.

Step 18: Choose the tool-machine combination with maximum $total_{tm}$ so that it is easier to assign the jobs requiring the tool-machine pair under consideration to other combinations, and cancel that tool-machine assignment. Then, go to Step 12.

Trying to assign certain tools to machines where necessary:

Step 19: If a job is chosen to be performed by a certain tool-machine combination but if the tool is not allocated to that machine, then assign the tool on the machine if there is enough slot space in the tool magazine of the machine.

Solving the infeasibility problem of constraint set (5.5):

Step 20: Check if all the tools which are required by some operations are assigned to the appropriate machines. If no, then start by a tool-machine pair which causes infeasibility. Otherwise, go to Step 25.

Step 21: Determine the jobs which are chosen to be performed by the tool-machine pair under consideration.

Step 22: Perform 'Procedure RLM5' to determine another tool-machine pair to assign the job.

Step 23: If the job cannot be assigned to any other tool-machine pair, then perform 'Procedure RLM6'.

Step 24: Check if there are still jobs which have to be assigned to another tool-machine pair. If there is, go to Step 22. Otherwise, continue.

Elimination of parts from the batch if necessary:

Step 25: If there are jobs which cannot be reallocated to any tool-machine combination, then eliminate the associated parts from the batch and stop.

Procedure RLM4:

Step 1: Determine the jobs which are allocated to the selected machine (or tool).

Step 2: Since both the costs and the processing times of the jobs must be considered in the job selection among the ones on the selected machine, compute the ratio between the cost and the processing time of each of these jobs as follows:

$$\text{ratio}_{ijtm} = \frac{c_{ijtm}}{p_{ijtm}} \tag{6.2.8}$$

Step 3: Choose the job with $\max\{\text{ratio}_{ijtm}\}$ since the jobs with higher costs and with lower processing times are required to have priority, and cancel the associated job-tool-machine assignment.

Step 4: Check if that tool-machine pair is required by any other job. If no, then also cancel that tool-machine assignment.

Procedure RLM5:

Step 1: Determine possible tool-machine combinations the job can be assigned to. If there is no possible combination, then reallocate the job to the initial tool-machine pair. Otherwise, continue.

Step 2: If there are tools which have already been allocated to the appropriate machines among these possible tool-machine combinations, then give them priority.

Step 3: Compute the product of the cost and the processing time of each of these alternatives by:

$$\text{product}_{ijtm} = c_{ijtm} \cdot p_{ijtm} \tag{6.2.9}$$

Step 4: Choose the alternative with $\min\{\text{product}_{ijtm}\}$ since it is easier to allocate jobs with lower processing times to other machines, and operations with lower costs should be preferred. Then, perform the allocation of the selected job.

Procedure RLM6:

Step 1: Compute the cost of performing jobs with the initial tool-machine pair.

Step 2: Determine the other tools which are allocated to the machine under consideration.

Step 3: Check, for each tool-machine combination determined, if enough space in the magazine of the machine is obtained for the initial tool when that tool-machine assignment is canceled. If no, then do not consider that tool-machine pair.

Step 4: Compute the cost of performing jobs with each of these remaining tool-machine pairs.

Step 5: Find the tool-machine pair which gives maximum cost.

Step 6: If this cost is more than the cost of performing jobs with initial tool-machine pair, then cancel the assignment of tool-machine combination with maximum cost and the allocation of the operations to this tool-machine pair, assign the initial tool to the machine, and continue. Otherwise, cancel all the job allocations on the initial tool-machine pair.

Step 7: Reallocate the jobs to other tool-machine pairs by performing 'Procedure RLM5'.

The overall subgradient procedure in Figure 6.2.1 is based on all these procedures to obtain a near-optimal solution to the original loading problem by making use of (RLM), and it is called Procedure RLM7.

Procedure RLM7:

Step 1: Set initial Lagrange multipliers v^0 to zero, and let $Z_{upper} = A$ and $Z_{lower} = 0$.

Step 2: Check if the maximum number of iterations is reached. If yes, stop. Otherwise, continue.

Step 3: Solve (SP1) in order to determine $\{x_{ijtm}\}$ values.

Step 4: Solve (SP2) in order to determine $\{y_{tm}\}$ values. If $\left(v_m^2 N_t - M v_{tm}^4\right)$ for any y_{tm} is zero, then check x values to see if any operation is assigned to that tool-machine combination. If yes, then set the associated y_{tm} to 1.

Step 5: Compute $Z_D^k(v)$, θ^k and v^{k+1} where

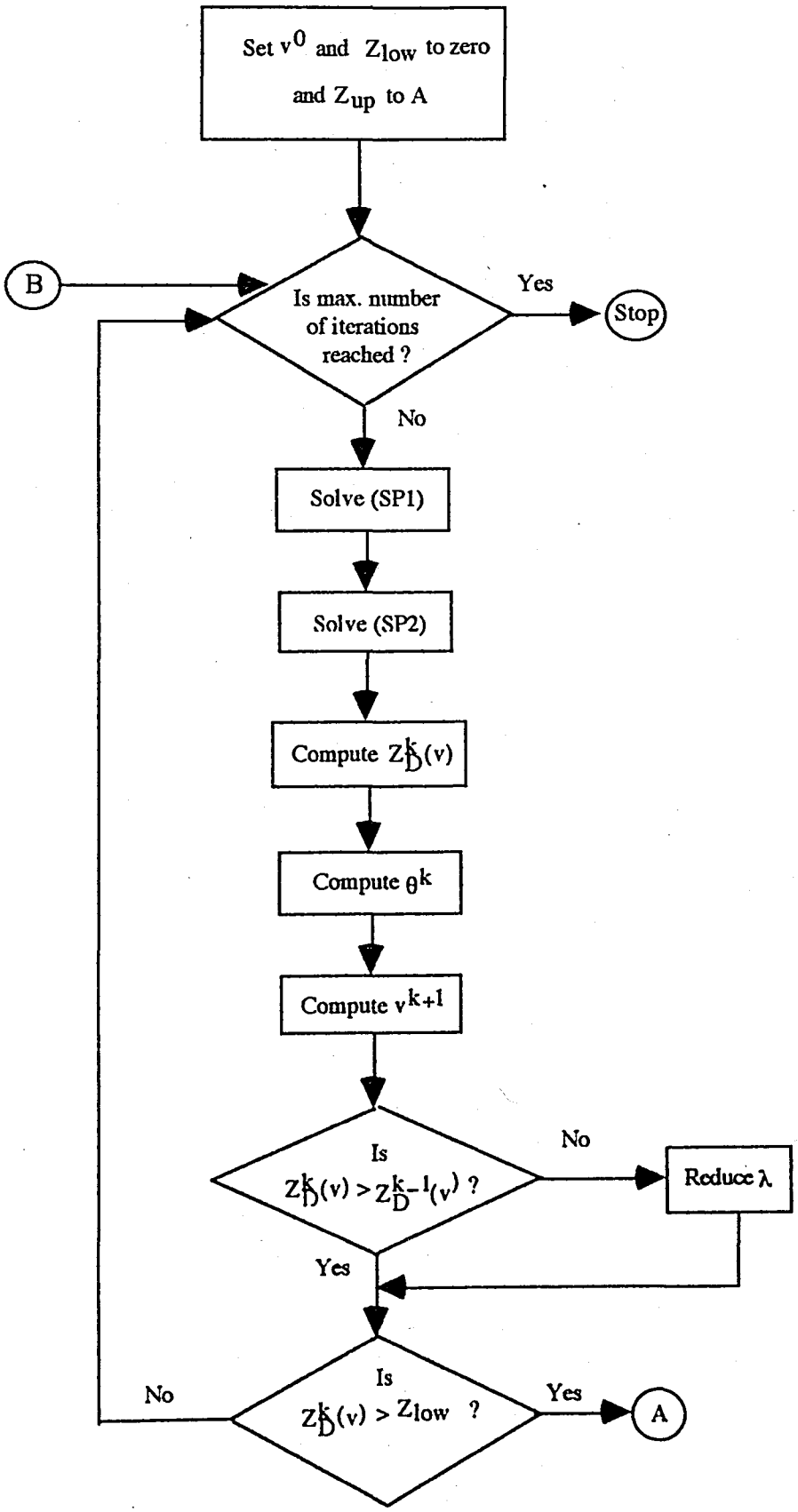

FIGURE 6.2.1. Subgradient optimization procedure to solve (RLM)

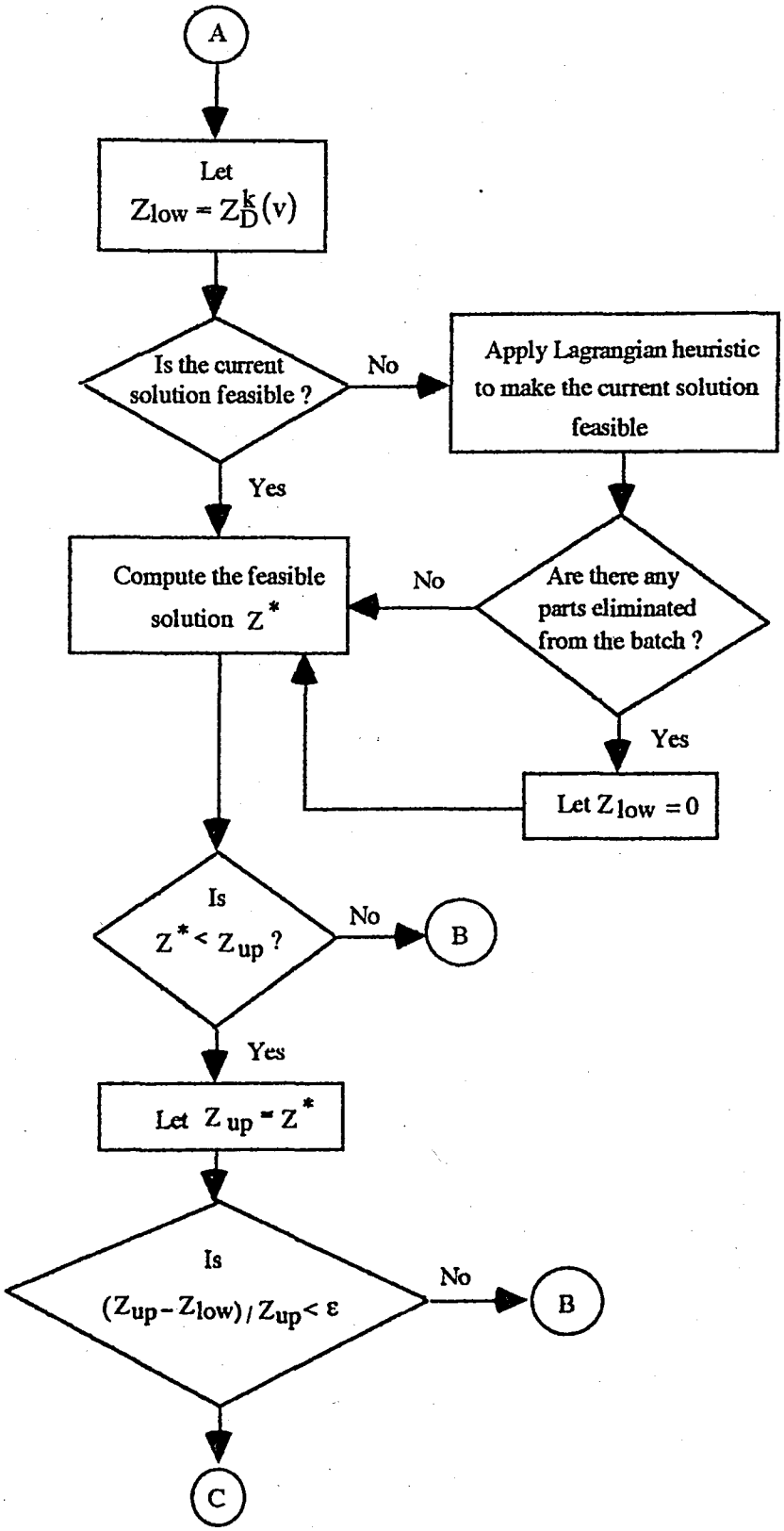

FIGURE 6.2.1. Subgradient optimization procedure to solve (RLM) (continued)

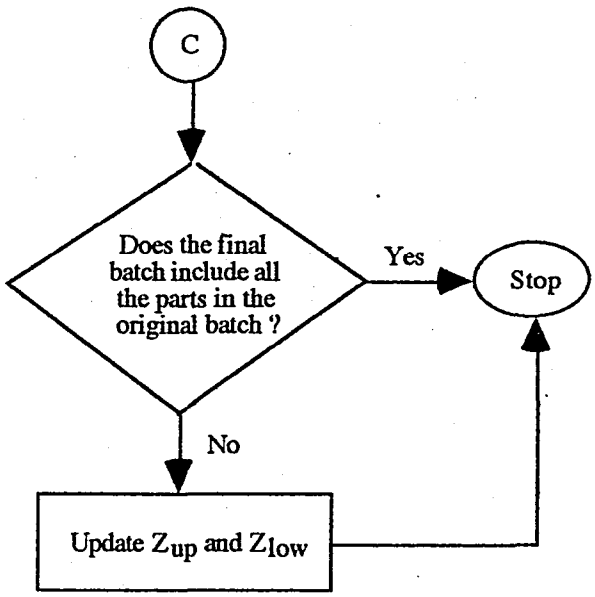

FIGURE 6.2.1. Subgradient optimization procedure to solve (RLM) (continued)

$$\theta^k = \lambda \frac{Z_{upper} - Z_D^k(v)}{\|Ax - b\|} \tag{6.2.10}$$

and v^{k+1} is obtained as in (6.1.5).These formulations can be given in detail as follows:

$$v_m^{1k+1} = v_m^{1k} + \theta_1^k \left(\sum_i \sum_j \sum_t p_{ijtm} x_{ijtm} - \alpha_m M_m \right) \tag{6.2.11}$$

$$v_m^{2k+1} = v_m^{2k} + \theta_2^k \left(\sum_t N_t y_{tm} - C_m \right) \tag{6.2.12}$$

$$v_t^{3k+1} = v_t^{3k} + \theta_3^k \left(\sum_i \sum_j \sum_m p_{ijtm} x_{ijtm} - \beta_t T_t \right) \tag{6.2.13}$$

$$v_{tm}^{4k+1} = v_{tm}^{4k} + \theta_4^k \left(\sum_i \sum_j x_{ijtm} - M y_{tm} \right) \tag{6.2.14}$$

$$\theta_1^k = \lambda \frac{Z_{upper} - Z_D^k(v)}{\left\| \sum_i \sum_j \sum_t p_{ijtm} x_{ijtm} - \alpha_m M_m \right\|} \tag{6.2.15}$$

$$\theta_2^k = \lambda \frac{Z_{upper} - Z_D^k(v)}{\left\| \sum_t N_t y_{tm} - C_m \right\|} \tag{6.2.16}$$

$$\theta_3^k = \lambda \frac{Z_{upper} - Z_D^k(v)}{\left\| \sum_i \sum_j \sum_m p_{ijtm} x_{ijtm} - \beta_t T_t \right\|} \tag{6.2.17}$$

$$\theta_4^k = \lambda \frac{Z_{upper} - Z_D^k(v)}{\left\| \sum_i \sum_j x_{ijtm} - M y_{tm} \right\|} \tag{6.2.18}$$

Step 6: Check if $Z_D^k(v)$ has increased. If no, then reduce λ.

Step 7: Check whether $Z_D^k(v) > Z_{lower}$. If it is, let $Z_{lower} = Z_D^k(v)$. Go to the next step. Otherwise, go to Step 2.

Step 8: Check whether the current solution is feasible to the original problem. If yes, update the best feasible solution (Z^*). Go to Step 11. Otherwise, continue.

Step 9: Make the current solution feasible by applying a 'Lagrangian heuristic' called Procedure RLM3. Let the feasible solution be Z^*.

Step 10: Check if any part is eliminated from the batch by the Lagrangian heuristic. If yes, let $Z_{lower} = 0$.

Step 11: If $Z^* < Z_{upper}$, let $Z_{upper} = Z^*$. Go to Step 12. Otherwise, go to Step 2.

Step 12: If $[(Z_{upper} - Z_{lower}) / Z_{upper}] > \varepsilon$, go to Step 2. Otherwise, continue.

Step 13: Check if the final batch includes all the parts in the original batch. If no, then update Z_{upper} and Z_{lower} by

$$Z_{upper} = Z_{upper} + \sum_i \sum_j \max_{t,m}\left(c_{ijtm}\right) \qquad (6.2.19)$$

$$Z_{lower} = Z_{lower} + \sum_i \sum_j \min_{t,m}\left(c_{ijtm}\right) \qquad (6.2.20)$$

where i stands for the parts eliminated from the original batch.

Step 14: Stop.

There are some important issues that must be paid attention to in these procedures:

(a) It has to be emphasized that (SP2) should be solved after the solution of (SP1) in Procedure RLM7. Although these problems could be considered independent from each other since they are decomposed as stated before and any infeasibility problem related to constraint set (5.5) is solved by the Lagrangian heuristic, first solving (SP1) then (SP2) improves the performance of the Lagrangian heuristic in the search for a feasible solution. When $\left(v_m^2 N_t - M v_{tm}^4\right)$ for any variable y_{tm} is found to be zero in (SP2), then whether y_{tm} is one or zero has no effect on the objective function. In this case, it is reasonable to set that y_{tm} to one if the related tool-machine combination is required by an operation according to the solution of (SP1).

(b) Another point to be emphasized in this procedure is that Z_{lower} is set to zero when a part is eliminated from the batch by the Lagrangian heuristic. The reason is that Z_{upper} is updated depending on the feasible solution Z^*, and Z_{lower} is determined according to the value of $Z_D^k(v)$. So, Z^* and $Z_D^k(v)$ must be compatible and thus calculated for the same batch. However, the elimination of a part from the batch implies that while Z^* is obtained for the reduced problem, Z_{lower} no more belongs to that new batch and must again be set to zero.

(c) After the iterative runs of this procedure, a range for the optimal value of the original problem Z can be determined where the limits are identified by Z_{lower} and Z_{upper}. If the batch which is transformed from the batching level to the loading stage at the beginning of this procedure is not preserved finally, then it must be taken into consideration that the range obtained is for the reduced problem. Hence, these bounds have to be updated to find a range for the original problem as given in (6.2.19) and (6.2.20).

(d) Halving λ whenever $Z_D^k(v)$ fails to increase gives good results, but it is also discovered that dividing λ by a number between 1 and 2 instead of halving it also performs well. In this study, both methods are examined, and it is observed that the alternative method may provide better bounds for some problems.

(e) The algorithm terminates either when a certain degree of precision (ε) is attained or when the maximum number of iterations (N) is reached. Here, ε is assumed to be 0.10 and 0.02 and the maximum number of iterations is taken as 50 for all problems.

This subgradient optimization method which also covers the execution of the Lagrangian heuristic is developed as a computer program written in PROGRESS 4GL, and the program is run for problems with different sizes. The sizes of the problems are determined according to the number of parts, operations and machines. The number of tools is considered to be the same for all problems as 20. The data for each problem are given in Tables A.3.4.1, A.6.2.1, A.6.2.2, A.6.2.3 and A.6.2.4. The results obtained at each iteration of the subgradient procedure for each problem are summarized in Tables 6.2.1, 6.2.2, 6.2.3, 6.2.4. 6.2.5, 6.2.6, 6.2.7, 6.2.8, 6.2.9 and 6.2.10. Tables 6.2.11 and 6.2.12 show how the Lagrangian heuristic makes the results feasible. In each table, the ε value and the procedure applied to λ for that problem are given.

The results of the subgradient optimization procedure show that it is possible to obtain reasonable bounds for the value of Z^* even when $(Z_{upper} - Z_{lower}) / Z_{upper} < \varepsilon$ criterion is not satisfied, and it is observed that the possibility of terminating the algorithm due to $(Z_{upper} - Z_{lower}) / Z_{upper} < \varepsilon$ criterion increases as the size of the problem gets larger. The effect of reducing λ is another issue to be studied in this procedure. As the results are examined, it can be seen that it is not possible to conclude whether halving it or dividing it by a number between one and two is better. Hence, it can be stated that the choice of the reduction method depends on the problem under consideration.

TABLE 6.2.1. Results of the Subgradient Optimization Procedure for the 4-part, 4-operation, 20-tool, 4-machine Problem where λ is divided by 1.2 when $Z_D^k(v)$ fails to increase.

Iteration Number (k)	$Z_D^k(v)$	Z_{lower}	Z^*	Z_{upper}	Step Size (λ)
1	301	301	355	355	2
2	-139	301	-	355	1.67
3	-212	301	-	355	1.39
4	-37	301	-	355	1.39
5	86	301	-	355	1.39
6	119	301	-	355	1.39
7	56	301	-	355	1.16
8	206	301	-	355	1.16
9	208	301	-	355	1.16
10	241	301	-	355	1.16
20	269	301	-	355	0.47
30	260	302	-	350	0.22
40	300	303	-	348	0.11
50	298	304	-	343	0.09
Final Bounds: [304, 343] $\varepsilon = 0.10$ Maximum number of iterations is reached.					

TABLE 6.2.2. Results of the Subgradient Optimization Procedure for the 4-part, 4-operation, 20-tool, 4-machine Problem where λ is halved when $Z_D^k(v)$ fails to increase.

Iteration Number (k)	$Z_D^k(v)$	Z_{lower}	Z^*	Z_{upper}	Step Size (λ)
1	301	301	355	355	2
2	-139	301	-	355	1
3	-212	301	-	355	0.5
4	80	301	-	355	0.5
5	204	301	-	355	0.5
6	225	301	-	355	0.5
7	250	301	-	355	0.5
8	254	301	-	355	0.5
9	261	301	-	355	0.5
10	278	301	-	355	0.5
20	295	301	-	355	0.03125
30	297	301	-	355	0.00391
40	298	301	-	355	0.00048
50	298	301	-	355	0.00003
Final Bounds: [301, 355] $\varepsilon = 0.10$ Maximum number of iterations is reached.					

