2001

Deadlock resolution in flexible manufacturing systems: A Petri nets based approach.

Tarek Younis. ElMekkawy
University of Windsor

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

Deadlock Resolution in Flexible Manufacturing Systems: A Petri Nets Based Approach

By

Tarek Younis ElMekkawy

A Dissertation
Submitted to the Faculty of Graduate Studies and Research
Through the Program of Industrial and Manufacturing Systems Engineering
in Partial Fulfillment of the Requirements for
the Degree of Doctor of Philosophy at the
University of Windsor

Windsor, Ontario, Canada

2001

ABSTRACT

Flexible Manufacturing Systems (FMSs) are characterized by concurrency, resource sharing, routing flexibility, limited buffer sizes, and variety of lot sizes. The sharing of resources and the limitations on buffer sizes may lead to deadlock situations. One of the most challenging problems in FMSs design and operation is to assign the shared resources to jobs efficiently and without causing deadlocks.

To date, little has been done to achieve deadlock-free scheduling in FMSs. In this research a new efficient scheduling algorithm for finding an optimal or near-optimal deadlock-free schedule was developed based on the depth-first and backtracking search technique. Two efficient truncation techniques and three heuristic functions were developed and tested using several randomly generated case studies.

The performance of flexible manufacturing systems that exhibits deadlocks was analyzed under different levels of routing flexibility and other factors using Petri Nets. It was expected that routing flexibility would complicate the Petri Net model and create new deadlocks, which in turn could negatively affect the system performance. The results showed that increasing routing flexibility improves the system performance, measured by average flow time, in systems exhibiting deadlocks.

A novel heuristic deadlock-free rescheduling algorithm based on Petri Nets was developed in order to deal with machine breakdowns in real-time. It guarantees a deadlock-free new schedule and relies on local rather than global rescheduling. The existence of alternative routes, availability of material handling facilities, and the limitations of buffer capacities were considered.

In conclusion, the thesis introduces an integrated approach for production scheduling, control and performance evaluation of flexible manufacturing systems that exhibit deadlocks. The first part takes care of optimizing the performance of the manufacturing system, by generating optimal or near optimal schedules, and avoiding the deadlock situations in the same time. The second part could be used in answering the questions of the what-if analysis. Finally, the third part maintains the production control in real-time.

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to Dr. Hoda A. ElMaraghy for her guidance and continuing support throughout the course of my Ph.D. program. I acknowledge and thank Dr. M. Wang, Dr. S. Taboun, and Dr. A. Asfour for accepting the burden of being part of the supervisory committee.

I would like to extend my gratitude to Dr. Imed Ben Abdallah who introduced me to the field of Petri Net modeling. Sincere thanks to Dr. Hoda A. ElMaraghy and the Office of Graduate Studies at University of Windsor for the financial support they provided during this research work.

I wish to express my utmost gratitude to my parents, wife and children. Without their encouragement and support, I would not have been able to pursue my graduate studies.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Flexible manufacturing systems came into existence as a union of modern computer based technology (CNC machine tools, measuring equipment, industrial robots and so on) and the new methods of production management and control (for instance, group technology, computer aided design and manufacturing, distributed and real-time computer control systems). Compared to conventional manufacturing systems, FMSs gave rise to new and different problems in the course of their design and operation. Problems of FMS can be classified into the following four groups [Banaszak 1991]:

- FMS design problems, including realization of strategic analysis and economic justification as well as determination of facilities design and their layout configuration,
- FMS planning problems, including determination of parts routing, machine loading and machine tooling,
- FMS scheduling problems, including determination of the optimal sequence at which parts are to be input into the system as well as determination of the priorities among the parts, tools, pallets and fixtures waiting for access to each machine tool or the components of materials handling system, and
- FMS control problems, including the workplace and system monitoring (for instance, tool setting, wear and breakage) as well as diagnosis of system components and emergency policies for machine tool breakdown service.