TABLE 6.2.3. Results of the Subgradient Optimization Procedure for the 8-part, 4-operation, 20-tool, 4-machine Problem where λ is divided by 1.2 when $Z_D^k(v)$ fails to increase.

Iteration Number (k)	$Z_D^k(v)$	Z_{lower}	Z^*	Z_{upper}	Step Size (λ)
1	828	828	953	953	2
2	451	828	-	953	1.67
3	533	828	-	953	1.67
4	536	828	-	953	1.67
5	622	828	-	953	1.67
6	412	828	-	953	1.39
7	608	828	-	953	1.39
8	656	828	-	953	1.39
9	574	828	-	953	1.16
10	626	828	-	953	1.16
20	805	828	-	953	0.56
29	832	0*	815	815	0.22
31	740	725	816	815	0.22
31**	-	809	-	980	-
Final Bounds: [809, 980] $\varepsilon = 0.02$ $(Z_{upper} - Z_{lower}) / Z_{upper} < \varepsilon$ criterion is satisfied.					

* A part is eliminated from the batch.

** Z_{upper} and Z_{lower} are updated according to (6.2.19) and (6.2.20).

TABLE 6.2.4. Results of the Subgradient Optimization Procedure for the 8-part, 4-operation, 20-tool, 4-machine Problem where λ is halved when $Z_D^k(v)$ fails to increase.

Iteration Number (k)	$Z_D^k(v)$	Z_{lower}	Z^*	Z_{upper}	Step Size (λ)
1	828	828	953	953	2
2	451	828	-	953	1
3	533	828	-	953	1
4	744	828	-	953	1
5	582	828	-	953	0.5
6	763	828	-	953	0.5
7	802	828	-	953	0.5
8	769	828	-	953	0.25
9	807	828	-	953	0.25
10	806	828	-	953	0.125
15	830	830	883	883	0.03125
Final Bounds: [830, 883] $\varepsilon = 0.02$ $(Z_{upper} - Z_{lower}) / Z_{upper} < \varepsilon$ criterion is satisfied.					

TABLE 6.2.5. Results of the Subgradient Optimization Procedure for the 15-part, 4-operation, 20-tool, 4-machine Problem where λ is divided by 1.2 when $Z_D^k(v)$ fails to increase.

Iteration Number (k)	$Z_D^k(v)$	Z_{lower}	Z^*	Z_{upper}	Step Size (λ)
1	1551	1551	1694	1694	2
2	1264	1551	-	1694	1.67
3	1227	1551	-	1694	1.39
4	1331	1551	-	1694	1.39
5	1304	1551	-	1694	1.16
6	1277	1551	-	1694	0.96
7	1386	1551	-	1694	0.96
8	1512	1551	-	1694	0.96
9	1481	1551	-	1694	0.80
10	1553	1553	1677	1677	0.80
20	1554	1557	-	1677	0.32
22	1559	1559	1589	1589	0.32

Final Bounds: [1559,1589]
$\varepsilon = 0.02$
$(Z_{upper} - Z_{lower}) / Z_{upper} < \varepsilon$ criterion is satisfied.

TABLE 6.2.6. Results of the Subgradient Optimization Procedure for the 15-part, 4-operation, 20-tool, 4-machine Problem where λ is halved when $Z_D^k(v)$ fails to increase.

Iteration Number (k)	$Z_D^k(v)$	Z_{lower}	Z^*	Z_{upper}	Step Size (λ)
1	1551	1551	1694	1694	2
2	1264	1551	-	1694	1
3	1227	1551	-	1694	0.5
4	1465	1551	-	1694	0.5
5	1514	1551	-	1694	0.5
6	1446	1551	-	1694	0.25
7	1541	1551	-	1694	0.25
8	1565	1565	1592	1592	0.25

Final Bounds: [1565, 1592]
λ is halved when $Z_D^k(v)$ fails to increase.
$\varepsilon = 0.02$
$(Z_{upper} - Z_{lower}) / Z_{upper} < \varepsilon$ criterion is satisfied.

TABLE 6.2.7. Results of the Subgradient Optimization Procedure for the 15-part, 4-operation, 20-tool, 8-machine Problem where λ is halved when $Z_D^k(v)$ fails to increase.

Iteration Number (k)	$Z_D^k(v)$	Z_{lower}	Z^*	Z_{upper}	Step Size (λ)
1	1482	1482	1642	1642	2
2	1548	1548	1681	1642	2
3	1547	1548	-	1642	1.67
4	1516	1548	-	1642	1.39
5	1440	1548	-	1642	1.16
6	1530	1548	-	1642	1.16
7	1504	1548	-	1642	0.96
8	1548	1548	-	1642	0.96
9	1507	1548	-	1642	0.80
10	1554	1554	1690	1642	0.80
17	1559	1559	1588	1588	0.39

Final Bounds: [1559, 1588]
$\varepsilon = 0.02$
$(Z_{upper} - Z_{lower}) / Z_{upper} < \varepsilon$ criterion is satisfied.

TABLE 6.2.8. Results of the Subgradient Optimization Procedure for the 15-part, 4-operation, 20-tool, 8-machine Problem where λ is halved when $Z_D^k(v)$ fails to increase.

Iteration Number (k)	$Z_D^k(v)$	Z_{lower}	Z^*	Z_{upper}	Step Size (λ)
1	1482	1482	1642	1642	2
2	1548	1548	1681	1642	2
3	1547	1548	-	1642	1
4	1516	1548	-	1642	0.5
5	1499	1548	-	1642	0.25
6	1552	1552	1736	1642	0.25
7	1558	1558	1694	1642	0.25
8	1559	1559	1588	1588	0.25
Final Bounds: [1559, 1588] $\varepsilon = 0.02$ $(Z_{upper} - Z_{lower}) / Z_{upper} < \varepsilon$ criterion is satisfied.					

TABLE 6.2.9. Results of the Subgradient Optimization Procedure for the 15-part, 6-operation, 20-tool, 4-machine Problem where λ is divided by 1.2 when $Z_D^k(v)$ fails to increase.

Iteration Number (k)	$Z_D^k(v)$	Z_{lower}	Z^*	Z_{upper}	Step Size (λ)
1	2435	2435	2720	2720	2
2	1613	2435	-	2720	1.67
3	1398	2435	-	2720	1.39
4	2039	2435	-	2720	1.39
5	1859	2435	-	2720	1.16
6	2152	2435	-	2720	1.16
7	2108	2435	-	2720	0.96
8	2207	2435	-	2720	0.96
9	1646	2435	-	2720	0.80
10	2122	2435	-	2720	0.80
20	2490	2590	-	2720	0.56
30	2573	2590	-	2720	0.27
40	2590	2590	2681	2681	0.16
43	2593	2593	2632	2632	0.13

Final Bounds: [304, 343]
$\varepsilon = 0.10$
$(Z_{upper} - Z_{lower}) / Z_{upper} < \varepsilon$ criterion is satisfied.

TABLE 6.2.10. Results of the Subgradient Optimization Procedure for the 15-part, 6-operation, 20-tool, 4-machine Problem where λ is halved when $Z_D^k(v)$ fails to increase.

Iteration Number (k)	$Z_D^k(v)$	Z_{lower}	Z^*	Z_{upper}	Step Size (λ)
1	2435	2435	2720	2720	2
2	1612	2435	-	2720	1
3	1398	2435	-	2720	0.5
4	1932	2435	-	2720	0.5
5	2413	2435	-	2720	0.5
6	2120	2435	-	2720	0.25
7	1727	2435	-	2720	0.125
8	2082	2435	-	2720	0.125
9	2187	2435	-	2720	0.125
10	2112	2435	-	2720	0.0625
20	2463	2463	2855	2720	0.0625
30	2547	2547	2788	2720	0.03125
40	2275	2572	-	2720	0.0078
50	2299	2572	-	2720	0.0039

Final Bounds: [2572, 2720]
$\varepsilon = 0.02$
Maximum number of iterations is reached.

TABLE 6.2.11. Loading Problem Solution by Subgradient Optimization Procedure before the Application of the Lagrangian Heuristic (4-parts, 4-operations, 20-tools, 4-machines)

x values:

i	j	t	m	x_{ijtm}	i	j	t	m	x_{ijtm}
1	1	1	1	0	2	4	12	3	0
1	1	1	2	0	3	1	12	1	1
1	1	2	3	1	3	1	12	3	0
1	2	4	2	0	3	1	15	4	0
1	2	4	3	0	3	2	9	3	1
1	2	7	1	1	3	2	9	4	0
1	2	7	2	0	3	2	18	4	0
1	3	6	2	1	3	3	11	2	1
1	3	6	3	0	3	3	11	4	0
1	4	10	1	1	3	3	19	2	0
1	4	10	2	0	3	4	3	1	1
1	4	13	4	0	3	4	3	2	0
2	1	1	1	1	3	4	14	2	0
2	1	1	2	0	4	1	2	3	1
2	1	3	1	0	4	1	4	2	0
2	1	3	2	0	4	1	4	3	0
2	2	8	3	1	4	2	5	1	0
2	2	16	1	0	4	2	5	4	1
2	3	10	1	0	4	2	20	2	0
2	3	10	2	1	4	3	13	4	1
2	3	17	3	0	4	3	14	2	0
2	4	4	2	1	4	4	7	1	0
2	4	4	3	0	4	4	7	2	1
2	4	12	1	0	4	4	8	3	0

y values:

t	m	y_{tm}	t	m	y_{tm}	t	m	y_{tm}
1	1	1	6	3	0	12	1	1
1	2	0	7	1	1	12	3	0
2	3	1	7	2	1	13	4	1
3	1	1	8	3	1	14	2	0
3	2	0	9	3	1	15	4	0
4	2	1	9	4	0	16	1	0
4	3	0	10	1	1	17	3	0
5	1	0	10	2	1	18	4	0
5	4	1	11	2	1	19	2	0
6	2	1	11	4	0	20	2	0

Slacks of constraint set (5.1):

m	slack
1	-35
2	-82
3	21
4	142

Slacks of constraint set (5.2):

m	slack
1	13
2	22
3	16
4	35

TABLE 6.2.12. Loading Problem Solution by Subgradient Optimization Procedure after the Application of the Lagrangian Heuristic (4-parts, 4-operations, 20-tools, 4-machines)

x values:

i	j	t	m	x_{ijtm}	i	j	t	m	x_{ijtm}
1	1	1	1	0	2	4	12	3	0
1	1	1	2	0	3	1	12	1	1
1	1	2	3	1	3	1	12	3	0
1	2	4	2	0	3	1	15	4	0
1	2	4	3	0	3	2	9	3	0
1	2	7	1	1	3	2	9	4	1
1	2	7	2	0	3	2	18	4	0
1	3	6	2	1	3	3	11	2	0
1	3	6	3	0	3	3	11	4	0
1	4	10	1	0	3	3	19	2	1
1	4	10	2	0	3	4	3	1	1
1	4	13	4	1	3	4	3	2	0
2	1	1	1	1	3	4	14	2	0
2	1	1	2	0	4	1	2	3	1
2	1	3	1	0	4	1	4	2	0
2	1	3	2	0	4	1	4	3	0
2	2	8	3	1	4	2	5	1	0
2	2	16	1	0	4	2	5	4	0
2	3	10	1	0	4	2	20	2	1
2	3	10	2	0	4	3	13	4	1
2	3	17	3	1	4	3	14	2	0
2	4	4	2	1	4	4	7	1	0
2	4	4	3	0	4	4	7	2	1
2	4	12	1	0	4	4	8	3	0

y values:

t	m	y_{tm}	t	m	y_{tm}	t	m	y_{tm}
1	1	1	6	3	0	12	1	1
1	2	0	7	1	1	12	3	0
2	3	1	7	2	1	13	4	1
3	1	1	8	3	1	14	2	0
3	2	0	9	3	0	15	4	0
4	2	1	9	4	1	16	1	0
4	3	0	10	1	0	17	3	1
5	1	0	10	2	0	18	4	0
5	4	0	11	2	0	19	2	1
6	2	1	11	4	0	20	2	1

Slacks of constraint set (5.1):

m	slack
1	49
2	16
3	82
4	40

Slacks of constraint set (5.2):

m	slack
1	24
2	28
3	11
4	23

7. CAPACITY AGGREGATION

The information exchange between the batching and loading levels in the hierarchical approach given in Figure 3.1.1 deserves special attention. As it is indicated before, the machining and tool-magazine capacities are treated in an aggregate manner in the batching level before the loading decisions are made. The capacity aggregation should be done so as to ensure a feasible disaggregation in the loading level.

7.1. Considerations on Hierarchical Interdependencies

In a hierarchically structured system, three different stages of interdependencies may be defined [28].

(a) Anticipation: In order to be able to find a feasible solution, the top-level of the hierarchy considers the relevant characteristics of the base-level, and these characteristics may be called "anticipated base-level". Choosing an anticipated base-level and taking its effects on the top-level into account are referred to as an "anticipation". In other words, anticipation is a bottom-up influence of the base-level on the top-decisions.

(b) Instruction: After anticipating the base-level, the top-level makes a decision which influences the base-level. This decision is called "instruction". So, instruction is a top-down influence of the top-level on the base-level.

(c) Reaction: After the influence of the top-level, the base-level creates a feedback to the top-level, and this reaction is referred to as a "reaction".

In many cases, there are more than one instruction-reaction cycle with anticipation phases prior to each instruction. This instruction-reaction cycle may also be viewed as an interaction cycle which is called a tandem, and the sequence of all tandems constitutes a tandem process. Finally, the top- and base-levels agree upon a decision which results in an implementation influencing the system.

The tandem process can be described as follows: The top-level anticipates the base-level and determines a suitable instruction IN_k^* which is given down to the base-level. The

base-level reacts in sending the signal R_k^* to the top-level after having anticipated a possible future instruction IN_{k+1}^*. The tandem represents one cycle of a communicator and/or negotiation process which finally results in a compromise decision.

7.2. Hierarchical Interdependency of Batching and Loading Levels

The hierarchical integration between the batching and loading levels can be better explained as a special case of the general framework which is described in [28] and which is presented in Section 7.1.

The top-level in this hierarchy is the batching level, and the loading level forms the base-level. Hence, the batching level must take the relevant characteristics of the loading level into account. So, the batching level anticipates the machine capacities $\{\alpha_m M_m\}$ or, shortly, $\{C_m\}$ of the loading level by the aggregate system capacity CAP through an anticipation function $AF^T(C^B)$. Here, $AF^T(C^B)$ represents an anticipation of the base (loading) level, B, at the top (batching) level, T. According to this anticipation, the batching level determines the batch of parts to be produced as a factual instruction IN for the loading level, which translates this information into a loading decision. If this instruction is not feasible, then the loading level transmits a reaction R to the batching level. The anticipation function $AF^T(C^B)$ is updated under the reaction R of the base-level, and a new tandem starts. Hence, it can be seen that the anticipation of the capacities in any cycle is, in fact, assumed to be a function of the loading reactions and can be defined as

$$AF^T(C^B) = AF^T(R(C^B)) \tag{7.2.1}$$

Once the instruction for the loading level sent by the batching level is feasible, then it means that a compromise decision is achieved. So, the instruction obtained by the loading level leaves the process to be executed in the scheduling level as a final instruction. This tandem process can be best explained by Figure 7.2.1. In the process of deriving the instruction for the loading level, the batching level takes into account the decision process of the loading level and especially the most current reaction transmitted by the loading level.

Although the loading level is assumed not to anticipate the decision process of the batching level, it generates its reaction dependent on the batching instructions.

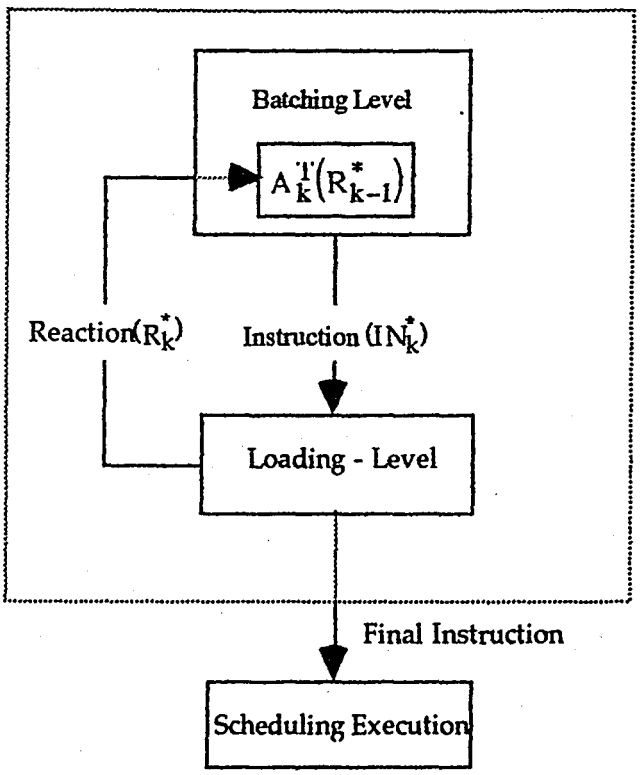

FIGURE 7.2.1. The tandem - process between the batching and loading levels

To describe the hierarchical structure in accordance with [28], let the decision space of the batching model A^T be defined by (4.2) - (4.9) and the decision space of the loading model A^B be (5.1) - (5.8). Of course in any cycle of the tandem A^T is dependent upon the reaction and anticipation function. Thus

$$A_k^T = A_k^T(R_{k-1}^*, AF_k^T) \tag{7.2.2}$$

Similarly A^B is dependent upon the batching instructions, that is,

$$A_k^B = A_k^B(IN_k^*) \tag{7.2.3}$$

Let us define V^T and Z^B as the objective functions of the top level and the base level, respectively, by

$$V^T = \sum_i x_i \tag{7.2.4}$$

$$Z^B = \sum_i \sum_j \sum_t \sum_m c_{ijtm} \cdot x_{ijtm} \tag{7.2.5}$$

The decision of the batching level in any cycle a_k^T is given by $\left\{x_i^*, y_{ij}^*\right\}$ and the decision of the loading level in any cycle a_k^B is defined by $\left\{x_{ijtm}^*, y_{tm}^*\right\}$. Thus the whole process can be summarized by

$$c_k^T(a_k^T) = \max_{a_k^T \in A_k^T(R_{k-1}^*, AF_k^T)} \left\{V_k^T \left[a_k^T \mid R_{k-1}^*, AF_k^T \right]\right\} \tag{7.2.6}$$

$$c_k^B(a_k^B) = \min_{a_k^B \in A_k^B(IN_k^*)} \left\{Z_k^B \left[a_k^B \mid IN_k^* \right]\right\} \tag{7.2.7}$$

Here (7.2.6) implies that the top level determines the optimal batch in cycle k considering the reaction R_{k-1}^* and the anticipation function AF_k^T. The optimal batch is given as an instruction IN_k^* to the base level. On the other hand, (7.2.7) implies that the base level determines an optimal loading schedule over a decision space and an objective function completely defined by IN_k^*. If the loading results are not satisfactory, the base level determines an optimal reaction R_k^*. It is important to note that since the problem is assumed to be deterministic, approximate explicit anticipation is employed without loss of generality by replacing all parameters by constant values. To solve (7.2.6) and (7.2.7) there exist two different types of approaches in literature: Pure Top-Down Hierarchy and Explicit Anticipation Approach. In this study, a heuristic procedure is developed to characterize the anticipation, instruction and reaction functions.

7.2.1. Pure Top-Down Hierarchy

In this approach there is no anticipation function; first the batching model is solved to yield a factual instruction and then the loading model is solved using this instruction. When the loading model cannot produce a feasible solution, this fact is stated to the top level. Knowing that its instruction is not applicable, the batching level would try to generate an acceptable solution with a different aggregate capacity. It is obvious that when a given CAP value does not permit a feasible loading solution, one may attain feasibility by a lower CAP value. A naive top-down approach without any formal anticipation would be to lower CAP iteratively until one obtains a feasible solution accepted by the base level. This may lead to a simple tandem process without any anticipation, but with a simple reaction just indicating the state of feasibility.

In this case, (7.2.6) and (7.2.7) would reduce to

$$c_k^T = \max_{a_k^T \in A_k^T} \left\{ V_k^T \left[a_k^T \right] \right\} \tag{7.2.1.1}$$

$$c_k^B = \min_{a_k^B \in A_k^B(IN_k^*)} \left\{ Z_k^B \left[a_k^B \middle| IN_k^* \right] \right\} \tag{7.2.1.2}$$

In cycle k, A_k^T is determined by CAP_k which is lowered successively according to a particular rule. In this study, a unidimensional search similar to Golden section is employed to generate a sharp and "efficient" CAP value. Thus, an algorithm which is called Procedure PTD is developed and is given below:

Procedure PTD:

Step 1: Determine two capacity (CAP) values, x_1 and x_2, where x_1 is a capacity value such that a feasible loading solution can be obtained, and x_2 stands for a capacity value which leads no feasible solution in the loading level.

Step 2: Calculate

$$\Delta^k = x_2^k - x_1^k \qquad (7.2.1.3)$$

where k indicates the iteration number.

Step 3: Calculate

$$y_1^k = x_1^k + F_1.\Delta^k \qquad (7.2.1.4)$$

$$y_2^k = x_1^k + F_2.\Delta^k = x_2^k - F_1.\Delta^k \qquad (7.2.1.5)$$

where F_1 and F_2 are Fibonacci numbers and 0.38 and 0.62, respectively.

Step 4: Check if the loading problem gives feasible solutions with new capacity values y_1^k and y_2^k.

Step 5: If y_1^k gives a feasible solution, but y_2^k causes infeasibility, then let $x_1^{k+1} = y_1^k$ and $x_2^{k+1} = y_2^k$.

If both y_1^k and y_2^k create infeasibility, then let $x_1^{k+1} = x_1^k$ and $x_2^{k+1} = y_1^k$.

If both y_1^k and y_2^k give a feasible loading solution, then let $x_1^{k+1} = y_2^k$ and $x_2^{k+1} = x_2^k$.