The selection of the modeling tool for an FMS depends on the problem perspective and the required level of details. Therefore, analyzers should give a lot of

attention when selecting a modeling tool. In the following section, some of the most known modeling tools will be introduced.

Deadlock is an emerging new problem of FMSs. A deadlock is a state where a set of parts is in "circular wait", i.e. each part in the set waits for a resource held by another part in the same set. One of the most challenging problems in FMSs design and operation is to assign the shared resources to jobs efficiently and without causing deadlocks. This problem can be solved at the different hierarchical levels of FMSs (design, planning, scheduling, and control). The main objective of this research was to resolve the deadlock problem during the scheduling and control stages.

1.1 MODELING TOOLS

The growth in the complexity of modern industrial systems, such as production, process control, communication systems, etc., creates numerous problems for their developers. The high capability of these systems represents a challenge for their analyzers at the different stages. In view of the complex nature of modern industrial systems, the design and operation of these systems require modeling and analysis in order to select the optimal design alternative, and operational policy. Therefore, special care must be paid for the modeling method for the different planning levels. In this section, the most known modeling techniques for FMS will be presented.

1.1.1 ANALYTICAL TECHNIQUES

Analytical techniques including heuristic methods as well as operations research techniques such as linear programming, queuing networks, branch-and-bound, and dynamic programming have been used to obtain optimal solutions.

Queuing networks among the modeling techniques applied to FMS description, optimization and control. Among the analytical models, queuing networks attract most attention. They consist of the following four components: a source population of customers, customer arrivals pattern, queue discipline, and the service mechanism.

Queuing networks provide information about the average system behavior observed over a long time-period and are useful for quantitative answers to the above mentioned problems, rather than to study control issues. These models are very useful at the preliminary design stage when it is desirable to determine the main features of the system. The exact analytical solutions, using queuing network, for complex systems are still limited to those that satisfy the following conditions: transition probabilities are fixed, workstations have limited buffers, and workstations have exponential service time distribution, or contain servers more than the customers. In spite of slight variation in processing time, FMSs are deterministic and so queuing models are very much approximate. Consequently, FMS models based on queuing networks do not provide any insight about the way the FMS should be controlled in order to achieve some desired performance.

Up-to-date, a number of operations research models have been applied to flexible manufacturing system design and operation. However, not all of them are used in

practice. The reason is that some of the models, especially the combinatorial models, are difficult to solve.

1.1.2 SIMULATION MODELING

Computer modeling and simulation designates the complex activities associated with constructing models of real world systems and simulating them on computer. A simulation is usually defined as the representation of the dynamic behavior of a modeled system.

The advantage of simulation is its capability of representing the most details of the manufacturing systems. They are able to capture all details of system operation, processing requirements and resources, i.e. the characteristics of jobs, machine behavior, work flow and job routing as well as complex control and sequencing rules. However, this ability to capture the complexity of the system has disadvantages. Each simulation run can be lengthy and expensive, therefore preventing the analyst from trying a wide range of parameter values.

Because of the complexity of the simulation model, the time to develop and validate it can be very long. Besides, the simulation is generally developed by experts in computer programmer rather than experts in manufacturing so there is often difficulty in ensuring that the model is an accurate representation of reality. Regarding the FMS, the simulation can help in two issues, as a design tool for FMS, and as a decision support tool to aid the operation of FMS. In addition, Simulation can help defining the relative locations of the machines, work-in-progress (WIP) philosophy, and scheduling/sequencing strategy.

1.1.3 PETRI NETS

Modern manufacturing systems are parallel and distributed. They need to be analyzed from qualitative and quantitative points of view. Qualitative analysis looks for properties like the absence of deadlocks, the absence of overflows, or the presence of certain mutual exclusions in the use of shared resources. Its ultimate goal is to prove the correctness of the modeled system. Quantitative analysis looks for performance properties like throughput, average flow time, average queue lengths, or utilization rates.

The modeling problem in FMS is characterized by concurrent and synchronous events that are typical for such discrete event dynamic systems. Petri Nets are well suited for modeling manufacturing systems because they capture the precedence relations and interactions among these events. In addition, a strong mathematical foundation exists for describing these nets. This allows a qualitative analysis of such properties like deadlock, conflict, and boundedness.

The Petri Net model can also be used as the basis of a real-time controller for a manufacturing system. The flow of tokens through the net establishes the sequence of events to carry out a specific task, such as the manufacturing of particular part type. Petri Net controllers have been used in factories in Japan and Europe.

Petri Nets are also very valuable performance analysis tools. When time is added to the firing of the transitions, it becomes possible to calculate temporal measures that analyze the merit of a particular production system. If time is allowed to be a random variable described by an exponential distribution, then these nets are referred to as stochastic nets. It can be shown that these nets are isomorphic to Markov chain, but much more compact in the representation of the system. The compactness allows for the

handling of systems with a large number of state (on the order of thousands) and the equivalence with Markov chains permits the calculation of performance measures like throughput, average flow time, average machine utilization, probability that the machine blocked or starved, etc.

As a single representation tool, Petri Nets can aid in modeling, analysis, validation, verification, simulation, scheduling, and performance evaluation at design stage. Once the system shows desirable behavior, the net can be converted into control and monitor operations at run time. Therefore, Petri Nets can be regarded as a powerful mathematical and graphical tool for design of various discrete event systems.

1.2 DEADLOCK RESOLUTION IN FMSs

The deadlock problems occur in a system with shared resources. In an FMS, machines, buffers, and automated guided vehicles (AGVs) are some of the shared resources. A set of resources is in a state of deadlock when any job movement to and from these resources is inhibited. In other words, a deadlock is a state where a set of parts are in "circular wait", i.e. each part in the set waits for a resource held by another part in the same set. A system may still process some jobs, even if two or more resources within the system are deadlocked. However, a deadlock usually propagates to other resources in the system, resulting in a complete system deadlock.

Despite the existence of many articles on FMSs, few articles specifically address deadlock problems in FMSs. However, in the area of computer science and operating systems, the deadlock problem has been studied extensively. Coffman et.al. (1971) stated that the following four conditions must be present simultaneously for a deadlock to occur

among concurrent processes: mutual exclusion, wait-for condition, no preemption, and circular wait. A deadlock can be resolved by relaxing any of these four necessary conditions. Unfortunately, the models developed for computer systems may not be directly applicable to an FMS [Minoura and Ding (1991)] because there are many differences between a computer system and a manufacturing system. For example, once a deadlock has been detected in a computer system, one job can be asked to release the resources it is holding and wait without any additional resources while other jobs are served. However, for a machine to serve other jobs, the current job must be sent to an extra resource for temporary holding before it serves other jobs. Also, in a computer system, resources are assigned fractionally and multiple jobs can share a resource simultaneously. However, in a manufacturing system, a resource can usually be assigned to only one job at a time.

There are three approaches used to deal with the deadlock problem in manufacturing systems, namely; deadlock prevention, deadlock avoidance, and deadlock detection and recovery.

The deadlock prevention algorithms precondition the system to remove any possibility of deadlock through predetermined rules. A major advantage of deadlock prevention algorithms is that they are simple and require no run-time cost, since problems are solved in system design or planning stages. Most of the deadlock prevention algorithms impose constraints on the ways in which processes request resources to prevent deadlock. Restricting resources requesting may cause inefficient system performance. Kundu and Akyildiz (1989) proposed a model to decide the amount of buffer spaces required to prevent deadlocks in a system based on closed queuing

networks. However, this model is more appropriate for use at the design level rather than at the scheduling and control levels.