Step 6: If a reasonable bound $[x_1, x_2]$ for the value of the capacity is obtained, then stop. Otherwise, perform another iteration, and go to Step 2.

This procedure is applied to a problem with eight parts, four operations, 20 tools and four machines, and the results are given in Table 7.2.1.1. The results of the same procedure for the 4-part, 4-operation, 20-tool, 4-machine problem are shown in Table 7.2.1.2.

TABLE 7.2.1.1. Results of Procedure PTD for the 8-part, 4-operation, 20-tool, 4-machine Problem

Iteration	x_1^k	x_2^k	Δ^k	y_1^k	Loading Solution	y_2^k	Loading Solution
1	500	1000	500	690	Infeasible	810	Infeasible
2	500	690	190	572.22	Infeasible	617.8	Infeasible
3	500	572.22	72.22	527.44	Infeasible	544.78	Infeasible
4	500	527.44	27.44	510.43	Feasible	517	Feasible
5	517	527.44	10.44	520.97	Feasible	524.1	Feasible
6	524.1	527.44	3.34	525.37	Infeasible	526.17	Infeasible
7	524.1	525.37	1.27	524.58	Infeasible	524.88	Infeasible
8	524.1	524.58	0.48	524.25	Infeasible	524.39	Infeasible
9	524.1	524.25	0.15	524.15	Infeasible	524.19	Infeasible
10	524.1	524.15	0.05	524.12	Infeasible	524.13	Infeasible
11	524.1	524.12	0.02	524.1	Feasible	524.11	Infeasible

TABLE 7.2.1.2. Results of Procedure PTD for the 4-part, 4-operation, 20-tool, 4-machine Problem

Iteration	x_1^k	x_2^k	Δ^k	y_1^k	Loading Solution	y_2^k	Loading Solution
1	750	1000	250	845	Feasible	905	Infeasible
2	845	905	60	867.8	Feasible	882.2	Feasible
3	882.2	905	22.8	890.86	Infeasible	896.34	Infeasible
4	882.2	890.86	8.66	885.49	Feasible	887.57	Feasible
5	887.57	890.86	3.29	888.82	Infeasible	889.61	Infeasible
6	887.57	888.82	1.25	888.05	Feasible	888.35	Feasible
7	888.35	888.82	0.47	888.53	Feasible	888.64	Feasible
8	888.64	888.82	0.18	888.71	Feasible	888.75	Infeasible
9	888.71	888.75	0.04	888.725	Infeasible	888.73	Infeasible

7.2.2. Explicit Anticipation Approach

In this case the anticipation function is defined by

$$AF(a^T) = a^{B^*}(a^T) = \underset{a^B \in A^B(IN(a^T))}{\arg\min} \left\{ Z^B\left[a^B \mid IN(a^T)\right] \right\} \qquad (7.2.2.1)$$

as it is given in [28]. Once (7.2.2.1) is substituted into (7.2.6), one gets a dynamic programming structure with the batching and loading levels as the two stages of the dynamic programming model. To avoid the solution of such a model, the top model should solve (7.2.2.1) for each a^T, which is a proper combination of parts to be included in the batch, and using different $\{C_m\}$ values. After this exhaustive "scenario analysis", the top level should optimize c^T in (7.2.6) with respect to a^T to provide an instruction IN^*.

In addition to these two approaches, a heuristic approach is proposed in this study to characterize the anticipation function.

7.3. Description of the Heuristic Anticipation Process

A heuristic anticipation process is developed in this study as another method of characterizing (7.2.6) and (7.2.7). Let us keep in mind that the anticipation function shows the information content of the top level about the machining capacities. The capacity constraints in both levels are given in Table 7.3.1. Since top level batching decisions are made as an instruction for the loading problem, the batching results should guarantee a feasible solution for the loading decision space. However as it can be seen above, the aggregate constraint is much weaker and may lead to batches for which there exists no feasible loading solution. Thus utmost attention must be given to the capacity aggregation,

and a methodology must be designed to generate a feasible loading solution in an iterative manner even if the initial capacity aggregation does not yield a feasible base level solution.

TABLE 7.3.1. Capacity Constraints in Detailed and Aggregate Levels

Detailed (Base) Level	Aggregate (Top) Level
$\sum_i \sum_j \sum_t p_{ijtm} \cdot x_{ijtm} \le C_m \quad \forall m$ (5.1)	$\sum_i \sum_{j \in O(i)} \hat{p}_{ij} \cdot y_{ij} \le CAP$ (4.5)
$\sum_t \sum_m x_{ijtm} = 1 \quad \forall i, \forall j \in O(i)$ (5.7)	

7.3.1. Aggregation Scheme A

Before any information exchange is exercised between the top and base levels, the anticipation in the batching level should be done according to **Aggregation Scheme A**, which basically clusters the variables related with a particular operation into one variable of the aggregate problem.

Given the detailed variables $\{x_{ijtm}\}$ and the aggregate variables $\{y_{ij}\}$, an aggregation scheme S_1 defined by

$$S_1(x_{ijtm}) \rightarrow y_{ij} \qquad (7.3.1.1)$$

should generate the aggregate columns $\{\hat{p}_{ij}\}$ from the detailed columns $\{p_{ijtm}\}$ according to

$$S_1(p_{ijtm}) \rightarrow \hat{p}_{ij} \qquad (7.3.1.2)$$

In order to achieve this, let $g^{ij}_k \geq 0$ be detailed variable weighting vectors such that $\sum_{k\in(t,m)} g^{ij}_k = 1$ for each (i,j), i = 1,..., N, j $\in$ O(i). Thus this is equivalent to clustering by

$$\sum_i \sum_j \sum_t \sum_m g^{ij}_{(t,m)} \cdot p_{ijtm} \cdot x_{ijtm} \Rightarrow \sum_i \sum_j \hat{p}_{ij} \cdot y_{ij} \qquad (7.3.1.3)$$

and accordingly

$$\hat{p}_{ij} = \sum_t \sum_m g^{ij}_{(t,m)} \cdot p_{ijtm} \qquad \forall i,j \in (O(i)) \quad (7.3.1.4)$$

where $g^{ij}_{(t,m)}$ can be arbitrarily satisfying $\sum_{(t,m)} g^{ij}_{(t,m)} = 1$. Because of (5.6), it is sufficient to define CAP in (4.5) by

$$CAP = \sum_m C_m \qquad (7.3.1.5)$$

The tool-slot capacities of the machines are also reflected upon the batching level in an aggregate manner by

$$SLT = \sum_m S_m \qquad (7.3.1.6)$$

The performance of **Aggregation Scheme A** is analyzed for all test problems by varying the capacities and processing times. It is observed that it performs fairly well under mild conditions: the loading model can find a feasible solution in the first cycle when the processing times are fairly close to each other and / or machine capacities are close to each other. However as capacities and processing times diverge, respectively, the performance declines, and the probability of infeasibility increases.

The 8-part, 4-operation, 4-machine test problem results will be cited as an example. The detailed processing times $\{p_{ijtm}\}$ are given in Table 7.3.1.1 while the aggregate processing times $\{\hat{p}_{ij}\}$ under **Aggregation Scheme A** are given in Table 7.3.1.2.

As it can be seen from Table 7.3.1.1, the processing times on M1 and M3 differ considerably than those on M2 and M4. Taking the aggregate capacity CAP in the batching level as 800 min and varying the individual machine capacities in the loading level, the results given in Table 7.3.1.3 are obtained under **Aggregation Scheme A**. A close look into the results relieves that as the difference between machine capacities increases, it is possible to have infeasible solutions; thus the next step is to design an anticipation procedure when infeasibility is encountered.

When the loading problem turns out to be infeasible for the optimal batch B^* obtained in the batching level, first it is necessary to relax the loading model by dropping integrality constraints (5.7) and (5.8) and replacing them by

$$0 \le x_{ijtm} \le 1 \qquad (7.3.1.7)$$

$$0 \le y_{tm} \le 1 \qquad (7.3.1.8)$$

respectively. If the relaxed loading turns out to be infeasible, an auxiliary operation selection (AOS) problem is solved to determine the maximum number of operations among the operations of parts included in B^* utilizing the total machine capability. Thus, the following model is formulated:

(AOS)

Maximize $$B = \sum_{j} \sum_{i \in B^*} y_{ij} \qquad (7.3.1.9)$$

subject to $$\sum_{i \in B^*} \sum_{j} \sum_{t} p_{ijtm} \cdot x_{ijtm} \le C_m \qquad \forall m \qquad (7.3.1.10)$$

TABLE 7.3.1.1. Detailed Processing Times $\{p_{ijtm}\}$

Part number		1												
Operation number		1			2				3			4		
Compatible tool set		1	3	9	7	8	11	20	2	5	6	1	16	19
	Mach.													
Processing	1	63	54		51					54		126	123	
time on	2	37	35		15		38	40			31	38		22
each	3			54		66			39		58			
machine	4			25			30			16				
Part number		2												
Operation number		1			2				3			4		
Compatible tool set		9	15	1	7	20	3	4	13	4	7	12		
	Mach.													
Processing	1			57	102		39				18	45		
time on	2			27	39	20	10	12		7	9			
each	3		56					22		22		26		
machine	4		34	39					18					
Part number		3												
Operation number		1			2			3			4			
Compatible tool set		12	17	18	1	9	5	6	5	14				
	Mach.													
Processing	1	15			33		51		54					
time on	2				25			23		23				
each	3	16	18			48		28						
machine	4			12		20	22		21					
Part number		4												
Operation number		1		2		3			4					
Compatible tool set		7	15	3	5	6	11	14	18					
	Mach.													
Processing	1	66		96	57									
time on	2	25		31		25	19	28						
each	3					54								
machine	4		28		28		23		31					

TABLE 7.3.1.1. Detailed Processing Times $\{p_{ijtm}\}$ (continued)

Part number						5						
Operation number			1			2			3		4	
Compatible tool set			18	19	4	7	20	1	3	5		
	Mach.											
Processing	1							39	45	36		
time on	2			17	10	5	11	12	18			
each	3				18	16						
machine	4		15							9		
Part number						6						
Operation number		1			2			3			4	
Compatible tool set		11	14	5	9	18	6	8	3	7	8	
	Mach.											
Processing	1			123					57	84		
time on	2	18	21				28		18	25		
each	3				74		62	74			40	
machine	4	19		39	35	30						
Part number							7					
Operation number			1			2			3		4	
Compatible tool set		8	12	14	5	8	1	2	9	10	14	
	Mach.											
Processing	1		51		63		36			9		
time on	2			19			11			4	6	
each	3	30	40			40		10	14			
machine	4				24				6			
Part number							8					
Operation number			1			2			3		4	
Compatible tool set	2	4	8	9	13	15	6	7	18	4	6	7
	Mach.											
Processing	1							57				96
time on	2		23				15	25		23	29	30
each	3	38	40	50	56		34			48	62	
machine	4				26	25	27		21			

TABLE 7.3.1.2 Aggregate processing times $\{\hat{p}_{ij}\}$

$\hat{p}_{ij}$								
i / j	1	2	3	4	5	6	7	8
1	44.67	43	15.25	39.67	16	19.33	35	37.75
2	40	49	31.5	63.5	12	60.2	42.33	33.5
3	39.4	20.2	31	34.33	28.5	54.67	15.4	30.4
4	77.25	21.17	32.67	29.5	22.5	44.8	6.33	43.5

TABLE 7.3.1.3. Loading level results

	Loading Capacities				Loading Result
	M1	M2	M3	M4	
RUN 1	200 min	200 min	200 min	200 min	Feasible
RUN 2	75 min	175 min	75 min	475 min	Feasible
RUN 3	50 min	150 min	50 min	550 min	Feasible
RUN 4	75 min	100 min	75 min	550 min	Feasible
RUN 5	150 min	75 min	75 min	500 min	Feasible
RUN 6	300 min	100 min	300 min	100 min	Feasible
RUN 7	400 min	100 min	200 min	100 min	Feasible
RUN 8	350 min	50 min	350 min	50 min	Infeasible
RUN 9	400 min	30 min	300 min	120 min	Infeasible
RUN 10	450 min	50 min	250 min	50 min	Infeasible

$$y_{ij} \le M. \sum_{t} \sum_{m} x_{ijtm} \qquad \forall i \in B^*, j \quad (7.3.1.11)$$

$$x_{ijtm} = (0,1) \qquad \forall i \in B^*, j, t, m \quad (7.3.1.12)$$

$$y_{ij} = (0,1) \qquad \forall i \in B^*, j \quad (7.3.1.13)$$

Since (AOS) is a relaxed version of (M1), it is always possible to find a feasible solution. Then the total processing time obtained in the (AOS) solution provides an upper bound about a feasible operation-machine capacity configuration. Then the next cycle of the tandem starts solving the batching model under **Aggregation Scheme A** with CAP equal to the total processing time obtained in the optimal (AOS) solution.

If the relaxed loading on the other hand turns out to be feasible, the dual-variables associated with capacity constraints convey valuable information about capacity aggregation and a surrogate constraint should be developed by using the dual variables of the relaxed loading model. This surrogate constraint will be included in the batching model of the next tandem cycle.

Let us consider the following model (SM):

(SM)

$$\text{Max } Z = \sum_j c_j . x_j$$

$$\text{subject to} \quad \sum_j a_{ij} . x_j \le s_i \qquad \forall i \qquad (7.3.1.14)$$

$$x_j \in \{0,1\}$$

In this model (SM), the following constraint

$$\sum_j a_{ij} . x_j \le s_i \qquad \forall i \qquad (7.3.1.15)$$

can be replaced by the surrogate constraints

$$\sum_i \sum_j u_i . a_{ij} . x_j \le \sum_i u_i . s_i \qquad \forall i \qquad (7.3.1.16)$$

where $u \geq 0$, and S and S(u) are the set of binary solutions to (7.3.1.15) and (7.3.1.16), and since $S(u) \supseteq S$, (7.3.1.16) is weaker than (7.3.1.15).

A good surrogate should have the property of eliminating as many nonoptimal completions as possible. Let

$$f(u) = \max \sum_j c_j . x_j \qquad x \in S(u) \qquad (7.3.1.17)$$

A way to define a strongest surrogate constraint is to say that u generates a stronger constraint than u' if $f(u) < f(u')$. In other words, if the best solution in S(u) is worse than the best in S(u'), it is likely that there are more nonoptimal solutions in S(u') than in S(u). Hence, the best surrogate constraint would be generated by u^*, where

$$f(u^*) = \min_{u \geq 0} f(u) \qquad (7.3.1.18)$$

Since it is hard to solve (7.3.1.18), a reasonable estimate of u^* can be calculated by dropping the integrality restrictions in S(u) as

$$f'(u) = \max \sum_j c_j . x_j$$

subject to

$$\sum_i \sum_j u_i . a_{ij} . x_j \leq \sum_i u_i . s_i \qquad (7.3.1.19)$$

$$0 \leq x_j \leq 1$$

and generating the best surrogate constraint by u^0, where

$$f'(u^0) = \min_{u \geq 0} f'(u) \qquad (7.3.1.20)$$

and u^0 stands for the dual variables corresponding to the constraint set (7.3.1.15).

7.3.2. Aggregation Scheme B

The inclusion of dual variables in the development of the surrogate constraint requires additional refinement in the processing time aggregation. Thus, the **Aggregation Scheme B** is developed to serve this purpose.

Given the detailed variables $\{x_{ijtm}\}$ and aggregate variables $\{\hat{x}_{ijm}\}$, an aggregation scheme S_2 defined by

$$S_2\{x_{ijtm}\} \rightarrow \hat{x}_{ijm} \tag{7.3.2.1}$$

should generate the aggregate columns $\left\{\hat{pp}_{ijm}\right\}$ from the detailed columns $\{pp_{ijtm}\}$ according to

$$S_2\{p_{ijtm}\} \rightarrow \hat{pp}_{ijm} \tag{7.3.2.2}$$

In order to achieve this, let $h_k^{ijm} \geq 0$ be arbitrary detailed variable weighting vectors such that $\sum_t h_t^{ijm} = 1$ for each $(i,j), i = 1,\ldots,N, j \in O(i), m = 1,\ldots,M$. Thus this is equivalent to clustering by

$$\sum_i \sum_j \sum_t h_t^{ijm} \cdot p_{ijtm} \cdot x_{ijtm} \Rightarrow \sum_i \sum_j \hat{pp}_{ijm} \cdot \hat{x}_{ijm} \quad \forall m \tag{7.3.2.3}$$

and accordingly

$$\hat{pp}_{ijm} = \sum_t h_t^{ijm} . p_{ijtm} \qquad \forall i,j \in O(i), m \qquad (7.3.2.4)$$

Thus, the capacity constraint (5.1) turns out to be

$$\sum_i \sum_j \hat{pp}_{ijm} . \hat{x}_{ijm} \leq C_m \qquad \forall m \qquad (7.3.2.5)$$

Let u_m be the dual variable associated with the capacity constraint of machine m. Then the aggregate capacity constraint to be included into the batching model is defined by

$$\sum_m u_m \sum_i \sum_j \hat{pp}_{ijm} . \hat{x}_{ijm} \leq \sum_m u_m . C_m \qquad (7.3.2.6)$$

Figure 7.3.2.1 summarizes the overall process of developing a capacity anticipation function in an iterative manner in the direction of improving loading feasibility. Furthermore the results of the proposed anticipation process for the 8-part, 4-operation, 4-machine problem in RUN8 are given in Table 7.3.2.1 and for the 4-part, 4-operation, 4-machine problem in Table 7.3.2.2.

When the results obtained by Pure Top-Down Hierarchy and Heuristic Anticipation Process are examined, it is not possible to state which one of the approaches performs better than the other, and it can be concluded that the performance of these two procedures depend on the problem setting they are applied. For example, the solution of the problem with four parts, four operations, 20 tools and four machines by Pure Top-Down Hierarchy creates less slack processing time (240 hours) in the system when compared to the slack obtained by Heuristic Anticipation Process (442 hours). However, it is observed that Heuristic Anticipation Process performs better than the Pure Top-Down Hierarchy for the 8-part, 4-operation, 20-tool, 4-machine problem in terms of the slack processing time in the system because Pure Top-Down Hierarchy gives a slack of 320 hours for this problem while the other method creates 215 hours slack processing time.

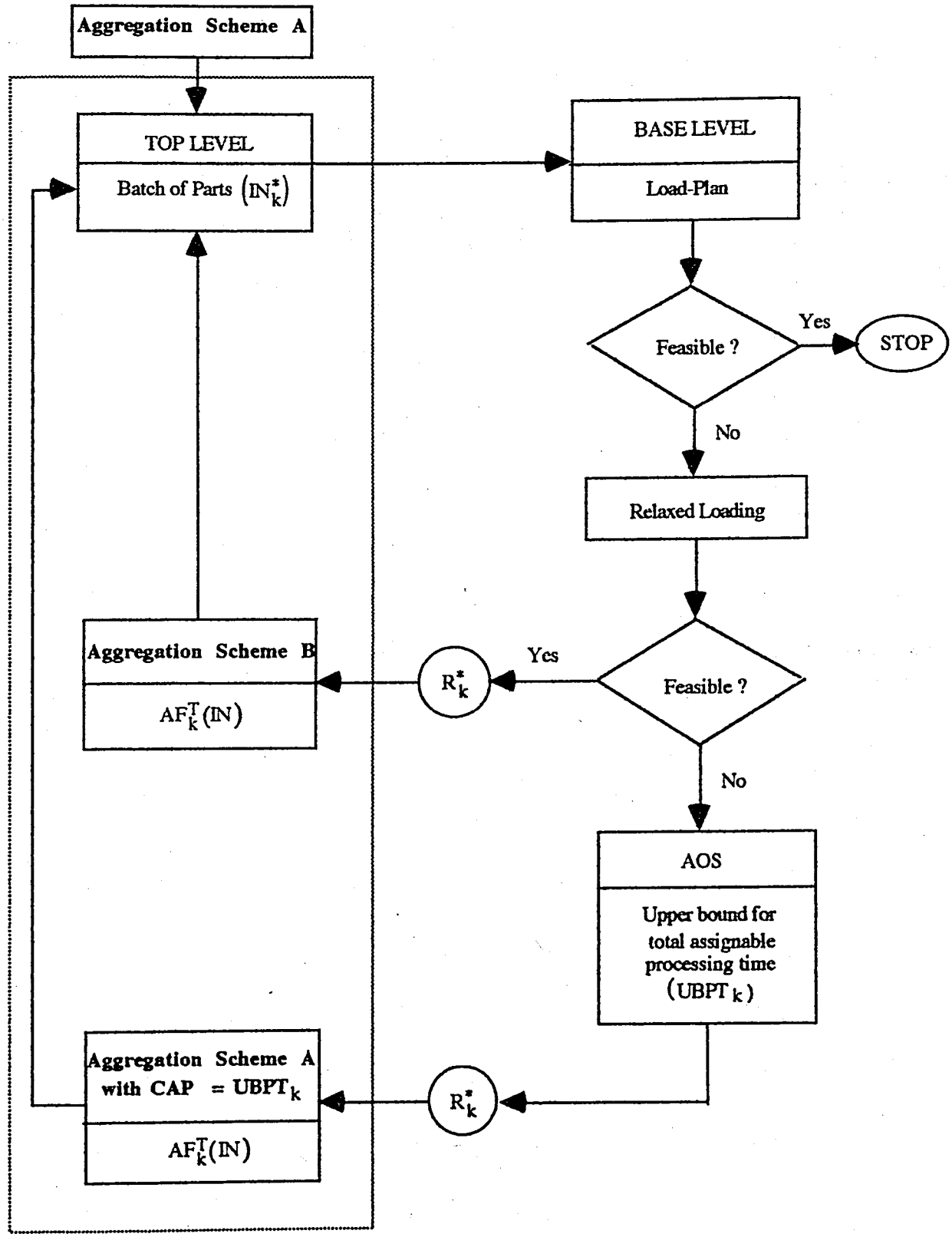

FIGURE 7.3.2.1. The flow-chart of the anticipation process

TABLE 7.3.2.1. Results of the anticipation process for the 8-part, 4-operation. 20-tool, 4 -machine problem

k=1	**Batching Problem**	
	System Capacity = 1000 min.	$x_1, x_2, x_3, x_4, x_5, x_6, x_7$ and x_8 are included in the batch.
	Loading Problem	
	Objective: Cost Minimization 0 - 1 Integer Problem	No feasible solution
	Relaxed Loading Problem	
	Objective: Cost Minimization $0 \le x_{ijtm} \le 1, 0 \le y_{tm} \le 1$	No feasible solution
	Auxiliary Operation Selection Problem(AOS)	
	Feasible solution	Total processing time = 794 min.
k=2	**Batching Problem**	
	System Capacity = 794 min.	x_3, x_4, x_5, x_6, x_7 and x_8 are included in the batch.
	Loading Problem	
	Objective: Cost Minimization 0 - 1 Integer Problem	No feasible solution
	Relaxed Loading Problem	
	Objective: Cost Minimization $0 \le x_{ijtm} \le 1, 0 \le y_{tm} \le 1$	No feasible solution
	Auxiliary Operation Selection Problem(AOS)	
	Feasible solution	Total processing time = 769 min.
k=3	**Batching Problem**	
	System Capacity = 769 min.	x_2, x_3, x_4, x_5, x_7 and x_8 are included in the batch.
	Loading Problem	
	Objective: Cost Minimization 0 - 1 Integer Problem	No feasible solution
	Relaxed Loading Problem	
	Objective: Cost Minimization $0 \le x_{ijtm} \le 1, 0 \le y_{tm} \le 1$	Feasible solution Machine — Dual Prices 1 — 0.956158 2 — 3.057781 3 — 1.027797 4 — 2.503802
k=4	**Batching Problem**	
	Capacity of the Surrogate Constraint = 972.46 System Capacity = 769	x_1, x_5, x_6 and x_7 are included in the batch.
	Loading Problem	
	Objective: Cost Minimization 0 - 1 Integer Problem	Feasible solution

TABLE 7.3.2.2. Results of the anticipation process for the 4-part, 4-operation. 20-tool, 4 -machine problem

k=1	**Batching Problem**	
	System Capacity = 1000 min.	x_2 and x_4 are included in the batch.
	Loading Problem	
	Objective: Cost Minimization 0 - 1 Integer Problem	No feasible solution
	Relaxed Loading Problem	
	Objective: Cost Minimization $0 \le x_{ijtm} \le 1, 0 \le y_{tm} \le 1$	No feasible solution
	Auxiliary Operation Selection Problem(AOS)	
	Feasible solution	Total processing time = 622 min.
k=2	**Batching Problem**	
	System Capacity = 622 min.	x_2 is included in the batch.
	Loading Problem	
	Objective: Cost Minimization 0 - 1 Integer Problem	Feasible solution

8. SCHEDULING LEVEL

Once a machine loading and tool allocation scheme is obtained, it becomes necessary to schedule the parts on the shop-floor, which is the goal of the third level. An FMS scheduling problem is considered to be a detailed minute-by-minute scheduling of the machines, material-handling system, and other support equipment [15]. Given the actual shop conditions and a set of parts determined in the batching level with known processing requirements obtained according to the loading level decisions, scheduling deals with performing the following tasks:

(a) Schedule actual release times of the jobs to the shop-floor.