The deadlock avoidance algorithms dynamically examine the system state, and when a deadlock is approached, it is carefully avoided. The goal of deadlock avoidance algorithm is to impose less strict conditions than the deadlock prevention algorithms, hoping to achieve better resource utilization. Major disadvantages of deadlock avoidance algorithms are that the future resource requirements must be known [Isloor and Marsland (1980)] and since predicting possible future deadlocks sometimes requires an exhaustive search, and run-time cost is high.

Deadlock detection and recovery algorithms presume deadlocks can occur. Their goal is to detect the occurrence of a deadlock and to clear deadlock conditions from the system by a recovery algorithm. The performance of this approach depends on the speed of detecting the deadlock situations, the time required to do the recovery action, and the frequency of deadlock occurrence.

Barkaoui, and Alloua (1997) evaluated, using simulation, the performances of two strategies: the first is a detection/recovery method based on digraph representation; the second is a prevention method based on Petri Nets structure theory. Their results confirmed the idea that the detection/recovery methods are more efficient than prevention or avoidance methods only if the time corresponding to their recovery is negligible relative to processing time. In addition, the authors concluded that this condition is unrealistic in FMS environment and consequently, the deadlock detection/recovery strategies seem to be not appropriate in this context.

1.3 THESIS OVERVIEW

The main objective of this work has been to investigate the deadlock resolution in FMSs while optimizing the performance of the system off-line and on-line. The advantage of this approach of deadlock resolution is that it does not restrict the system flexibility. In addition, it does not impose complexity on the system control.

In this research, the work was divided into three parts. In the first part, some heuristic algorithms were developed to find an off-line optimal (or near -optimal) deadlock-free schedule in FMSs. In the second part, the effect of routing flexibility on the performance of systems that exhibits deadlocks was analyzed using factorial design and ANOVA methods. In the third part, a rescheduling algorithm was developed to deal with machine breakdowns and resolve the deadlock problem in real-time. This thesis is organized as presented below.

Chapter 2 reviews the literature of deadlock resolution in FMSs, real-time scheduling in FMSs with concentration on deadlock resolution, research work related to manufacturing systems flexibility. The research needs in these directions were highlighted. Finally, the motivation and objectives of the research were described.

Chapter 3 introduces Petri Nets as powerful tools for modeling and optimizing the scheduling of manufacturing systems. In addition, some truncation techniques were presented as a means for limiting the search space when Petri Nets are used in scheduling optimization.

Chapter 4 introduces a search algorithm that finds an off-line optimal (or near -optimal) deadlock-free schedule in FMSs. The algorithm was based on the depth-first principle and backtracking. In this chapter a performance evaluation of the developed

search algorithm extended with truncation techniques and heuristics, was conducted. Finally, three heuristic functions, for optimizing average flow time, were introduced and their performance was tested.

Chapter 5 was devoted to the performance of flexible manufacturing systems under different levels of routing flexibility. The effect of other factors such as system loading, probability of machine breakdowns and processing time variability were considered. The factorial design of experiments was introduced and ANOVA method was used for analysis.

Chapter 6 presents deadlock-free rescheduling in FMSs. An algorithm that deals with machine breakdowns and guarantees deadlock-free scheduling in real time was introduced. The developed algorithm takes into consideration the existence of alternative routes and the limitation on material handling and buffers capacity.

Chapter 7 exhibits the research summary and conclusions, and recommendation for future research.

CHAPTER 2

LITERATURE REVIEW

The concept of flexible manufacturing systems (FMSs) has introduced significant improvements in system performance compared to traditional manufacturing systems. To gain these improvements, the problems of FMSs at design, production planning, scheduling, and control levels should be critically addressed before system installation. In the literature [Suri (1985)] a variety of approaches (analytical, heuristic, and simulation) have been proposed to deal with FMSs problems at the different levels. Buzacott (1985), and Buzacott and Yao (1986) reviewed the planning models in FMSs. The scheduling problems of FMSs have been extensively reviewed in the literature [Gupta et al. (1989), Gupt et al. (1991), and Stecke and Rachimadugue (1994)].