(b) Sequence the jobs and determine the start and completion times of each operation on the required machine.

(c) Assign free AGVs to the transportation of parts from the central buffer or from the output buffers of the machines to the input buffers of the machines.

(d) Monitor the execution of the schedule and provide effective contingency handling.

For a dynamic and highly integrated system such as an FMS, the scheduling of machines and of the materials-handling system in real time and considerations of limited input/output buffer capacities are of particular importance.

From a system point of view, an FMS consists of two interrelated subsystems as a machining subsystem and a materials-handling subsystem. The machining subsystem can be defined as a job-shop where parts with different processing steps and routing plans are processed. The material handling system, on the other hand, is generally an AGV system where mobile vehicles move the parts from one machine to another along a pre-defined path.

These subsystems affect the performance of each other because any task performed in one of these systems create a 'driving force' for the other. For example, a job completion at a machining center generates a requirement for an AGV while completion of a transportation service by an AGV determines job potentials for machining centers. In an FMS, these subsystems must be concurrently treated since none of these is dominant on the other. At one time, the machining subsystem can be a 'constrained resource' where at some other time, the AGV system becomes a 'critical resource'. Due to this interaction, both subsystems must be considered simultaneously in scheduling an FMS.

Apart from these two subsystems, the input and output queue capacities at workstations must also be taken into account. Because of the limited capacities of these queues, there is always a possibility that a machine can be blocked or the system can be locked. In this study, blocking is defined as the state of a machine when it cannot move the currently processed part to the output buffer. Locking of the system occurs when no further action can be taken, so that system capacities should be modified to resolve it.

The loading models discussed in this study all attempt to assign each operation to a single machine-tool pair as given by (5.6); thus the route along which a part can flow through the system is determined uniquely, and there is no need for a routing decision. Therefore the output from the loading stage is the necessary machine-tool assignments for the operations of a given batch of parts, and the next stage is concerned with scheduling the operations to the machines in the two subsystems described above.

The parts waiting in the input buffer of a machine are released to be processed on the machine according to a dispatching policy and taken to the output buffer of the machine when the processing terminates. The part should wait in the output buffer queue until an AGV becomes free and selects it to transport to its next operation. Thus, in a general sense it is necessary to determine

(a) a part dispatching rule which will choose a part in the input buffer queue to start processing on the machine;

(b) an AGV dispatching procedure which will assign a free AGV to the transportation of a particular part from the output buffer to the input buffer of the machine in which the next operation will commence.

The purpose of the scheduling stage is to analyze the effect of different dispatching rules on the performance measures of an FMS environment by varying model parameters of the hypothetical system described in Section 3. To make reliable conclusions, the system utilization or tightness is designed to be high in all problems where the AGV is always the bottleneck resource.

8.1 Scheduling Assumptions

To simplify the scheduling model, the following assumptions are made:

(a) a given set of parts with equal release times is considered in a static problem setup, and on-line scheduling to schedule all operations of already available jobs for the entire scheduling period is employed;

(b) the loading and tool allocation decisions have been made to provide a single route for each part;

(c) the tools required to process the parts under consideration during the scheduling period have already been installed upon the tool magazines, and there is no setup during the scheduling period;

(d) machine and AGV breakdown is not included;

(e) each machine can process at most one part at a time, and an AGV can transfer at most one part at a time;

(f) pre-emption is not permitted;

(g) traffic congestion, thus the blocking of AGVs, is not included;

(h) the AGV travel time is directly proportional to the distance traveled, and without loss of generality traveling times are computed on the single-loop unidirectional distances;

(i) part load / unload time to an AGV is known and included in the computation of travel times,

(j) pallet availability is not considered, and an infinite number of pallets is assumed.

8.2 The Scheduling Algorithm

A scheduling procedure is developed in this paper to assign the parts to the AGVs for a transfer between two consecutive operations and to release parts to be processed on the machines. In the process of scheduling, the first action has to be releasing jobs to the shop-floor from the central buffer until all the AGVs in the system become busy by transportation

of these jobs for their first operations. The time point that this action is taken is referred to as t = 0. A decision should be made whenever

(a) an AGV completes its current transfer and becomes ready for the next transfer;

(b) a machine completes processing a part and becomes ready for processing some other part.

The situation when a machine completes processing a part does not create an inter-machine activity. The part movement takes place within the machine center. If the next operation of the part is to be processed on another machine, then it is taken to the output buffer of the machine depending on the number of parts waiting in the output buffer and the buffer capacity. If the part can be moved to the output buffer, then a part from the input buffer is selected to be processed on that machine. An inter-machine part movement may occur when an AGV completes its current transfer and becomes free. In dispatching an AGV, push and pull-based rules can be implemented simultaneously according to a hierarchical execution similar to Sabuncuoğlu and Hommertzheim [16] . As it is well-known, in the push procedure an idle AGV selects a part among the set of jobs waiting in the output buffers of the machines to transfer to the next operation; however in the pull-procedure first a machine with the highest priority for part replenishment is selected, and then a part is selected among the set of jobs which can move to the selected machine as described in Yim and Linn [25]. Whenever an AGV becomes available for its next transfer, a hierarchical search is executed according to the flow-chart given in Figure 8.2.1.

In this hierarchical approach, the blocked machines have the highest priority since blocking may later cause locking of the system. In order to remove blocking, the output buffers must be emptied by transferring the parts in these output buffers to their destination machines. Hence, since the selection of a part by the free AGV is under consideration, push logic applies at this point of the hierarchy. The next level of the hierarchy deals with the selection of parts from the central buffer again with the push logic. In the third level, since first the destination machine is selected, it can be stated that the actions taken at this stage are the results of the pull-procedure. In the lowest level of the hierarchy, a kind of look-ahead rule is applied, and the parts that have not finished being processed or have not even started being processed are also taken to be candidates for transportation. This way time loss may be prevented by allowing the transportation of an AGV and the processing of the part which will be transferred to its next operation by that AGV take place simultaneously. Hence, in this level both the AGV and the part can initiate transportation, so that the procedures at this stage may be performed according to either push or pull logic.

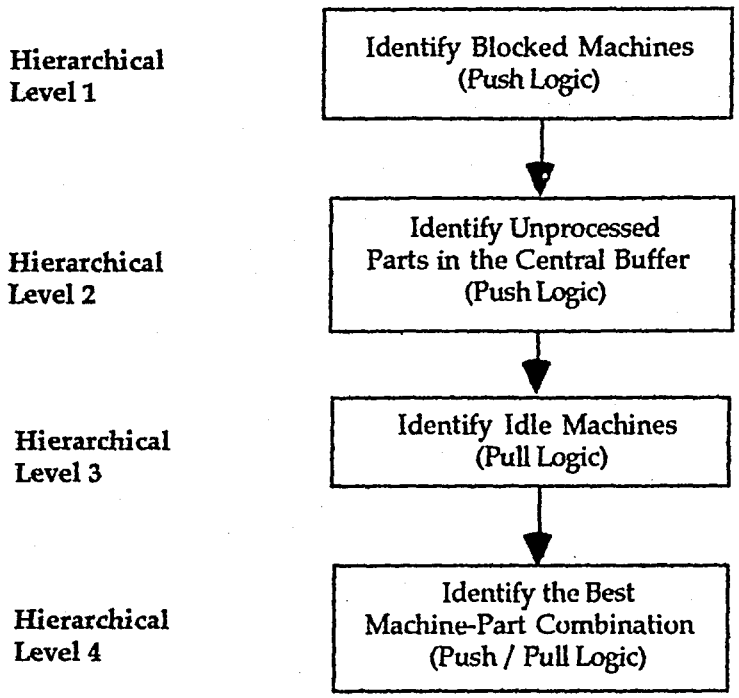

Figure 8.2.1. Hierarchical structure of the scheduling algorithm

The details of the scheduling algorithm are given below in an iterative manner.

The Scheduling Algorithm:

Initial Release of the Jobs from the Central Buffer:

Step 1: If t = 0, then continue. Otherwise, go to Step 8.

Step 2: Check if there is any job in the central buffer. If yes, then continue. Otherwise, go to Step 28.

Step 3: Check if there is a free AGV in the system. If there is, then continue. Otherwise, go to Step 28.

Step 4: Choose an AGV randomly.

Step 5: Select one of the jobs in the central buffer according to rule 1 (**R1**).

Step 6: If the destination machine of the selected job is free, then move the job onto the machine. Otherwise, move it to the input buffer of the machine if there is free space in the corresponding input buffer.

Step 7: Go to Step 1.

Dispatching of Parts on the Machines:

Step 8: If t is equal to the finishing time of some jobs, then check, for each finishing job, if its next operation is assigned to the same machine. Start processing the next operations for the parts which satisfy this condition. For each of the other jobs, check if there is space in the output buffer of the current machine. If there are jobs satisfying this condition, then select one of them randomly. Move the selected job to the output buffer.

Step 9: Check the input buffer of the machine from which the selected job is moved to see if there are any jobs waiting to be processed. If yes, then choose one of them according to rule 2 (**R2**). Move the job selected onto the machine.

Step 10: If t is equal to the arrival time of any AGV, then continue. Otherwise, go to Step 27.

Identification of Blocked Machines:

Step 11: Check if there are any machines blocked in the system. Consider only the ones which have at least one job for which the destination machine has available space in its input buffer. If such machine(s) exist(s), then continue. Otherwise, go to Step 15.

Step 12: If there exist more than one blocked machine which satisfy this condition, then select one according to rule 3 (**R3**).

Step 13: Choose one of the jobs waiting in the output buffer of the selected machine according to rule 4 (**R4**).

Step 14: To move the job chosen to its destination machine so that the machine is no more blocked, assign the AGV to the blocked machine. If the arrival time of the AGV is equal to time t, then go to Step 8. Otherwise, go to Step 28.

Identification of Unprocessed Jobs in the Central Buffer:

Step 15: Check the central buffer to see if there are any jobs waiting to be transferred and the destination machines of which are available. If there is, then continue. Otherwise, go to Step 18.

Step 16: Choose any job from the central buffer according to rule 1 (**R1**).

Step 17: Assign the AGV to move the selected job to its destination machine. Go to Step 28.

Identification of Idle Machines:

Step 18: Check if there is any idle machine in the system. If yes, then continue. Otherwise, go to Step 21.

Step 19: Select a job which can proceed to one of the idle machines according to rule 5 (**R5**). If there is no such job, then go to Step 21. Otherwise, continue.

Step 20: Assign the AGV to move the selected job to the idle destination machine. Go to Step 28.

Identification of AGV / Part Priorities:

Step 21: Determine the arrival time of each AGV to each machine.

Step 22: Find max(AGV Arrival Time, Operation Completion Time) for each machine and for each job being processed on the machines or in the output buffers of the machines, so that the earliest starting time of transportation (ESTT) for each AGV-job combination is determined.

Step 23: Eliminate the jobs for which the input buffers of the destination machines are not available.

Step 24: Choose the job among the remaining ones with min (ESTT).

Step 25: If more than one job satisfy this condition, then select the job according to rule 6 (**R6**).

Step 26: If the arrival time of the AGV is equal to time t, then go to Step 8. Otherwise, go to Step 28.

Step 27: Check the system to see if there are still operations to be performed in the system. If yes, then go to Step 21. Otherwise, continue.

Step 28: t = t + 1.

Step 29: Check if t is equal to the finishing time of a job which has already been assigned to an AGV at Step 24 before it has finished being processed. If yes, then move it to the output buffer in order to make it ready for transportation.

Step 30: Check if there are any jobs waiting in the input buffers of the machines which become free at time t at Step 29. If yes, then move these jobs onto the machines by selecting them according to rule 2 (**R2**).

Step 31: Go to Step 8.

As it can be seen, the algorithm is uniquely defined by the choice of six dispatching rules at different decision points. The combined effects of various dispatching rules are tested to analyze the sensitivity of the system.

8.3 Performance Measures

The relative performance of rules is compared against different measures given below:

(a) *Mean waiting-time (WT)* includes total waiting time in the central buffer, total waiting time in the input buffer, total waiting time in the output buffer, total waiting time on the machine during blocking, and the mean waiting-time (WT) is the average of the waiting times of all parts.

(b) *Flow-time (FT)* consists of total processing time, total waiting time, total transfer time, total load / unload time, and the mean flow-time (MFT) is the average of the flow times of all part types.

(c) *Makespan (MS).*

(d) *Blocking-time percentage (BT)* is obtained by dividing total blocking time on all machines by the makespan.

(e) *AGV utilization* is calculated by dividing the average duration that an AGV is busy by the makespan.

8.4 Scheduling Rules

Since the system performance is sensitive to the joint-effect of all rules imbedded in the algorithm, it is necessary to try all relevant combinations of rules at different decision points, and thus a scheduling strategy is defined by a 6-tuple **(R1, R2, R3, R4, R5, R6)**. Since there exist a large number of rules in the scheduling literature, without loss of

generality some well-known rules given in Table 8.4.1 are selected to be analyzed under the experimental conditions to be mentioned below.

TABLE 8.4.1. Scheduling Rules

Decision Point	Short Name	Description
R1	SPT	Choose the part with the shortest processing time.
R1	LPT	Choose the part with the longest processing time.
R2	FCFS	Choose the part that arrived first at the input buffer.
R2	SPT	Choose the part with the shortest processing time.
R2	MWKR	Choose the part with the most work remaining.
R3	SDT	Choose the blocked machine which is closest to the free AGV.
R4	LWJ	Choose the part waiting for the longest time.
R4	SWTIB	Choose the part which will wait for the shortest time in the input buffer of the destination station.
R5	LWJ	Choose the part waiting for the longest time.
R5	SDT	Choose the part which is closest to the AGV.
R6	LWJ	Choose the part waiting for the longest time.
R6	SWTIB	Choose the part which will wait for the shortest time in the input buffer of the destination station.

8.5 Experimental Design and Results

The purpose of the scheduling stage is to study the performance of the FMS algorithm discussed in Section 8.2. All the test problems generated in the batching and loading stages are analyzed under the following experimental conditions:

(a) varying the number of AGVs between one and five;

(b) varying the input / output buffer capacities between one and five and output buffer capacities between one and four;

(c) varying the ratio of AGV traveling time to the machine processing time (TPR) between 0.25 and 1.00;

(d) varying the scheduling rules in the algorithm;

(e) using different performance measures.

The test environments are not designed with the same parameter values for all problems. In fact, (a), (b) and (c) implicitly address some inherent design questions for a tight system load, so the problems are created in a manner where the AGV system becomes the bottleneck resource. It must be emphasized that the number of AGVs is increased whenever locking problem arises in the system, and input/output buffer capacities may also be changed to prevent this problem still keeping the AGVs as the critical resources. Hence, the experiments carried out for each test problem are performed in systems with different parameters in terms of the number of AGVs and the input/output buffer capacities to yield the same system load tightness.

A computer program is written in PROGRESS 4GL to perform the FMS scheduling according to the scheduling algorithm developed and to calculate the required statics.

8.5.1. Interrelations between Model Parameters and Performance Measures

The following results are found to be valid for all the models under varying experimental conditions:

(a) decrease in the input buffer capacity increases the mean-flow time and makespan;

(b) decrease in the number of AGVs increases the mean-flow time and makespan;

(c) decrease in the number of AGVs increases the waiting time in the output buffer and blocking time percentage;

(d) joint consideration of smaller input buffer capacity and higher number of AGVs does not necessarily improve the mean-flow time and the makespan;

(f) AGV utilization is correlated with the TPR value. It increases as the TPR increases. However, it must also be emphasized that the AGV utilization is high in most cases whatever the value of the TPR is, and the utilization generally varies between 0.80 and 1.00 related to the fact that AGV is the critical resource of the system.

8.5.2. Testing Performance of the Rules

(A) Mean-flow and makespan performances of **(R1)** in the proposed algorithm under varying number of operations are shown in Figures 8.5.1 and 8.5.2, respectively.

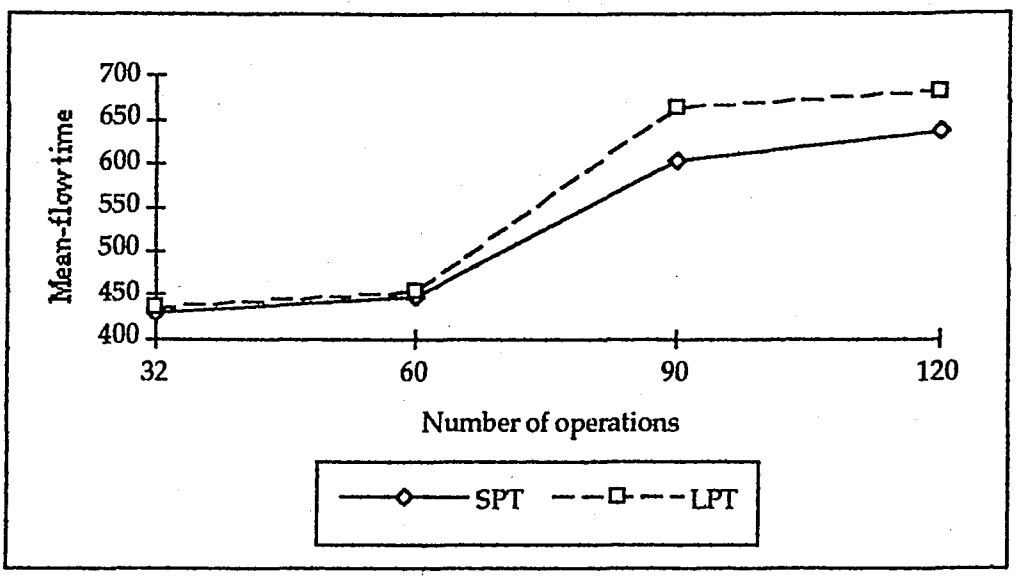

FIGURE 8.5.1. Mean-flow time performances of LPT and SPT in R1

As far as the mean-flow time and makespan criteria are concerned, it is observed that the performance of both rules are nearly the same for fairly low values of the number of operations. However, the difference between the performances of these rules in terms of mean-flow time and makespan becomes more apparent in favor of SPT as the number of operations in the system increases.

It is observed that locking may occur when the parts enter the system according to the LPT rule while no such problem arises if SPT rule is used instead. The existence of locking means that no part movement can take place in the system related to the buffer capacities and the number of AGVs in the system, and one way to avoid locking is to increase the input buffer capacity which is also observed in this study.