The deadlock is an emerging new problem of FMSs, which attracted the attention of the researchers. This problem can be solved at the different hierarchical levels of FMSs (design, planning, scheduling, and control). The main objective of this research is to resolve the deadlock problem during the scheduling and control stages. Furthermore, the performance of systems exhibiting deadlocks will be analyzed under different routing flexibility levels.

In section 2.1, the literature of deadlock resolution in FMSs is reviewed. Section 2.2 is devoted to literature review of FMS real-time scheduling with concentration on deadlock resolution. Section 2.3 introduces the flexibility of manufacturing systems and its effect on the system's performance. Finally, the research motivation and objectives are highlighted in sections 2.4 and 2.5.

2.1 DEADLOCK RESOLUTION IN FMS

By imposing restrictions on the buffers capacity Kundu and Akyildiz (1989) proposed a model to decide the amount of buffer spaces required to prevent a deadlock in a system based on closed queuing networks. This model is more appropriate for use at the design level. Banazak and Krogh. (1990) used the same approach. They have developed deadlock avoidance algorithms for FMSs using the PN. One of their algorithms is for the buffer management in FMSs to avoid deadlocking. The algorithm uses a restriction policy on buffer allocation to avoid deadlocks.

Reveliotis (2000) has undertaken an analytical investigation of optimally selecting the deadlock resolution strategy for buffer space allocation in flexible automated production systems. His analytical methodology was based on Markovian chain, with the objective of maximizing the steady state system throughput. The results indicated that the optimal selection scheme switches between detection and recovery and the pure deadlock avoidance based on the threshold level of the time of the deadlock recovery.

Ferrarini et al. (1999) addressed the problem of evaluating and comparing the performance of deadlock avoidance, prevention and recovery control policies applied to FMS. They discussed the problem for untimed and timed models. Different control algorithms, adopted from the literature, have been considered. In addition, some indices were proposed to assess the performance of the FMSs under the different control policies.

Zuberek and Kubiak (1999) showed that a large class of flexible manufacturing cells can be modeled using timed Petri nets. Two complementary approaches to analysis of such models are presented: invariant analysis and throughput analysis. Invariant analysis provides analytic (or symbolic) solutions for the cycle time of a cell analyzing

subnets of the original net. Throughput analysis performs a series of performance-preserving net reductions to simplify the original model. The authors did not consider the deadlock problem in their model.

Abdallah and ElMaraghy (1998-a) developed a deadlock prevention and avoidance methods based on the structure theory of Petri nets for a class called Systems of Sequential Systems with Shared Resources (S^4R). Abdallah and ElMaraghy (1998-b) developed a methodology to design an efficient deadlock-free schedule for FMS modeled by S^3PR net. The proposed methodology was composed of two steps. First a resource allocation policy was added off-line to an S^3PR net to prevent deadlocks. Second, a search algorithm was applied to the net resulting from first step in order to identify a near optimal schedule for the FMS.

Lawley et al. (1997) addressed the design of control policies for allocating buffer space to competing parts in an FMS with avoiding deadlocks. They assumed that each machine has a finite amount of buffer capacity. They developed the criteria that real-time FMS deadlock-handling strategies must satisfy. These criteria are based on a digraph representation of the FMS state space. The scheduling optimization was not considered in this paper.

Ezpeleta et al. (1995) used the structure theory of Petri nets to develop a deadlock prevention policy in FMS modeled by a class of Petri nets called S^3PR. Barkaoui and Abdallah (1995) developed a deadlock prevention method for the same class of Petri nets by using the structure theory of Petri nets.

Deadlock prevention and avoidance methods have been developed for some classes of FMS modeled by subclasses of Petri nets such as the linear manufacturing line

(LML) [Minoura and Ding (1991)]. Production Petri Nets [Banazak and Krogh (1990). Xiong and Chen (1995)]. and Systems of Simple Sequential Processes with Shared Resources (S^3PR) [Barkaoui and Abdallah (1995)].