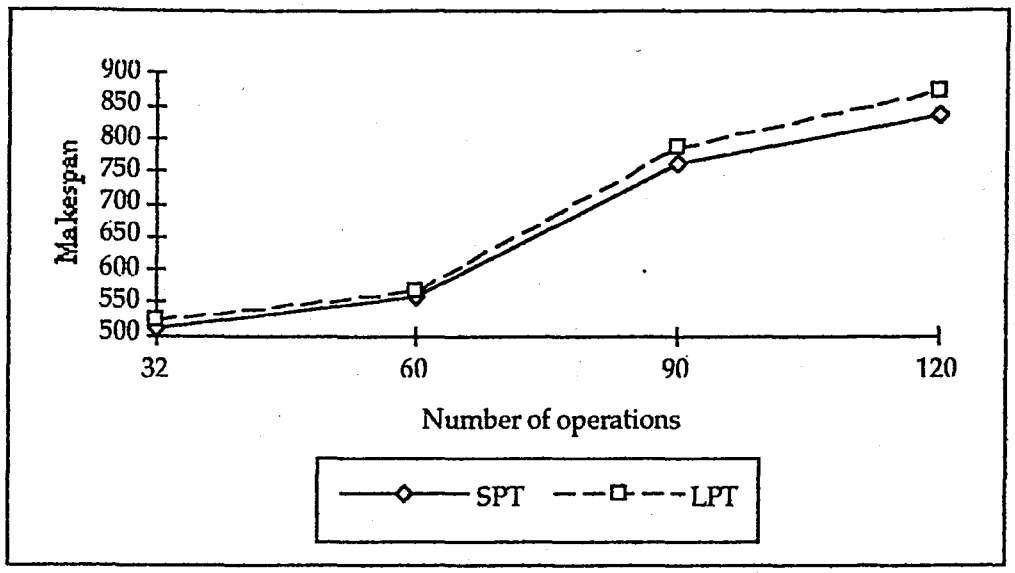

FIGURE 8.5.2. Makespan performances of LPT and SPT in R1

(B) Mean-flow and makespan performances of (**R2**) under varying TPR values are given in Figures 8.5.3 and 8.5.4, respectively.

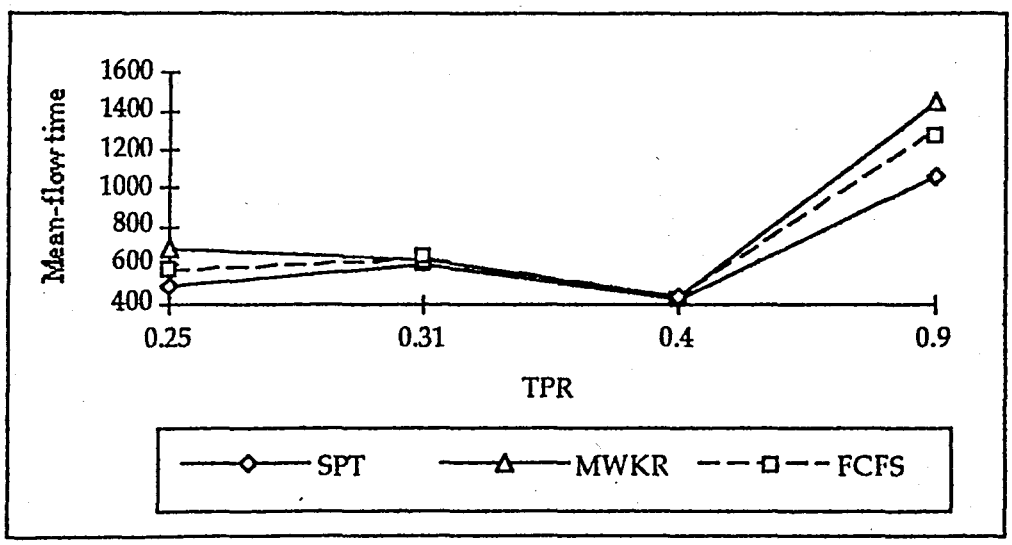

FIGURE 8.5.3. Mean-flow time performances of MWKR, FCFS and SPT in R2

Again it is observed that MWKR and SPT rules may create locking in some problems which do not experience locking under SPT and FCFS. With respect to mean-flow time optimization, MWKR rule gives the worst results. In fact, in all problems with different scheduling rule combinations SPT outperforms FCFS and MWKR with respect to mean-flow time optimization. However the respective performances of the rules show variations with respect to makespan optimization. For a given number of operations, by varying TPR

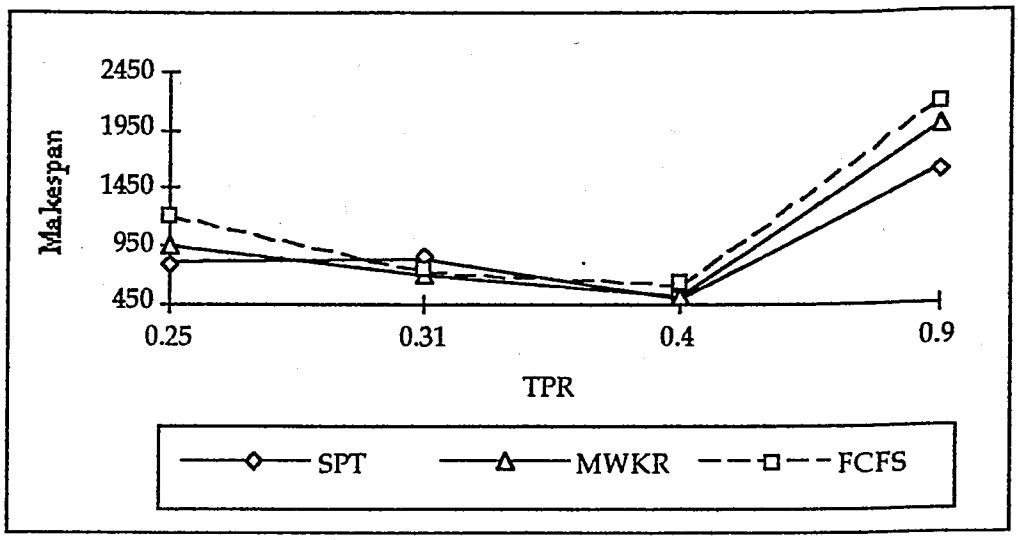

FIGURE 8.5.4. Makespan performances of MWKR, FCFS and SPT in R2

values one may obtain a few cases where MWKR or FCFS provides better results than SPT as it can be seen in Figure 8.5.4. On the other hand for a given TPR, by varying the number of operations one observes that MWKR rule performs better than the others as it can be seen in Figure 8.5.5 for a TPR value of 0.35.

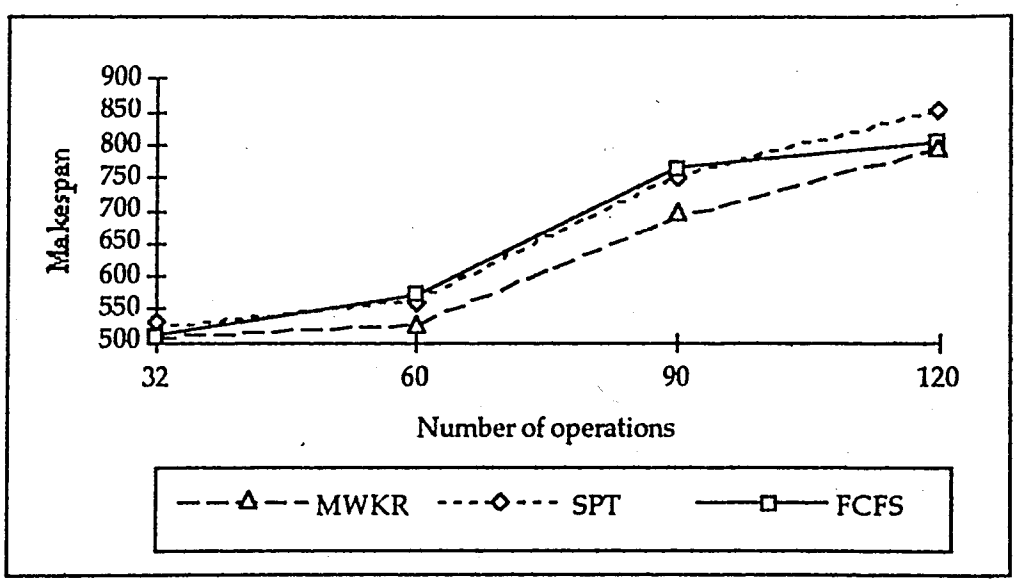

FIGURE 8.5.5. Makespan performances of MWKR, FCFS and SPT for R2

(C) Mean-flow and makespan performances of (**R4**) under varying number of operations are shown in Figures 8.5.6 and 8.5.7, respectively.

In all cases LWJ performs better than SWTIB regardless of TPR.

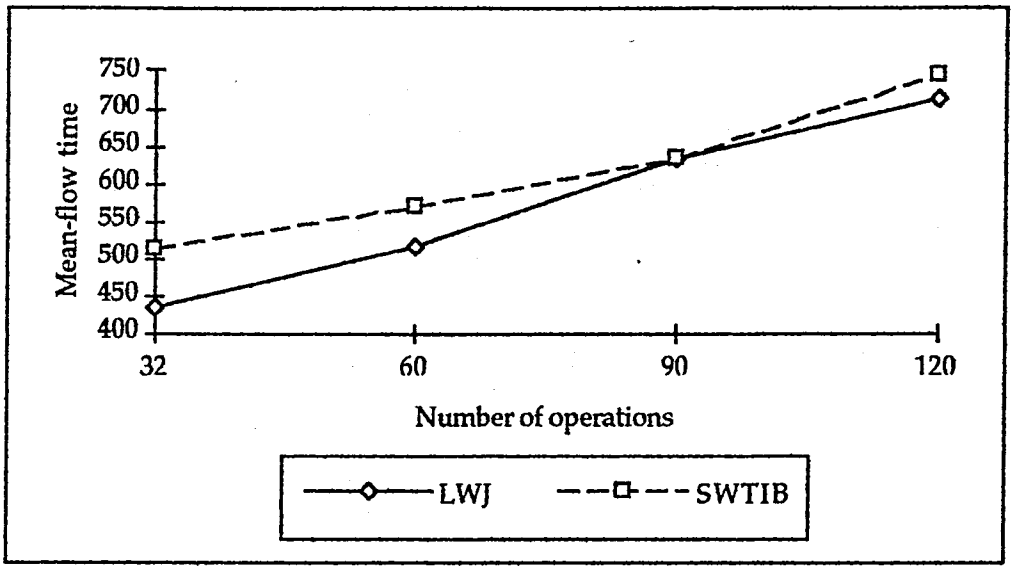

FIGURE 8.5.6. Mean-flow time performances of LWJ and SWTIB in R4

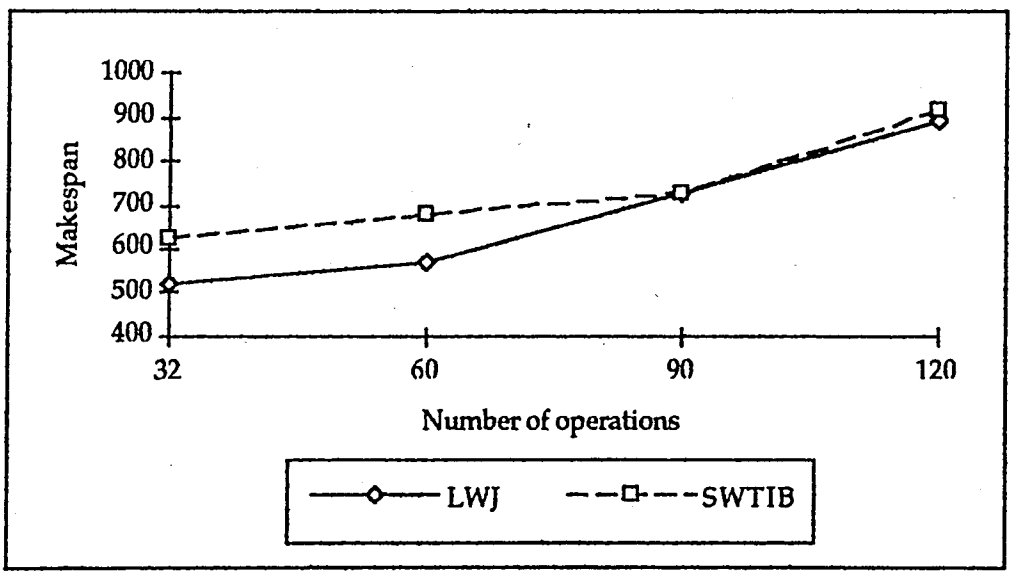

FIGURE 8.5.7. Makespan performances of LWJ and SWTIB in R4.

(D) Mean-flow and makespan performances of (**R5/R6**) combination under varying TPR values are shown in Figures 8.5.8 and 8.5.9, respectively.

It is observed that in most cases, LWJ/SWTIB rule combination gives the best results with respect to makespan optimization while SDT/SWTIB rule combination performs better with respect to mean flow time criterion.

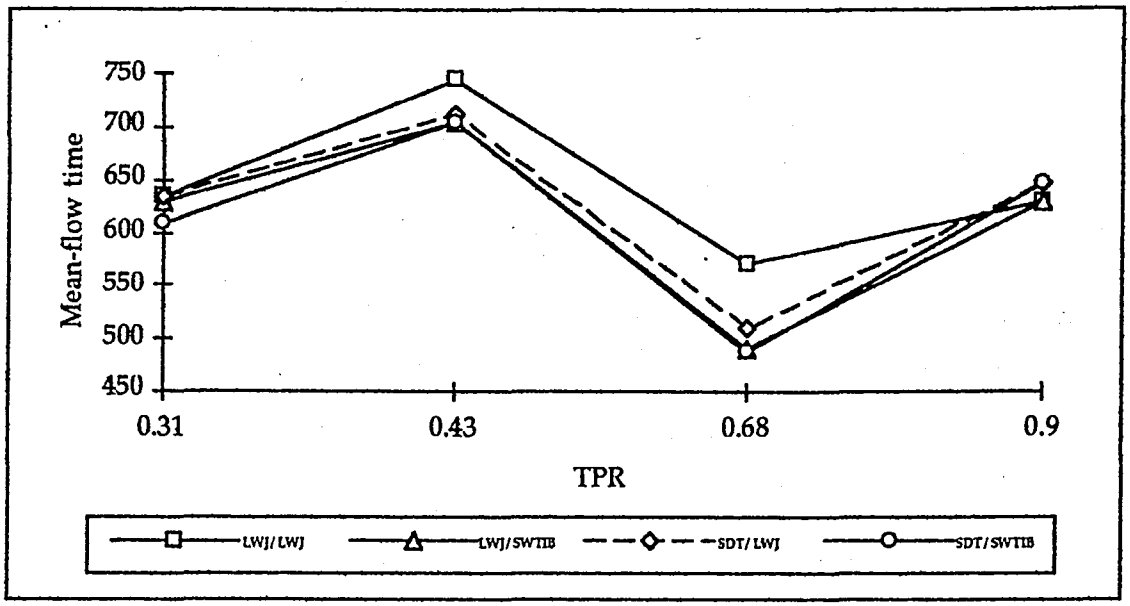

FIGURE 8.5.8. Mean-flow time performances of LWJ/LWJ, LWJ/SWTIB, SDT/LWJ and SDT/SWTIB rule combinations in (R5/R6)

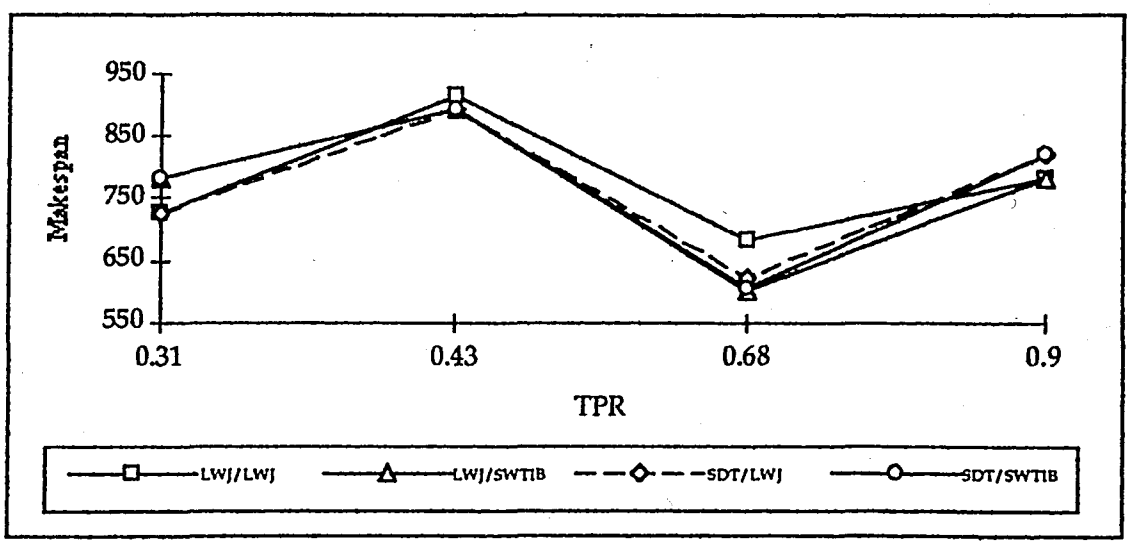

FIGURE 8.5.9. Makespan performances of LWJ/LWJ, LWJ/SWTIB, SDT/LWJ and SDT/SWTIB rule combinations in (R5/R6)

(E) Mean-flow time and makespan performances of **(R4/R5/R6)** combination under varying TPR values are shown in Figures 8.5.10, 8.5.11, 8.5.12 and 8.5.13.

When the results are examined, it is observed that there is not a rule combination which outperforms the other rule combinations. However, it can be seen that in many cases the rule combinations where R5 is LWJ and R6 is SWTIB provide better results than the

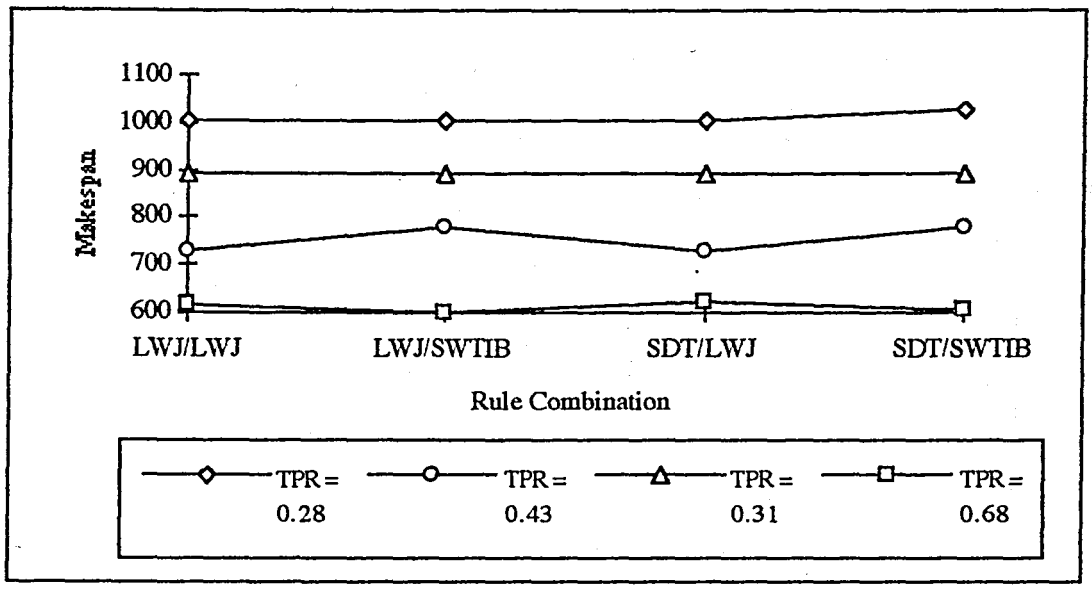

FIGURE 8.5.10. Makespan performances of LWJ/LWJ, LWJ/SWTIB, SDT/LWJ and SDT/SWTIB rule combinations in (R5/R6) while R4 is LWJ

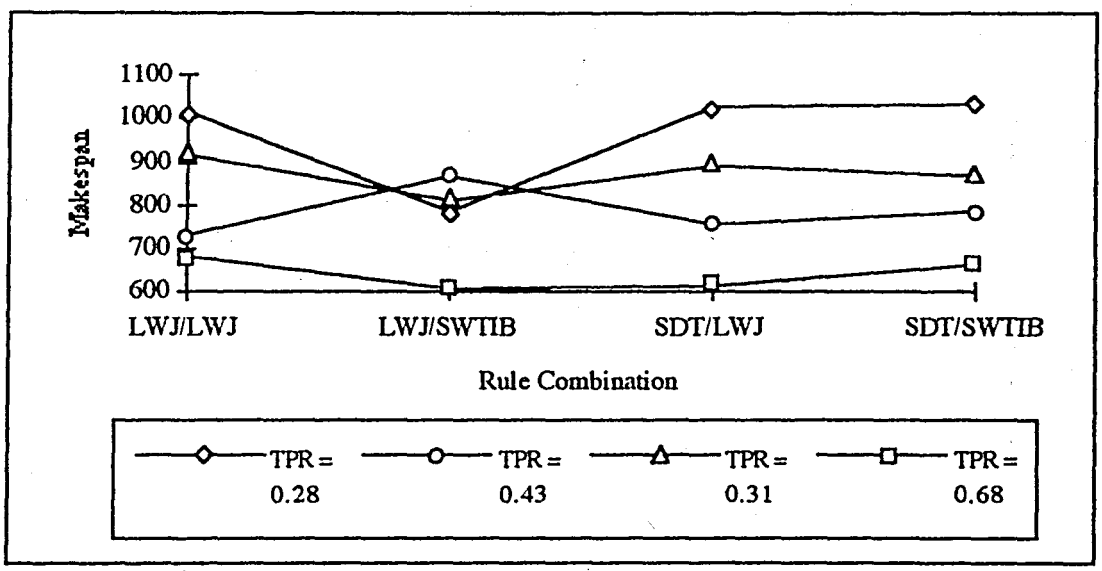

FIGURE 8.5.11. Makespan performances of LWJ/LWJ, LWJ/SWTIB, SDT/LWJ and SDT/SWTIB rule combinations in (R5/R6) while R4 is SWTIB

others, and this observation is in fact consistent with the conclusions drawn about (R5/R6) combinations in Part (D).