Viswandaham et al. (1990) proposed a deadlock avoidance model based on the Petri net reachasbility analysis. In the model. the current state of the system was classified into three states: deadlocked. blocked. and safe. Using reachability analysis. possible future states were anticipated. The number of steps to which the procedure looks ahead is a user defined input and it determines how far ahead the procedure looks from current state. The more the number of "look ahead steps". the greater the possibility of avoiding future deadlocks. but the longer the response time. For larger real world FMSs. they recommended using this avoidance model in conjunction with other deadlock recovery approaches.

Wysk et al. (1991) proposed a scheme of deadlock detection based on a system status graph and an application of the string multiplication algorithm. They investigated the method applied in flexible manufacturing cells (FMCs) that had no internal buffer storage. However. they did not consider multiple routings of a job. Wysk et al. (1994) extended their work by developing another approach of deadlock avoidance. A simulation study was conducted to compare different deadlock resolution approaches. Their results revealed that when handling time is relatively high compared to average processing time. the avoidance and recovery approaches should be applied. On the other hand. the deadlock prevention approach is appropriate choice for manufacturing systems with relatively small handling times.

Although a considerable amount of interest has been arisen in resolving the deadlock problem of FMSs, a limited research has been done in the area of FMSs scheduling optimization with deadlock-freeness. To the knowledge of the author, the pioneers of research in this direction are Ramaswamy and Joshi (1996), Chen and Jeng (1995), Xiong and Zhou (1997), and ElMekkawy et al. (1998).

Jeng and Chen (1998) developed a heuristic algorithm based on the best-first search technique and the analytic theory of timed Petri Net for scheduling in FMSs. Their goal was to minimize the makespan. Heuristic functions, based on the state equation of the timed Petri Net model, were used to predict the total makespan from the initial state through the current state to the goal state. This heuristic approach may require extensive computational effort for large systems. Furthermore, the authors did not consider the deadlock problem.

Xiong and Zhou (1997) proposed an algorithm based on the best-first and backtracking techniques for deadlock-free scheduling (minimizing the makespan) in automated manufacturing systems. Their algorithm searched the RG of the timed Petri Net, and when a deadlock marking was reached; it looked for another marking for further exploration. They used a heuristic function similar to the second one used by Lee and DiCesare (1994).

Ramaswamy and Joshi (1996) developed a mathematical programming formulation for deadlock-free schedules in automated manufacturing systems. Their goal was to minimize the total or average flow time. In their work, they showed the importance of considering material handling and limited buffers capacity when tackling the deadlock problem. First, they optimized the schedule, then used a heuristic algorithm

to include the material handling. They used a Lagrangian relaxation heuristic to simplify the models in order to deal with larger systems.

Lee and DiCesare (1994) developed a heuristic search algorithm based on the A* search technique and Petri Nets. The search algorithm seeks optimal or near-optimal schedules (makespan) by searching only the necessary portion of the PN reachability graph (RG). They introduced three heuristic functions to limit the search space instead of searching the whole reachability graph. The first function considers the depth of the markings and favors the marking that is deeper in the reachability graph to reach the final marking. The second function is based on estimating the minimum remaining operation time and favors a marking hat has an operation ending soon. The third one is combination of both of the first two functions.

2.2 REAL-TIME SCHEDULING/RESCHEDULING

The first step of the production scheduling is determining an initial schedule over a certain time horizon. In a dynamic environment, this schedule is subject to uncertainties such as machine breakdowns, rush order arrival, orders cancellation, change in due dates, and materials shortage. Therefore, the need for a real-time scheduling (rescheduling) in this environment becomes mandatory.