The results showing the mean-flow time performances of the rules show that again it is not possible to determine the best rule combination, but it can be concluded that

generally the combinations formed by taking R5 as SDT and R6 as SWTIB create better performances as stated in Part (D).

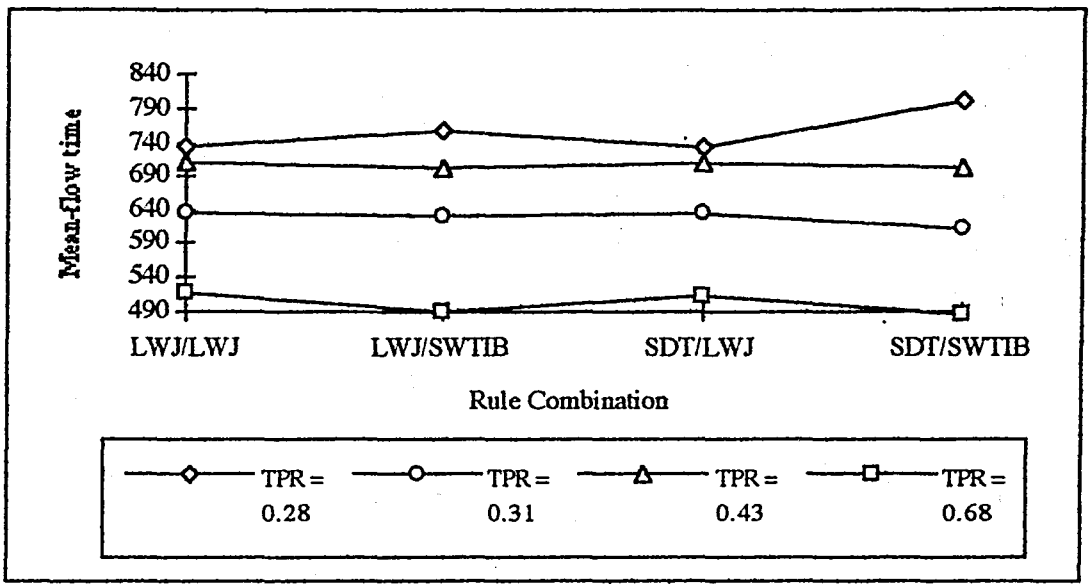

FIGURE 8.5.12 Mean-flow time performances of LWJ/LWJ, LWJ/SWTIB, SDT/LWJ and SDT/SWTIB rule combinations in (R5/R6) while R4 is LWJ

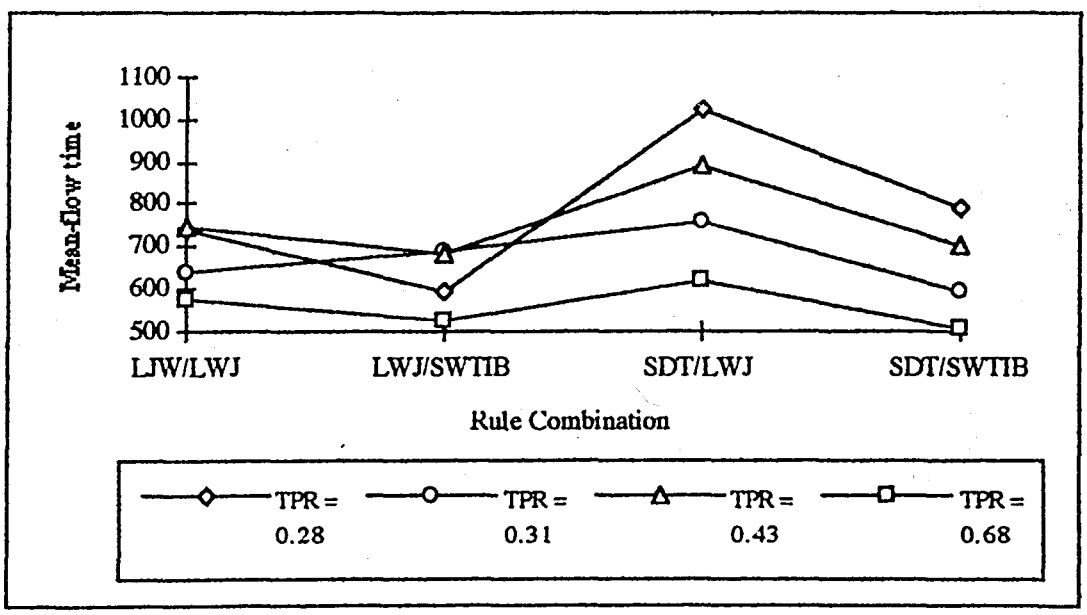

FIGURE 8.5.13. Mean-flow time performances of LWJ/LWJ, LWJ/SWTIB, SDT/LWJ and SDT/SWTIB rule combinations in (R5/R6) while R4 is SWTIB

Other conclusions that can be drawn about (R4/R5/R6) combination is that in most cases taking R4 as LWJ performs better than taking it as SWTIB in terms of both mean-flow

time and makespan, and (SWTIB/SDT/LWJ) combination gives the worst mean-flow time results in all cases.

9. CONCLUSION

In this study, the operational decisions which must be considered in the planning of Flexible Manufacturing Systems are structured into a hierarchy of three levels considering the fact that there exists a high interaction between them. At the top level of the hierarchy, batching decisions are considered by taking all the information in an aggregate manner. Later in the second level, the information sent from the upper level is disaggregated into detailed information in order to solve the loading problem. Finally at the lowest level, machines and material handling system are scheduled according to the information obtained from the loading level. Thus, an information flow from up to bottom is achieved, but also the information of the upper levels is updated by the feedback of the lower levels.

Batching problem in this study is solved by making use of a linear mathematical model and a greedy heuristic, and it is found out that such a sequential procedure provides enough flexibility to cope with the incoming orders while the current order is being processed and with unexpected conditions such as machine breakdowns. Thus, it is suggested that such a procedure with its dynamic structure can be a good alternative to the approach which divides the entire order into batches all at once.

The reason that the batching problem is expressed in aggregate terms is that it is not worth the effort to include all the details of the process while only determining the parts to enter the shift. However, it is observed in this study that batching decisions do not always provide good information for the loading level so that a feasible loading solution cannot always be obtained. Then, it is observed that there exists no efficient solution in literature to cope with this problem, and in this study, two solution approaches are suggested: an algorithm called Procedure PTD which is a unidimensional search similar to Golden section and which can generate a sharp and an efficient capacity value for the batching problem developed according to the pure top-down hierarchy approach and a heuristic anticipation process which is designed to generate a feasible loading solution in an iterative manner even if the initial capacity aggregation does not yield a feasible base level solution. Both procedures are run for several test problems, and it is found out that they provide really good upper bounds for the aggregate capacity values in the batching problem. It is also seen that the algorithm called Procedure PTD gives more sensitive values than the heuristic anticipation process.

For the loading problem, it is noticed that the integer models become hard-to-solve as the size of the problem gets larger especially for the models with objective functions (5.10), (5.12) and (5.13). Hence, it is seen that an alternative method is really necessary to solve the loading problem as the problem size gets larger, and it is observed that Lagrangian relaxation turns out to be an efficient method. The Lagrangian approach used in this study is furthermore the most extensive one in literature as far as the number and the type of the constraints are concerned. At the beginning of the runs, it was expected that the subgradient optimization procedure with the relaxed loading problem would be sufficient to obtain feasible results. However, the experiments showed that this was not always possible. Hence, a Lagrangian heuristic which makes the loading solutions feasible is suggested in this study, and it is found out that the heuristic gives reasonable bounds for the optimal values of the loading objectives. Another conclusion about the Lagrangian relaxation approach to the loading problem states that the choice of the reduction method for the step size in the subgradient optimization procedure depends on the problem under consideration, so no general conclusions can be drawn. In fact, it must be emphasized that halving the step size which is the most common method found in literature is not always the best method for all types of problems.

In the scheduling level, the test environments are designed in a manner that the AGV system is the critical resource. Hence, the number of AGVs is only increased when the locking problem arises, and the input/output buffer capacities may also be increased to prevent locking still keeping the AGVs as the critical resources. On the other hand, it is found out that increase in the input buffer capacity decreases the mean-flow time and makespan, and increase in the number of AGVs decreases the mean-flow time, makespan, waiting time in the output buffers and the blocking time percentage. It is also observed that the AGV utilization is highly correlated with the TPR value and the AGV system becomes a more critical resource as the TPR value increases.

APPENDIX

TABLE A.3.4.1. Costs and Processing Times for the 4-part, 4-operation, 20-tool, 4-machine Problem

i	j	t	m	c_{ijtm}	p_{ijtm}	i	j	t	m	c_{ijtm}	p_{ijtm}
1	1	1	1	24	104	2	4	12	3	34	84
1	1	1	2	23	170	3	1	12	1	10	67
1	1	2	3	14	101	3	1	12	3	34	84
1	2	4	2	35	120	3	1	15	4	14	134
1	2	4	3	30	106	3	2	9	3	21	90
1	2	7	1	14	68	3	2	9	4	25	120
1	2	7	2	29	130	3	2	18	4	40	40
1	3	6	2	15	110	3	3	11	2	17	117
1	3	6	3	17	118	3	3	11	4	22	132
1	4	10	1	21	84	3	3	19	2	27	47
1	4	10	2	35	76	3	4	3	1	8	82
1	4	13	4	28	100	3	4	3	2	17	85
2	1	1	1	21	114	3	4	14	2	19	110
2	1	1	2	35	126	4	1	2	3	22	49
2	1	3	1	28	114	4	1	4	2	31	114
2	1	3	2	21	98	4	1	4	3	37	140
2	2	8	3	33	119	4	2	5	1	36	137
2	2	16	1	54	25	4	2	5	4	29	118
2	3	10	1	27	106	4	2	20	2	39	38
2	3	17	3	42	29	4	3	13	4	35	120
2	4	4	2	14	116	4	4	7	1	13	68
2	4	4	3	25	112	4	4	7	2	8	53
2	4	12	1	17	96	4	4	8	3	24	87
$\alpha_m M_m = 380$ $S_m = 60$											

TABLE A.5.1. Processing times (15 parts, 8 operations, 20 tools, 6 machines) in the loading model

Part number		1																						
Operation number		1			2				3			4			5		6		7			8		
Compatible tool set		1	3	9	7	8	11	20	2	5	6	1	16	19	15	16	12	14	4	10	13	3	17	19
	Mach.																							
	1	21	18		17					18		42	41			12	26			19		32		
Processing	2	37	35		15		38	40			31	38		22				33	15	21		29		19
time on	3			27		33			19		29						29		18				25	
each	4			25			30			16					28			30			28			
machine	5		32	29	20				24					24			21			16	25			22
	6	25			23		32			22		39	45	30	31	14							28	23

Part number		2																						
Operation number		1		2			3			4			5			6				7		8		
Compatible tool set		9	15	1	7	20	3	4	13	4	7	12	1	11	20	3	7	8	12	4	6	8	9	14
	Mach.																							
	1			19	34		13				6	15	26			8	12		9					
Processing	2			27	39	20	10	12		7	9		16	17	23	10	6			14	12			30
time on	3	28						11		11		13						7	14	17	10	15	21	33
each	4	34	39						18					19									24	
machine	5	30			38		14		21		11	12				12	5		18				27	
	6		42	26	31						14		22	21			9							

Part number		3																						
Operation number		1			2			3			4			5			6			7		8		
Compatible tool set		12	17	18	1	9	5	6	5	14	7	8	11	10	16	15	2	6	17					
	Mach.																							
	1		5		11		17		18		14			15	28									
Processing	2				25			23		23	15		19	21										
time on	3		8	9		24		14				21								30	29	31		
each	4			12		20	22		21				24			30								
machine	5		11			28				25										37				
	6			12	28		25		23						24	35					35			

TABLE A.5.1. Processing times (15 parts, 8 operations, 20 tools, 6 machines) in the loading model (continued)

Part number		4																	
Operation number		1		2		3		4		5		6		7		8			
Compatible tool set		7	15	3	5	6	11	14	18	12	13	1	2	1	4	5	6	20	
	Mach.																		
	1	22		32	19					18		26		31		11			
Processing	2	25		31		25	19	28				28		33	31		8	12	
time on	3					27				23			30		28		7		
each	4		28		28		23		31		27					9			
machine	5	30		38				32		27	22		34						
	6	20	22		20		28					30		39		12			

Part number		5																	
Operation number		1			2		3		4	5		6			7			8	
Compatible tool set		18	19	4	7	20	1	3	5	5	6	1	9	19	3	7	15	14	18
	Mach.																		
	1						13	15	12	18		11			21	28			
Processing	2		17	10	5	11	12	18			10	13		16	18	29		16	
time on	3			9	8						9		18						
each	4	15							9	13			17				16		16
machine	5		20		11			23					22	18	27	31		12	
	6		23		14		18		17	19		17		23		23	21		

Part number		6																	
Operation number		1		2			3		4			5		6	7			8	
Compatible tool set		11	14	5	9	18	6	8	3	7	8	1	14	16	2	5	7	8	1	2	4
	Mach.																				
	1			41					19	28		11		14		22	21		15		
Processing	2	18	21				28		18	25		10	13				20		14		19
time on	3				37		31	37			20				15			24		16	18
each	4	19		39	35	30										19					
machine	5		23		30				22	21			17				18			13	
	6	23		28						18		13		17	20	20	19		11		

TABLE A.5.1. Processing times (15 parts, 8 operations, 20 tools, 6 machines) in the loading model (continued)

Part number		7																			
Operation number		1			2		3			4		5			6		7		8		
Compatible tool set		8	12	14	5	8	1	2	9	10	14	1	6	20	18	19	17	20	14	15	
	Mach.																				
	1		17		21		12			3		9									
Processing	2			19			11			4	6	11	8	13		10		25	28		
time on	3	15	20			20		5	7				15				23				
each	4				24				6						12					31	
machine	5		23	20				7		8	5					12			32		
	6				20		15		11			15				8	28			35	

Part number		8																						
Operation number		1			2			3			4			5			6			7			8		
Compatible tool set		2	4	8	9	13	15	6	7	18	4	6	7	8	10	14	6	9	12	3	4	17	18	19	20
	Mach.																								
	1								19				32		10				31	15					
Processing	2		23					15	25		23	29	30		11	12	17			10	11			21	23
time on	3	19	20	25	28			17			24	31		13			25	29	30		17	18			
each	4				26	25	27			21								32					19		
machine	5	23				21			29						16	15			28	19				17	
	6						23		21													21		15	

Part number		9															
Operation number		1		2		3	4		5		6		7		8		
Compatible tool set		1	5	2	7	9	1	20	18	19	1	2	7	9	11	13	
	Mach.																
	1	23	20		15		35				10		24				
Processing	2	24			20		37	34		14	15		30		31		
time on	3			19		27						21		28			
each	4		28			30			19					25	29	32	
machine	5			15	18	35				18		28	20	30		27	
	6	28	30		24		30			23	25		29		26		

TABLE A.5.1. Processing times (15 parts, 8 operations, 20 tools, 6 machines) in the loading model (continued)

Part number		10															
Operation number		1		2		3		4		5		6		7		8	
Compatible tool set		4	6	2	8	4	9	11	12	7	8	2	20	15	17	16	20
	Mach.																
	1								20	15						19	
Processing	2	10	15			13		22		21			21				23
time on	3	14		21	27	15	28		18		24	25			18		
each	4						31	24						14			
machine	5			17			36		24	28		24					
	6							28		17				19	24	23	

Part number		11															
Operation number		1		2		3		4		5		6		7		8	
Compatible tool set		18	20	3	5	2	9	4	7	8	11	12	14	1	5	7	
	Mach.																
	1			21	20				44			40		20	15	8	
Processing	2		15	18				33	50		48		30	18		14	
time on	3					15	24	41		51		35					
each	4	12			20		30				45				14		
machine	5			30		19			40			29	25			19	
	6				29				39		40			23	19	21	

Part number		12															
Operation number		1		2		3		4		5		6		7		8	
Compatible tool set		4	7	1	4	8	11	12	17	2	20	4	9	10	13	7	
	Mach.																
	1		14	10				14						35		45	
Processing	2	21	19	8	5		5				24	25		40		42	
time on	3	28			6	10		17	19	20		29	31				
each	4						3						37		44		
machine	5		23					19		24			34	29	40	48	
	6		27	19			8		24							50	

TABLE A.5.1. Processing times (15 parts, 8 operations, 20 tools, 6 machines) in the loading model (continued)

Part number		13															
Operation number		1		2		3		4		5		6		7		8	
Compatible tool set		2	7	8	10	10	15	4	6	3	14	7	20	8	12	13	17
	Mach.																
	1		20		19	27				45		33			25		
Processing	2		21		21	29		34	40	42	39	38	41				
time on	3	25		14				38	44					29	31		32
each	4						31									37	
machine	5	29	24		24					39	43	34			28	30	
	6		29				34					30					28

Part number		14															
Operation number		1		2		3		4		5		6		7		8	
Compatible tool set		1	3	5	8	9	10	7	13	2	5	4	8	10	14	12	18
	Mach.																
	1	10	17	19			27	20			29			30		17	
Processing	2	15	21				15	15				18		40	23		
time on	3				30	32				14		12	19			30	
each	4			24		35			19		31						35
machine	5		24			27	29	18	14	19				35	29	33	
	6	19		17				23			34						

Part number		15															
Operation number		1		2		3		4		5		6		7		8	
Compatible tool set		4	7	2	5	1	3	9	11	14	17	18	20	1	4	2	7
	Mach.																
	1		20		21	15	22							18			
Processing	2	11	19			19	27		9	12			25	24	29		15
time on	3	31		35				15			17				31	37	23
each	4				28			10	17			22					
machine	5		24	31			24	18		15						30	20
	6		17		22	22			20		19			21			24

TABLE A.5.2. Required Tool Slots for Each Tool Type

Tool type number	1	2	3	4	5	6	7	8	9	10
Required tool slots for each tool type	10	14	8	12	13	6	5	18	12	11
Tool type number	11	12	13	14	15	16	17	18	19	20
Required tool slots for each tool type	4	13	12	15	4	4	5	5	5	4

TABLE A.5.3. Aggregate Processing Times (15 parts, 8 operations, 20 tools, 6 machines) in the Batching Level

i \ j	1	2	3	4	5	6	7	8
1	27.67	27.56	22.71	35.13	21.25	27.8	20.29	25.43
2	34.6	29.25	14.14	11.33	20.57	10	13.6	27
3	9.5	22.67	20.2	22	18.6	22	32.5	32.83
4	24.5	33.67	23.63	30.33	23.4	29.6	32.4	9.83
5	18.75	9.71	16.5	12.67	13.8	17.22	23.78	14.67
6	20.8	34.29	32	21.43	13.57	19.2	20.40	15.14
7	19	21.25	9.25	6	11.83	10.5	25.33	31.5
8	22	25	21	28.17	12.83	27.43	15.86	19
9	25.5	18.5	30.67	34	17	19.8	26.57	29
10	13	21.67	24.6	22.67	21	23.33	18.75	21.67
11	14.5	23.67	22	41.17	46	31.8	18.17	15.5
12	22	9.6	6.5	18.6	22.67	31.2	37.6	46.25
13	24.67	19.5	30.25	39	41.6	35.2	28.25	31.75
14	17.67	22.5	27.5	18.17	25.4	16.33	31.4	28.75
15	20.33	27.4	21.5	14.83	15.75	23.5	24.6	24.83

TABLE A.6.2.1. Costs and Processing Times for the 8-part, 4-operation, 20-tool, 4-machine Problem

i	j	t	m	c_{ijtm}	p_{ijtm}	i	j	t	m	c_{ijtm}	p_{ijtm}
1	1	1	1	21	34	2	3	4	3	33	11
1	1	1	2	37	24	2	3	13	4	42	18
1	1	3	1	18	40	2	4	4	2	12	7
1	1	3	2	35	21	2	4	4	3	23	11
1	1	9	3	27	24	2	4	7	1	27	6
1	1	9	4	25	30	2	4	7	2	30	9
1	2	7	1	17	28	2	4	12	1	23	15
1	2	7	2	15	32	2	4	12	3	51	17
1	2	8	3	33	43	3	1	12	1	45	5
1	2	11	2	38	34	3	1	12	3	49	8
1	2	11	4	30	35	3	1	17	3	37	9
1	2	20	2	40	28	3	1	18	4	30	12
1	3	2	3	19	57	3	2	1	1	23	11
1	3	5	1	45	18	3	2	1	2	20	25
1	3	5	4	23	16	3	2	9	3	34	24
1	3	6	2	65	31	3	3	5	1	38	17
1	3	6	3	87	29	3	3	5	4	34	22
1	4	1	1	56	42	3	3	6	2	31	23
1	4	1	2	57	38	3	3	6	3	42	14
1	4	16	1	59	41	3	4	5	1	46	18
1	4	19	2	32	22	3	4	5	4	37	21
2	1	9	3	84	28	3	4	14	2	67	23
2	1	9	4	60	34	4	1	7	1	34	22
2	1	15	4	30	39	4	1	7	2	56	25
2	2	1	1	21	19	4	1	15	4	38	28
2	2	1	2	47	27	4	2	3	1	30	32
2	2	7	1	45	34	4	2	3	2	54	31
2	2	7	2	78	39	4	3	5	1	43	19
2	2	20	2	30	20	4	3	5	4	67	28
2	3	3	1	10	13	4	3	6	2	56	25
2	3	3	2	12	10	4	3	6	3	51	27
2	3	4	2	23	12	4	3	11	2	78	19

TABLE A.6.2.1. Costs and Processing Times for the 8-part, 4-operation, 20-tool, 4-machine Problem (continued)

i	j	t	m	c_{ijtm}	p_{ijtm}	i	j	t	m	c_{ijtm}	p_{ijtm}
4	3	11	4	54	23	7	1	8	3	32	15
4	4	14	2	65	28	7	1	12	1	31	17
4	4	18	4	57	31	7	1	12	3	29	20
5	1	18	4	60	15	7	1	14	2	37	19
5	1	19	2	31	17	7	2	5	1	24	21
5	2	4	2	20	10	7	2	5	4	37	24
5	2	4	3	18	9	7	2	8	3	32	20
5	2	7	1	12	5	7	3	1	1	21	12
5	2	7	2	23	8	7	3	1	2	45	11
5	2	20	2	24	11	7	3	2	3	51	5
5	3	1	1	29	13	7	3	9	3	18	7
5	3	1	2	28	12	7	3	9	4	25	6
5	3	3	1	30	15	7	4	10	1	23	3
5	3	3	2	30	18	7	4	10	2	32	4
5	4	5	1	34	12	7	4	14	2	40	6
5	4	5	4	25	9	8	1	2	3	21	19
6	1	11	2	36	18	8	1	4	2	28	23
6	1	11	4	38	19	8	1	4	3	27	20
6	1	14	2	40	21	8	1	8	3	32	25
6	2	5	1	23	41	8	2	9	3	27	28
6	2	5	4	30	39	8	2	9	4	38	26
6	2	9	3	45	37	8	2	13	4	32	25
6	2	9	4	56	35	8	2	15	4	38	27
6	2	18	4	57	30	8	3	6	2	23	15
6	3	6	2	32	28	8	3	6	3	38	17
6	3	6	3	46	31	8	3	7	1	23	19
6	3	8	3	31	37	8	3	7	2	20	25
6	4	3	1	21	19	8	3	18	4	25	21
6	4	3	2	39	18	8	4	4	2	34	23
6	4	7	1	33	28	8	4	4	3	20	24
6	4	7	2	35	25	8	4	6	2	56	29
6	4	8	3	25	20	8	4	6	3	54	31

TABLE A.6.2.1. Costs and Processing Times for the 8-part, 4-operation, 20-tool, 4-machine Problem (continued)

i	j	t	m	c_{ijtm}	P_{ijtm}	i	j	t	m	c_{ijtm}	P_{ijtm}
8	4	7	1	36	32	8	4	7	2	40	30
$\alpha_m M_m = 190$ $S_m = 70$											

TABLE A.6.2.2. Costs and Processing Times for the 15-part, 4-operation, 20-tool, 4-machine Problem

i	j	t	m	c_{ijtm}	p_{ijtm}	i	j	t	m	c_{ijtm}	p_{ijtm}
1	1	1	1	21	34	2	3	4	3	33	11
1	1	1	2	37	24	2	3	13	4	42	18
1	1	3	1	18	40	2	4	4	2	12	7
1	1	3	2	35	21	2	4	4	3	23	11
1	1	9	3	27	24	2	4	7	1	27	6
1	1	9	4	25	30	2	4	7	2	30	9
1	2	7	1	17	28	2	4	12	1	23	15
1	2	7	2	15	32	2	4	12	3	51	17
1	2	8	3	33	43	3	1	12	1	45	5
1	2	11	2	38	34	3	1	12	3	49	8
1	2	11	4	30	35	3	1	17	3	37	9
1	2	20	2	40	28	3	1	18	4	30	12
1	3	2	3	19	57	3	2	1	1	23	11
1	3	5	1	45	18	3	2	1	2	20	25
1	3	5	4	23	16	3	2	9	3	34	24
1	3	6	2	65	31	3	2	9	4	23	20
1	3	6	3	87	29	3	3	5	1	38	17
1	4	1	1	56	42	3	3	5	4	34	22
1	4	1	2	57	38	3	3	6	2	31	23
1	4	16	1	59	41	3	3	6	3	42	14
1	4	19	2	32	22	3	4	5	1	46	18
2	1	9	3	84	28	3	4	5	4	37	21
2	1	9	4	60	34	3	4	14	2	67	23
2	1	15	4	30	39	4	1	7	1	34	22
2	2	1	1	21	19	4	1	7	2	56	25
2	2	1	2	47	27	4	1	15	4	38	28
2	2	7	1	45	34	4	2	3	1	30	32
2	2	7	2	78	39	4	2	3	2	54	31
2	2	20	2	30	20	4	3	5	1	43	19
2	3	3	1	10	13	4	3	5	4	67	28
2	3	3	2	12	10	4	3	6	2	56	25
2	3	4	2	23	12	4	3	6	3	51	27

TABLE A.6.2.2. Costs and Processing Times for the 15-part, 4-operation, 20-tool, 4-machine Problem (continued)

i	j	t	m	c_{ijtm}	p_{ijtm}	i	j	t	m	c_{ijtm}	p_{ijtm}
4	3	11	2	78	19	6	4	8	3	25	20
4	3	11	4	54	23	7	1	8	3	32	15
4	4	14	2	65	28	7	1	12	1	31	17
4	4	18	4	57	31	7	1	12	3	29	20
5	1	18	4	60	15	7	1	14	2	37	19
5	1	19	2	31	17	7	2	5	1	24	21
5	2	4	2	20	10	7	2	5	4	37	24
5	2	4	3	18	9	7	2	8	3	32	20
5	2	7	1	12	5	7	3	1	1	21	12
5	2	7	2	23	8	7	3	1	2	45	11
5	2	20	2	24	11	7	3	2	3	51	5
5	3	1	1	29	13	7	3	9	3	18	7
5	3	1	2	28	12	7	3	9	4	25	6
5	3	3	1	30	15	7	4	10	1	23	3
5	3	3	2	30	18	7	4	10	2	32	4
5	4	5	1	34	12	8	1	2	3	21	19
5	4	5	4	25	9	8	1	4	2	28	23
6	1	11	2	36	18	8	1	4	3	27	20
6	1	11	4	38	19	8	1	8	3	32	25
6	1	14	2	40	21	8	2	9	3	27	28
6	2	5	1	23	41	8	2	9	4	38	26
6	2	5	4	30	39	8	2	13	4	32	25
6	2	9	3	45	37	8	2	15	4	38	27
6	2	9	4	56	35	8	3	6	2	23	15
6	2	18	4	57	30	8	3	6	3	38	17
6	3	6	2	32	28	8	3	7	1	23	19
6	3	6	3	46	31	8	3	7	2	20	25
6	3	8	3	31	37	8	3	18	4	25	21
6	4	3	1	21	19	8	4	4	2	34	23
6	4	3	2	39	18	8	4	4	3	20	24
6	4	7	1	33	28	8	4	6	2	56	29
6	4	7	2	35	25	8	4	6	3	54	31

TABLE A.6.2.2. Costs and Processing Times for the 15-part, 4-operation, 20-tool, 4-machine Problem (continued)

i	j	t	m	c_{ijtm}	p_{ijtm}	i	j	t	m	c_{ijtm}	p_{ijtm}
8	4	7	1	36	32	11	2	5	1	24	20
8	4	7	2	40	30	11	2	5	4	31	24
9	1	1	1	56	23	11	3	2	3	28	15
9	1	1	2	32	24	11	3	9	3	30	24
9	1	5	1	43	20	11	3	9	4	35	30
9	1	5	4	61	28	11	4	4	2	35	33
9	2	2	3	23	19	11	4	4	3	47	41
9	2	7	1	32	15	11	4	7	1	56	44
9	2	7	2	25	20	11	4	7	2	59	50
9	3	9	3	30	27	12	1	4	2	58	21
9	3	9	4	32	30	12	1	4	3	45	28
9	4	1	1	32	35	12	1	7	1	56	14
9	4	1	2	34	24	12	1	7	2	56	19
9	4	20	2	30	34	12	2	1	1	23	10
10	1	4	2	23	10	12	2	1	2	45	8
10	1	4	3	24	14	12	2	4	2	56	5
10	1	6	2	27	15	12	2	4	3	32	6
10	1	6	3	44	18	12	3	8	3	23	10
10	2	2	3	37	21	12	3	11	2	43	5
10	2	8	3	45	27	12	3	11	4	67	3
10	3	4	2	20	13	12	4	12	1	22	14
10	3	4	3	38	15	12	4	12	3	32	17
10	3	9	3	34	28	12	4	17	3	44	19
10	3	9	4	35	31	13	1	2	3	27	25
10	4	11	2	24	22	13	1	7	1	24	20
10	4	11	4	34	24	13	1	7	2	55	21
10	4	12	1	35	20	13	2	8	3	15	14
10	4	12	3	32	18	13	2	10	1	10	19
11	1	18	4	29	12	13	2	10	2	33	21
11	1	20	2	38	15	13	3	10	1	34	17
11	2	3	1	23	21	13	3	10	2	23	29
11	2	3	2	35	18	13	3	15	4	21	31

TABLE A.6.2.2. Costs and Processing Times for the 15-part, 4-operation, 20-tool, 4-machine Problem (continued)

i	j	t	m	c_{ijtm}	p_{ijtm}	i	j	t	m	c_{ijtm}	p_{ijtm}
13	4	4	2	55	34	14	4	7	2	17	15
13	4	4	3	60	38	14	4	13	4	21	19
13	4	6	2	45	40	15	1	4	2	17	11
13	4	6	3	40	44	15	1	4	3	34	31
14	1	1	1	21	10	15	1	7	1	25	20
14	1	1	2	45	15	15	1	7	2	23	19
14	1	3	1	40	17	15	2	2	3	45	35
14	1	3	2	50	21	15	2	5	1	36	21
14	2	5	1	32	19	15	2	5	4	34	28
14	2	5	4	45	24	15	3	1	1	27	15
14	2	8	3	35	30	15	3	1	2	56	19
14	3	9	3	43	32	15	3	3	1	27	15
14	3	9	4	67	35	15	3	3	2	55	27
14	3	10	1	21	27	15	4	9	3	23	15
14	3	10	2	34	15	15	4	9	4	18	9
14	4	7	1	24	20	15	4	11	2	44	17
$\alpha_m M_m = 350 \quad S_m = 80$											

TABLE A.6.2.3. Costs and Processing Times for the 15-part, 4-operation, 20-tool, 8-machine Problem

i	j	t	m	c_{ijtm}	p_{ijtm}	i	j	t	m	c_{ijtm}	p_{ijtm}
1	1	1	1	21	34	2	1	9	4	60	34
1	1	1	2	37	24	2	1	9	5	37	30
1	1	3	1	18	40	2	1	15	4	30	39
1	1	3	2	35	21	2	1	15	6	54	42
1	1	3	5	30	32	2	2	1	1	21	19
1	1	9	3	27	24	2	2	1	2	47	27
1	1	9	4	25	30	2	2	1	6	48	26
1	1	9	5	54	29	2	2	7	1	45	34
1	2	7	1	17	28	2	2	7	2	78	39
1	2	7	2	15	32	2	2	7	5	67	38
1	2	7	5	56	20	2	2	7	6	60	31
1	2	7	6	54	23	2	2	20	2	30	20
1	2	8	3	33	43	2	3	3	1	10	13
1	2	8	7	54	35	2	3	3	2	12	10
1	2	11	2	38	34	2	3	3	5	34	14
1	2	11	4	30	35	2	3	4	2	23	12
1	2	20	2	40	28	2	3	4	3	33	11
1	3	2	3	19	57	2	3	13	4	42	18
1	3	2	5	33	24	2	3	13	5	43	21
1	3	5	1	45	18	2	3	13	8	24	13
1	3	5	4	23	16	2	4	4	2	12	7
1	3	6	2	65	31	2	4	4	3	23	11
1	3	6	3	87	29	2	4	7	1	27	6
1	4	1	1	56	42	2	4	7	2	30	9
1	4	1	2	57	38	2	4	7	5	33	11
1	4	1	6	42	39	2	4	7	6	55	14
1	4	16	1	59	41	2	4	12	1	23	15
1	4	16	6	40	45	2	4	12	3	51	17
1	4	19	2	32	22	2	4	12	5	54	12
1	4	19	5	33	24	3	1	12	1	45	5
1	4	19	6	35	30	3	1	12	3	49	8
2	1	9	3	84	28	3	1	12	5	24	11

TABLE A.6.2.3. Costs and Processing Times for the 15-part, 4-operation, 20-tool, 8-machine Problem (continued)

i	j	t	m	c_{ijtm}	p_{ijtm}	i	j	t	m	c_{ijtm}	p_{ijtm}
3	1	17	3	37	9	4	3	11	2	78	19
3	1	17	6	38	12	4	3	11	4	54	23
3	1	18	4	30	12	4	3	11	6	52	28
3	2	1	1	23	11	4	4	14	2	65	28
3	2	1	2	20	25	4	4	14	5	30	32
3	2	1	6	44	28	4	4	18	4	57	31
3	2	9	3	34	24	5	1	18	4	60	15
3	2	9	4	34	28	5	1	19	2	31	17
3	2	9	5	34	28	5	1	19	5	67	20
3	3	5	1	38	17	5	1	19	6	68	23
3	3	5	4	34	22	5	2	4	2	20	10
3	3	5	6	50	25	5	2	4	3	18	9
3	3	6	2	31	23	5	2	7	1	12	5
3	3	6	3	42	14	5	2	7	2	23	8
3	4	5	1	46	18	5	2	7	5	42	11
3	4	5	4	37	21	5	2	7	6	60	14
3	4	5	6	43	23	5	2	20	2	24	11
3	4	14	2	67	23	5	3	1	1	29	13
3	4	14	5	27	25	5	3	1	2	28	12
4	1	7	1	34	22	5	3	1	6	52	18
4	1	7	2	56	25	5	3	3	1	30	15
4	1	7	5	46	30	5	3	3	2	30	18
4	1	7	6	40	20	5	3	3	5	32	23
4	1	15	4	38	28	5	4	5	1	34	12
4	1	15	6	35	22	5	4	5	4	25	9
4	2	3	1	30	32	5	4	5	6	42	17
1	2	3	5	54	38	6	1	11	2	36	18
4	3	5	1	43	19	6	1	11	4	38	19
4	3	5	4	67	28	6	1	11	6	44	23
4	3	5	6	63	20	6	1	14	2	40	21
4	3	6	2	56	25	6	1	14	5	45	23
4	3	6	3	51	27	6	2	5	1	23	41

TABLE A.6.2.3. Costs and Processing Times for the 15-part, 4-operation, 20-tool, 8-machine Problem (continued)

i	j	t	m	c_{ijtm}	p_{ijtm}	i	j	t	m	c_{ijtm}	p_{ijtm}
6	2	5	4	30	39	7	3	2	5	44	7
6	2	5	6	34	28	7	3	9	3	18	7
6	2	9	3	45	37	7	3	9	4	25	6
6	2	9	4	56	35	7	3	9	5	22	11
6	2	9	5	37	30	7	4	10	1	23	3
6	2	18	4	57	30	7	4	10	2	32	4
6	3	6	2	32	28	7	4	10	5	31	8
6	3	6	3	46	31	7	4	10	7	39	6
6	3	8	3	31	37	7	4	14	2	40	6
6	4	3	1	21	19	7	4	14	5	44	9
6	4	3	2	39	18	8	1	2	3	21	19
6	4	3	5	54	22	8	1	2	5	56	23
6	4	7	1	33	28	8	1	4	2	28	23
6	4	7	2	35	25	8	1	4	3	27	20
6	4	7	5	67	21	8	1	8	3	32	25
6	4	7	6	43	18	8	1	8	7	31	27
6	4	8	3	25	20	8	2	9	3	27	28
6	4	8	7	41	23	8	2	9	4	38	26
7	1	8	3	32	15	8	2	13	4	32	25
7	1	12	1	31	17	8	2	13	5	51	25
7	1	12	3	29	20	8	2	13	8	48	15
7	1	12	5	43	23	8	2	15	4	38	27
7	1	14	2	37	19	8	2	15	6	66	23
7	1	14	5	43	23	8	3	6	2	23	15
7	2	5	1	24	21	8	3	6	3	38	17
7	2	5	4	37	24	8	3	7	1	23	19
7	2	5	6	23	20	8	3	7	2	20	25
7	2	8	3	32	20	8	3	7	5	31	29
7	3	1	1	21	12	8	3	7	6	43	21
7	3	1	2	45	11	8	3	18	4	25	21
7	3	1	6	23	15	8	4	4	2	34	23
7	3	2	3	51	5	8	4	4	3	20	24

TABLE A.6.2.3. Costs and Processing Times for the 15-part, 4-operation, 20-tool, 8-machine Problem (continued)

i	j	t	m	c_{ijtm}	p_{ijtm}	i	j	t	m	c_{ijtm}	p_{ijtm}
8	4	6	2	56	29	10	3	4	3	38	15
8	4	6	3	54	31	10	3	9	3	34	28
8	4	7	1	36	32	10	3	9	4	35	31
8	4	7	2	40	30	10	3	9	5	41	36
9	1	1	1	56	23	10	4	11	2	24	22
9	1	1	2	32	24	10	4	11	4	34	24
9	1	1	6	48	28	10	4	11	6	32	28
9	1	5	1	43	20	10	4	12	1	35	20
9	1	5	4	61	28	10	4	12	3	32	18
9	1	5	6	55	30	10	4	12	5	33	24
9	2	2	3	23	19	11	1	20	2	38	15
9	2	2	5	76	15	11	1	18	4	29	12
9	2	7	1	32	15	11	2	3	1	23	21
9	2	7	2	25	20	11	2	3	2	35	18
9	2	7	5	31	18	11	2	3	5	50	30
9	2	7	6	48	24	11	2	5	1	24	20
9	3	9	3	30	27	11	2	5	4	31	24
9	3	9	4	32	30	11	2	5	6	31	29
9	3	9	5	58	35	11	3	2	3	28	15
9	4	1	1	32	35	11	3	2	5	32	19
9	4	1	2	34	24	11	3	9	3	30	24
9	4	1	6	35	30	11	3	9	4	35	30
9	4	20	2	30	34	11	4	4	2	35	33
10	1	4	2	23	10	11	4	4	3	47	41
10	1	4	3	24	14	11	4	7	1	56	44
10	1	6	2	27	15	11	4	7	2	59	50
10	1	6	3	44	18	11	4	7	5	44	40
10	2	2	3	37	21	11	4	7	6	43	39
10	2	2	5	33	17	12	1	4	2	58	21
10	2	8	3	45	27	12	1	4	3	45	28
10	2	8	7	43	29	12	1	7	1	56	14
10	3	4	2	20	13	12	1	7	2	59	50

TABLE A.6.2.3. Costs and Processing Times for the 15-part, 4-operation, 20-tool, 8-machine Problem (continued)

i	j	t	m	c_{ijtm}	P_{ijtm}	i	j	t	m	c_{ijtm}	P_{ijtm}
12	1	7	5	34	23	13	3	15	4	21	31
12	1	7	6	54	27	13	3	15	6	32	34
12	2	1	1	23	10	13	4	4	2	55	34
12	2	1	2	45	8	13	4	4	3	60	38
12	2	1	6	41	19	13	4	6	2	45	40
12	2	4	2	56	5	13	4	6	3	40	44
12	2	4	3	32	6	14	1	1	1	21	10
12	3	8	3	23	10	14	1	1	2	45	15
12	3	8	7	28	14	14	1	3	1	40	17
12	3	11	2	43	5	14	1	3	2	50	21
12	3	11	4	67	3	14	1	3	5	50	24
12	3	11	6	26	8	14	2	5	1	32	19
12	4	12	1	22	14	14	2	5	4	45	24
12	4	12	3	32	17	14	2	5	6	23	17
12	4	12	5	31	19	14	2	8	3	35	30
12	4	17	3	44	19	14	2	8	7	45	33
12	4	17	6	40	24	14	3	9	3	43	32
13	1	2	3	27	25	14	3	9	4	67	35
13	1	2	5	32	29	14	3	9	5	32	27
13	1	7	1	24	20	14	3	10	1	21	27
13	1	7	2	55	21	14	3	10	2	34	15
13	1	7	5	44	24	14	3	10	5	25	29
13	1	7	6	32	29	14	3	10	7	29	13
13	2	8	3	15	14	14	4	7	1	24	20
13	2	8	7	21	12	14	4	7	2	17	15
13	2	10	1	10	19	14	4	7	5	36	18
13	2	10	2	33	21	14	4	7	6	40	23
13	2	10	5	43	24	14	4	13	4	21	19
13	2	10	7	27	24	14	4	13	5	21	14
13	3	10	1	34	17	14	4	13	8	30	12
13	3	10	2	23	29	15	1	4	2	17	11
13	3	10	7	21	28	15	1	4	3	34	31

TABLE A.6.2.3. Costs and Processing Times for the 15-part, 4-operation, 20-tool, 8-machine Problem (continued)

i	j	t	m	c_{ijtm}	p_{ijtm}	i	j	t	m	c_{ijtm}	p_{ijtm}
15	1	7	1	25	20	15	3	1	6	51	22
15	1	7	2	23	19	15	3	3	1	21	22
15	1	7	5	36	24	15	3	3	2	55	27
15	1	7	6	43	17	15	3	3	5	31	24
15	2	2	3	45	35	15	4	9	3	23	15
15	2	2	5	34	31	15	4	9	4	23	10
15	2	5	1	36	21	15	4	9	5	42	18
15	2	5	4	34	28	15	4	11	2	18	9
15	2	5	6	43	22	15	4	11	4	44	17
15	3	1	1	27	15	15	4	11	6	39	20
$\alpha_m M_m = 350$ $S_m = 80$											