Rescheduling implies that a new schedule is generated from an existing schedule upon an occurrence of some uncertainty. Rescheduling can be done manually by revising the original schedule. This method is not practical for complex systems especially in automated manufacturing systems. In addition, rescheduling can be done using computers methods. The most popular computer methods are electronic Gantt chart, simulation,

expert systems. and hybrid approaches. Some of the general surveys published in the area of real-time scheduling of FMS are Harmonosky and Robohn (1991). Basnet and Mize (1994). Shukla and Chen (1996). A good survey of articles involving the use of simulation in real-time scheduling is presented in Harmonosky (1995). Szelke and Kerr (1994) provided an overview of research results in the domain of knowledge-based reactive scheduling. Hybrid systems are receiving greater attention from researchers because they seem to overcome the limitations of each other at the same time gain the strength from each other. Gonzalez (1995) reviewed the various ways in which on-line knowledge-based simulation for FMS has been approached.

2.2.1 HEURISTIC APPROACHES

Piramuthu et al. (2000) presented an adaptive scheduling policy for dynamic manufacturing system scheduling using information obtained from snapshots of the system at various points in time. The framework presented allows for information-based dynamic scheduling where information collected about the system is used to optimize the scheduling using genetic algorithms. Experimental studies indicated the superiority of the suggested approach over the alternative approach involving the repeated application of a single dispatching rule for randomly generated test problems.

Lin (2000) presented an approach that contains an alternative dynamic controller with a manufacturing system capability database. An initial schedule is generated whenever an unexpected perturbation is identified from the shop floor. A real-time procedure with a conflict-resolution mechanism then drives the control engine for alternative machines. Simulation experiments were also performed to evaluate the

approach with four comparable models. Simulation results demonstrated improvement in the mean flow time and makespan.

Jain and ElMaraghy (1997) developed local rescheduling algorithms that generate a new schedule without re-evaluating all tasks in the original schedule. These algorithms use the system status as input and reschedule the tasks when disturbance occurs. They considered four types of disturbances, these include machine breakdowns, increased order priority, rush order arrival, and order cancellations.

Li et al. (1993) proposed a rescheduling algorithm based on the construction of a scheduling binary tree and a net change concept adopted from MRP systems. A limitation of the algorithm is that it could only deal with a rescheduling situation that assumes no change in the existing operation sequence of each machine. They did not consider the alternate routings for rescheduling.

Yamamoto and Nof (1985) used a regeneration method in developing their rescheduling systems. This method involves rescheduling the entire set of operations, including those unaffected by the disturbance (uncertainty). This method response time is unacceptable for real-time applications especially with moderate or complex systems. In their work, they considered only machine breakdowns as a source of disturbances.

Cott and Marchietto (1985) proposed an on-line scheduling and control mechanism consisting of POMA (Projected Modification Algorithm) and a scheduling algorithm. The system employs POMA to deal with processing variations. The detailed methodology for coordinating the POMA and the scheduling algorithm, such as timing of scheduling modification, range of modification, and so no, was not described.

2.2.2 SIMULATION APPROACH

Burns and Harrison (2000) described the use of parallel multi-population genetic algorithms (GAs) to meet the dynamic nature of job-shop scheduling. A modified genetic technique was adopted by using a specially formulated genetic operator to provide an optimization search. The proposed technique has been implemented using the programming language MATrix LABoratory (MATLAB), providing a tool for job-shop scheduling. Comparisons indicated that the proposed genetic algorithm has successfully improved upon the solution obtained from conventional approaches, particularly in coping with job-shop scheduling.

Gargeya and Deane (1999) presented study of scheduling in the dynamic job shop under auxiliary resource constraints (e.g. Tooling). A global Contingency Based Scheduling (CBS) approach were developed and evaluated in a dynamic job shop constrained by auxiliary resources. The authors defined the CBS as a combination of local and look-ahead dispatching with the resource assignment rules. Some measures of performance were employed, including root mean square of tardiness, average system time and percentage of auxiliary resource changes. As shop utilization increases, the study revealed that the CBS algorithm is the only scheduling mechanism that consistently provides high performance on all three measures compared to simple scheduling rules such as Earliest Due Date (EDD) and Critical Ratio (CR).