TABLE A.6.2.4. Costs and Processing Times for the 15-part, 6-operation, 20-tool, 4-machine Problem

i	j	t	m	c_{ijtm}	p_{ijtm}	i	j	t	m	c_{ijtm}	p_{ijtm}
1	1	1	1	21	34	2	2	7	2	78	39
1	1	1	2	37	24	2	2	20	2	30	20
1	1	3	1	18	40	2	3	3	1	10	13
1	1	3	2	35	21	2	3	3	2	12	10
1	1	9	3	27	24	2	3	4	2	23	12
1	1	9	4	25	30	2	3	4	3	33	11
1	2	7	1	17	28	2	3	13	4	42	18
1	2	7	2	15	32	2	4	4	2	12	7
1	2	8	3	33	43	2	4	4	3	23	11
1	2	11	2	38	34	2	4	7	1	27	6
1	2	11	4	30	35	2	4	7	2	30	9
1	2	20	2	40	28	2	4	12	1	23	15
1	3	2	3	19	57	2	4	12	3	51	17
1	3	5	1	45	18	2	5	1	1	33	26
1	3	5	4	23	16	2	5	1	2	54	16
1	3	6	2	65	31	2	5	11	4	33	19
1	3	6	3	87	29	2	5	20	2	37	23
1	4	1	1	56	42	2	6	3	1	12	8
1	4	1	2	57	38	2	6	3	2	23	10
1	4	16	1	56	42	2	6	7	1	41	12
1	4	19	2	32	22	2	6	7	2	45	6
1	5	15	4	31	28	2	6	8	3	46	7
1	5	16	1	43	12	2	6	12	1	45	5
1	6	12	1	48	26	2	6	12	3	52	14
1	6	12	3	44	29	3	1	12	1	45	5
1	6	14	2	37	33	3	1	12	3	49	8
2	1	9	3	84	28	3	1	17	3	37	9
2	1	9	4	60	34	3	1	18	4	30	12
2	1	15	4	30	39	3	2	1	1	23	11
2	2	1	1	21	19	3	2	1	2	20	25
2	2	1	2	47	27	3	2	9	3	34	24
2	2	7	1	45	34	3	2	9	4	23	20

TABLE A.6.2.4. Costs and Processing Times for the 15-part, 6-operation, 20-tool, 4-machine Problem (continued)

i	j	t	m	c_{ijtm}	P_{ijtm}	i	j	t	m	c_{ijtm}	P_{ijtm}
3	3	5	1	38	17	4	6	1	2	55	28
3	3	5	4	34	22	4	6	2	3	35	30
3	3	6	2	31	23	5	1	18	4	60	15
3	3	6	3	42	14	5	1	19	2	31	17
3	4	5	1	46	18	5	2	4	2	20	10
3	4	5	4	37	21	5	2	4	3	18	9
3	4	14	2	67	23	5	2	7	1	12	5
3	5	7	1	27	14	5	2	7	2	23	8
3	5	7	2	36	15	5	2	20	2	24	11
3	5	8	3	44	21	5	3	1	1	29	13
3	5	11	2	47	19	5	3	1	2	28	12
3	5	11	4	56	24	5	3	3	1	30	15
3	6	10	1	31	15	5	3	3	2	30	18
3	6	10	2	51	25	5	4	5	1	34	12
3	6	16	1	55	28	5	4	5	4	25	9
4	1	7	1	34	22	5	5	5	1	56	18
4	1	7	2	56	25	5	5	5	4	52	13
4	1	15	4	38	28	5	5	6	2	19	10
4	2	3	1	30	32	5	5	6	3	34	9
4	2	3	2	54	31	5	6	1	1	23	11
4	3	5	1	43	19	5	6	1	2	31	13
4	3	5	4	67	28	5	6	9	3	45	18
4	3	6	2	56	25	5	6	9	4	41	17
4	3	6	3	51	27	5	6	19	2	39	16
4	3	11	2	78	19	6	1	11	2	36	18
4	3	11	4	54	23	6	1	11	4	38	19
4	4	14	2	65	28	6	1	14	2	40	21
4	4	18	4	57	31	6	2	5	1	23	41
4	5	12	1	37	18	6	2	5	4	30	39
4	5	12	3	27	23	6	2	9	3	45	37
4	5	13	4	34	27	6	2	9	4	56	35
4	6	1	1	60	26	6	2	18	4	57	30

TABLE A.6.2.4. Costs and Processing Times for the 15-part, 6-operation, 20-tool, 4-machine Problem (continued)

i	j	t	m	c_{ijtm}	p_{ijtm}	i	j	t	m	c_{ijtm}	p_{ijtm}
6	3	6	2	32	28	7	5	6	3	60	15
6	3	6	3	46	31	7	5	20	2	31	13
6	3	8	3	31	37	7	6	18	4	44	12
6	4	3	1	21	19	7	6	19	2	29	10
6	4	3	2	39	18	8	1	2	3	21	19
6	4	7	1	33	28	8	1	4	2	28	23
6	4	7	2	35	25	8	1	4	3	27	20
6	4	8	3	25	20	8	1	8	3	32	25
6	5	1	1	23	11	8	2	9	3	27	28
6	5	1	2	24	10	8	2	9	4	38	26
6	5	14	2	27	13	8	2	13	4	32	25
6	5	16	1	23	14	8	2	15	4	38	27
6	6	2	3	30	15	8	3	6	2	23	15
6	6	5	1	28	22	8	3	6	3	38	17
6	6	5	4	38	19	8	3	7	1	23	19
7	1	8	3	32	15	8	3	7	2	20	25
7	1	12	1	31	17	8	3	18	4	25	21
7	1	12	3	29	20	8	4	4	2	34	23
7	1	14	2	37	19	8	4	4	3	20	24
7	2	5	1	24	21	8	4	6	2	56	29
7	2	5	4	37	24	8	4	6	3	54	31
7	2	8	3	32	20	8	4	7	1	23	19
7	3	1	1	21	12	8	4	7	2	40	30
7	3	1	2	45	11	8	5	8	3	28	13
7	3	2	3	51	5	8	5	10	1	20	10
7	3	9	3	18	7	8	5	10	2	23	11
7	3	9	4	25	6	8	5	14	2	19	12
7	4	10	1	23	3	8	6	6	2	34	17
7	4	10	2	32	4	8	6	6	3	50	25
7	5	1	1	27	9	8	6	9	3	48	29
7	5	1	2	52	11	8	6	9	4	40	32
7	5	6	2	41	8	8	6	12	1	40	31

TABLE A.6.2.4. Costs and Processing Times for the 15-part, 6-operation, 20-tool, 4-machine Problem (continued)

i	j	t	m	c_{ijtm}	p_{ijtm}	i	j	t	m	c_{ijtm}	p_{ijtm}
8	6	12	3	45	30	10	5	7	1	34	15
9	1	1	1	56	23	10	5	7	2	30	21
9	1	1	2	32	24	10	5	8	3	40	24
9	1	5	1	43	20	10	6	2	3	50	25
9	1	5	4	61	28	10	6	20	2	40	21
9	2	2	3	23	19	11	1	18	4	29	12
9	2	7	1	32	15	11	1	20	2	38	15
9	2	7	2	25	20	11	2	3	1	23	21
9	3	9	3	30	27	11	2	3	2	35	18
9	3	9	4	32	30	11	2	5	1	24	20
9	4	1	1	32	35	11	2	5	4	31	24
9	4	1	2	34	24	11	3	2	3	28	15
9	4	20	2	30	34	11	3	9	3	30	24
9	5	18	4	30	19	11	3	9	4	35	30
9	5	19	2	27	14	11	4	4	2	35	33
9	6	1	1	25	10	11	4	4	3	47	41
9	6	1	2	33	15	11	4	7	1	56	44
9	6	2	3	24	21	11	4	7	2	59	50
10	1	4	2	23	10	11	5	8	3	50	51
10	1	4	3	24	14	11	5	11	2	56	48
10	1	6	2	27	15	11	5	11	4	40	45
10	1	6	3	44	18	11	6	12	1	45	40
10	2	2	3	37	21	11	6	12	3	40	35
10	2	8	3	45	27	11	6	14	2	35	30
10	3	4	2	20	13	12	1	4	2	58	21
10	3	4	3	38	15	12	1	4	3	45	28
10	3	9	3	34	28	12	1	7	1	56	14
10	3	9	4	35	31	12	1	7	2	56	19
10	4	11	2	24	22	12	2	1	1	23	10
10	4	11	4	34	24	12	2	1	2	45	8
10	4	12	1	35	20	12	2	4	2	56	5
10	4	12	3	32	18	12	2	4	3	32	6

TABLE A.6.2.4. Costs and Processing Times for the 15-part, 6-operation, 20-tool, 4-machine Problem (continued)

i	j	t	m	c_{ijtm}	P_{ijtm}	i	j	t	m	c_{ijtm}	P_{ijtm}
12	3	8	3	23	10	14	1	1	2	45	15
12	3	11	2	43	5	14	1	3	1	40	17
12	3	11	4	67	3	14	1	3	2	50	21
12	4	12	1	22	14	14	2	5	1	32	19
12	4	12	3	32	17	14	2	5	4	45	24
12	4	17	3	44	19	14	2	8	3	35	30
12	5	2	3	43	20	14	3	9	3	43	32
12	5	20	2	37	24	14	3	9	4	67	35
12	6	4	2	31	25	14	3	10	1	21	27
12	6	4	3	34	29	14	3	10	2	34	15
12	6	9	3	34	31	14	4	7	1	24	20
12	6	9	4	53	37	14	4	7	2	17	15
13	1	2	3	27	25	14	4	13	4	21	19
13	1	7	1	24	20	14	5	2	3	26	14
13	1	7	2	55	21	14	5	5	1	31	29
13	2	8	3	15	14	14	5	5	4	46	31
13	2	10	1	10	19	14	6	4	2	32	18
13	2	10	2	33	21	14	6	4	3	35	12
13	3	10	1	34	17	14	6	8	3	29	19
13	3	10	2	23	29	15	1	4	2	17	11
13	3	15	4	21	31	15	1	4	3	34	31
13	4	4	2	55	34	15	1	7	1	25	20
13	4	4	3	60	38	15	1	7	2	23	19
13	4	6	2	45	40	15	2	2	3	45	35
13	4	6	3	40	44	15	2	5	1	36	21
13	5	3	1	40	45	15	2	5	4	34	28
13	5	3	2	50	42	15	3	1	1	27	15
13	5	14	2	43	39	15	3	1	2	56	19
13	6	7	1	46	33	15	3	3	1	27	15
13	6	7	2	42	38	15	3	3	2	55	27
13	6	20	2	45	41	15	4	9	3	23	15
14	1	1	1	21	10	15	4	9	4	18	9

TABLE A.6.2.4. Costs and Processing Times for the 15-part, 6-operation, 20-tool, 4-machine Problem (continued)

i	j	t	m	c_{ijtm}	p_{ijtm}	i	j	t	m	c_{ijtm}	p_{ijtm}
15	4	11	2	18	9	15	5	17	3	46	17
15	4	11	4	44	17	15	6	18	4	38	22
15	5	14	2	25	12	15	6	20	2	33	25

REFERENCES

1. Kusiak, A., "Flexible Manufacturing Systems: A Structural Approach, " *International Journal of Production Research*, Vol. 23, No. 6, pp. 1057-1073, 1985.

2. Das, S. R. and B. M. Khumawala, "Flexible Manufacturing Systems: A Production Management Perspective, " *Production and Inventory Management Journal*, Second Quarter, pp. 63-67, 1989.

3. Co, H. C., J. S. Biermann and S. K. Chen, " A Methodical Approach to the Flexible Manufacturing System Batching, Loading and Tool Configuration Problems, " *International Journal of Production Research*, Vol. 8, No. 12, pp. 2171-2186, 1990.

4. Liang, M. and S. P. Dutta, " An Integrated Approach to the Part Selection and Machine Loading Problem in a Class of Flexible Manufacturing System, " *European Journal of Operational Research*, Vol. 67, pp. 387-404, 1993.

5. Hwan, S. S. and A. W. Shogan, "Modelling and Solving an FMS Part Selection Problem, " *International Journal of Production Research*, Vol. 27, No. 8, pp. 1349-1366, 1989.

6. Shanker, K. and Y.-J. J. Tzen, " A Loading and Dispatching Problem in a Random Flexible Manufacturing System, " *International Journal of Production Research*, Vol. 23, No. 3, pp. 579-595, 1985.

7. Sarin, S. C. and C. S. Chen, " The Machine Loading and Tool Allocation Problem in a Flexible Manufacturing System, " *International Journal of Production Research*, Vol. 25, No. 7, 1081-1094, 1987.

8. Kirkavak, N. and C. Dinçer, "Analytical Loading Models in Flexible Manufacturing Systems, " *European Journal of Operational Research*, Vol. 71, pp. 17-31, 1993.

9. Chen, Y.-J. and R. G. Askin, "A Multiobjective Evaluation of Flexible Manufacturing System Loading Heuristics, " *International Journal of Production Research*, Vol. 28, No. 5, pp. 895-911, 1990.

10. Stecke, K. E., " Formulation and Solution of Nonlinear Integer Planning Problems for Flexible Manufacturing Systems, " *Management Systems*, Vol. 29, No.3, 1983.

11. Kim, Y.-D. and C. A. Yano, "Heuristic Approaches for Loading Problems in Flexible Manufacturing Systems, " *IIE Transactions*, Vol. 25, No. 1,pp.26-39, January 1993.

12. Sodhi, M. S., R. G. Askin and S. Sen, " Multiperiod Tool and Production Assignment in Flexible Manufacturing Systems, " *International Journal Production Research*, Vol. 32, No. 6, pp. 1281-1294, 1994.

13. Ram, B., S. Sarin and C. S. Chen, "A Model and a Solution Approach for the Machine Loading and Tool Allocation Problem in a Flexible Manufacturing System, " *International Journal of Production Research*, Vol. 28, No. 4, pp. 637-645, 1990.

14. Leung, L. C., S. K. Maheshwari and W. A. Miller, " Concurrent Part Assignment and Tool Allocation in FMS with Material Handling Considerations, " *International Journal of Production Research*, Vol. 31, No. 1, pp. 117-138, 1993.

15. Mukhopadhyay, S. K., S. Midha and V. M. Krishna, " A Heuristic Procedure for Loading Problems in Flexible Manufacturing Systems, " *International Journal of Production Research*, Vol. 30, No. 9, pp. 2213-2228, 1992.

16. Sabuncuoğlu, I. and D. L. Hommertzheim, "Dynamic Dispatching Algorithm for Scheduling Machines and Automated Guided Vehicles in a Flexible Manufacturing System, " *International Journal of Production Research*, Vol. 30, No. 5, pp. 1059-1079, 1992.

17. Mukhopadhyay, S. K., B. Maiti and S. Garg, " Heuristic Solution to the Scheduling Problems in Flexible Manufacturing Systems, " *International Journal of Production Research*, Vol. 29, No. 10, 2003-2024, 1991.

18. Ulusoy, G. and Ü. Bilge, "Simultaneous Scheduling of Machines and Automated Guided Vehicles, " *International Journal of Production Research*, Vol. 31, No. 12, pp. 2857-2873, 1993.

19. Moreno, A. A. and F.-Y. Ding, "A Constructive Heuristic Algorithm for Concurrently Selecting and Sequencing Jobs in an FMS Environment, " *International Journal of Production Research*, Vol. 31, No. 5, pp. 1157-1169, 1993.

20. Hirabayashi, N. H. Nagasawa and N. Nishiyama, " A Decomposition Scheduling Method for Operating Flexible Manufacturing Systems, " *International Journal of Production Research*, Vol. 32, No. 1, pp. 161-178, 1994.

21. Kim, Y.-D., "Heuristics for Flowshop Scheduling Problems Minimizing Mean Tardiness, " *Journal of Operational Research Society*, Vol. 44, No. 1, pp. 19-28, 1993.

22. Aanen, E., J. Gaalman and W. M. Nawijn, "A Scheduling Approach for a Flexible Manufacturing System, " *International Journal of Production Research*, Vol. 31, No. 10, pp. 2369-2385, 1993.

23. Stecke, K. E. and J. J. Solberg, "Loading and Control Policies for a Flexible Manufacturing System, " *International Journal of Production Research*, Vol. 19, No. 5, pp. 481-490, 1981.

24. Sabuncuoğlu, I. and D. L. Hommertzheim, "Experimental Investigation of FMS Machine and AGV Scheduling Rules against the Mean Flow-Time Criterion, " *International Journal of Production Research*, Vol. 30, No. 7, pp. 1617-1635, 1992.

25. Yim, D.-S. and R. J. Linn, "Push and Pull Rules for Dispatching Automated Guided Vehicles in a Flexible Manufacturing System, " *International Journal of Production Research*, Vol. 31, No. 1, pp. 43-57, 1993.

26. Denzler, D. R. and W. J. Boe, "Experimental Investigation of Flexible Manufacturing System Schduling Decision Rules, " *International Journal of Production Research*, Vol. 25, No. 7, pp. 979-994, 1987.

27. Egbelu, P. J. and J. M. A. Tanchoco, "Characterization of Automatic Guided Vehicle Dispatching Rules, " *International Journal of Production Research*, Vol. 22, No. 3, pp. 359-374, 1984.

28. Schneeweiss, C., " Hierarchical Structures in Organisations - A Conceptual Framework, ", *EIASM Workshop on Hierarchical Planning in Organizational Design*, Brussels, March 1994.

29. Bitran, G.R., E.A. Haas and A.C. Hax, "Hierarchical Production Planning: A Single Stage System, " *Operations Research*, Vol. 29, No. 4, pp. 717-742, 1981.

30. Barbarosoğlu, G., "Hierarchical Production Planning, " *EIASM Workshop on Hierarchical Planning in Organizational Design* , Brussels, March 1994.

31. Axsäter, S., "Aggregation of Product Data for Hierarchical Production Planning, " *Operations Research*, Vol. 29, No.4, pp. 744-756, 1981.

32. Axsäter, S. and H. Jönsson, "Aggregation and Disaggregation in Hierarchical Production Planning, " *European Journal of Operational Research*, Vol. 17, pp. 338-350, 1984.

33. Axsäter, S., "On the Feasibility of Aggregate Production Plans, " *Operations Research*, Vol. 34, pp. 796-800, 1986.

34. Fisher, M. L. , "The Lagrangian Relaxation Method for Solving Integer Programming Problems, " *Management Science*, Vol. 27, No. 1, pp. 1-18, 1981.

35. Geoffrion, J.L., "On the Convergence Rates of Subgradient Optimization Methods, " *Mathematical Programming*, Vol. 13, pp. 329-347, 1977.

36. Shapiro, J.F., "Generalized Lagrange Multipliers in Integer Programming, " *Operations Research*, Vol. 19, pp. 68-79, 1971.

37. Fisher, M.L., "Optimal Solution of Scheduling Problems Using Lagrange Multipliers: Part I, " *Operations Research*, Vol. 21, pp. 1114-1127, 1973.

38. Held, M., P. Wolfe and H.D. Crowder, "Validation of Subgradient Optimization, " *Mathematical Programming*, Vol. 6, pp. 62-88, 1974.

39. Gunasekaran, A., T. Mantikainen and P. Yli-Olli, "Flexible Manufacturing Systems: An Integration for Research and Applications, " *European Journal of Operational Research*, Vol. 66, pp. 1-26, 1993.

40. Hedin, S. R. M. K. Malhotra and P. R. Philipoom, "Shop Floor Control and Tool Loading Policies in Flexible Manufacturing Systems, " *International Journal of Production Research*, Vol. 32, No. 11, pp. 2495-2512, 1994.

REFERENCES NOT CITED

Balasubramanian, R,. S. Sarin and C.S. Chen, "A Model and a Solution Approach for the Machine Loading and Tool Allocation Problem in a Flexible Manufacturing System, " *International Journal of Production Research*, Vol. 28, No. 4, pp. 637-645, 1990.

Barrett, R. M. and S. N. Kadıpaşaoğlu, " Dispatching Rules for a Dynamic Flow Shop, " *Production and Inventory Management Journal*, First Quarter, pp. 54-58, 1990.

Bayburan, B., *Turbo Pascal*, Beta Basım Yayım Dağıtım A.Ş., 1990.

Byrkett, D. L., M. H. Ozden and J. M. Patton, "Integrating Flexible Manufacturing Systems with Traditional Manufacturing, Planning and Control, " *Production and Inventory Management Journal,* Third Quarter, pp. 15-20, 1988.

Garetti, M., A. Pozzetti and A. Bareggi, "On-Line Loading and Dispatching in Flexible Manufacturing Systems, " *International Journal of Production Research,* Vol. 28, No. 7, pp.1271-1292, 1990.

Greene, T. J. and R. P. Sadowski, "A Miexed Integer Program for Loading and Scheduling Multiple Flexible Manufacturing Cells, " *European Journal of Operational Research,* Vol. 24, pp. 379-386, 1986.

Kim, Y.-D., "A Study on Surrogate Objectives for Loading a Certain Type of Flexible Manufacturing Systems, " *International Journal of Production Research,* Vol. 31, No. 2, pp. 381-392, 1993.

Kim, Y.-D. and C. A. Yano, "A Due Date-Based Approach to Part Type Selection in Flexible Manufacturing Systems, " *International Journal of Production Research,* Vol. 32, No. 5, pp. 1027-1043, 1994.

Newhauser, G. L. and L. A. Wolsey, *Integer and Combinatorial Optimization*, New York: John Wiley & Sons, Inc., 1988.

O'Grady, P. J. and U. Menon, "Loading a Flexible Manufacturing System, " *International Journal of Production Research*, Vol. 25, No. 7, pp. 1053-1068, 1987.

O'Keefe, R. M. and T. Kasirajan, "Interaction between Dispatching and Next Station Rules in a Dedecated Flexible Manufacturing System, " *International Journal of Production Research*, Vol. 30, No. 8, pp. 1753-1772, 1992.

Programming in Progress-Student Guide, Progress Software Corporation, 1991.

Progress Language Refernece, Progress Software Corporation, 1992.

Progress Language Tutorial, Progress Software Corporation, 1990.

Progress Programming Handbook, Progress Software Corporation, 1992.

Ro, I.-K. and J.-I. Kim, "Multi-Criteria Operational Control Rules in Flexible Manufacturing Systems (FMSs), " *International Journal of Production Research*, Vol. 28, No. 1, pp. 47-63, 1990.

Taha, H., *Operations Research*, New York: Macmillan Publishing Company, 1982.

Warnecke, H. J., R. Steinhilper and H.-P. Roth, "Developments and Planning for FMS-Requirements, Examples and Experiences, " *International Journal of Production Research*, Vol. 24, No. 4, pp. 763-772, 1986.