Zhang and Chen (1999) reported the development of a dynamic job-scheduling system in the low-volume/high-variety manufacturing environment. Their system is based on a heuristic algorithm that takes into account the influence of machine setup

times. cell changes. replacement machines and load balancing among machines. The algorithm is mainly based on simple priority rules.

Pierce and Yurtsever (1999) presented the development and implementation of Motorola's Graphical Manufacturing Monitoring System (GraMMS) that provides real-time graphical monitoring of manufacturing data including work-in-process (WIP). equipment. throughput. and dispatch rankings. GraMMS consists of four main applications namely; Dynamic Dispatch. WIP Monitoring System (WMS). Equipment Management System (EMS). and Throughput Monitoring System (TMS). GraMMS provides Motorola semiconductor wafer fabs with instant data-driven decision support and trend analysis as well as a short-interval scheduler via discrete-event simulation. The Dynamic Dispatch system is a combination of simulation and heuristics based on concepts from the Theory of Constraints.

Tipi and Bennett (1999) presented a simulation approach for dynamic and stochastic scheduling using ARENA software. The objectives for the first part of this study was to investigate the system behavior in order to meet the following measures of performance: minimum mean flow time. maximum throughput, minimum WIP, and maximum machine utilization. Ten dispatching rules were implemented for scheduling. The target for the second part of study was to implement a look-ahead control technique to maintain WIP to a minimum level while maintaining a high level of throughput. A launch controller was implemented to improve the system behavior.

Merchawi and ElMaraghy (1998) presented an analytical approach to determine the suitability of discrete-event simulation for real-time decision-making in flexible manufacturing systems (FMSs). A formula was developed to predict the simulation CPU

run time for a given manufacturing system, a planning horizon, and a computer system. It was shown that there are only three main factors that affect simulation run time: the planning horizon, the overall system average interarrival time, and the average number of workstations per part routing. An approach to reduce simulation run time was presented. This approach is based on aggregating workstations to reduce the average number of workstations per part routing. A theoretical approximation of the error incurred as a result of aggregations was derived. The results showed that simulation CPU time savings of up to 400% can be achieved as a result . The developed concepts have been integrated in algorithm to serve as bases for discrete-event simulation to adapt simulation models to real-time decision-making requirements.

Smith et al. (1994) used simulation (SIMAN/Arena) as a task generator in addition to an analyzer. A simulation of the physical system acts as a decision maker which determines what task must occur next and sends that task to the execusion software.

Kim and Kim (1994) proposed a real-time scheduling mechanism that consisted of a real-time scheduling module and a simulation module. A rule selector invoked the simulation module at the beginning of a planning horizon or when the system did not perform according to expectations because of disturbances. A series of discrete-event simulations were performed to select a new control rule. The methodology described does not consider resource failures, and is confined to the selection of the best dispatching rule. The authors admitted that their methodology may not work efficiently in systems in which machines are not reliable and urgent orders arrive often.

Harmonosky and Robbon (1991) investigated how physical system parameters affect the amount of time it takes to perform the simulation given a specific level of precision for statisical accuracy of the simulation output. The objective was to develop the profile of the system that would be a good candidate for real- time simulation. Their results suggested that the profile of a system that would be amenable to simulation as a real-time tool would be a system with longer average processing time, a WIP, and flow shop type characteristics, since *CPU* time was shorter with these parameters.

2.2.3 EXPERT SYSTEMS APPROACH

Many researchers are more skeptical that true human experts exist in scheduling [Aytug et al. (1994)]. They believe that expert system approaches are not suitable for scheduling because the complexity of most real-world environment is beyond the cognitive capabilities of most schedulers and that most environments are so dynamic that knowledge becomes obsolete too fast anyway.

Sagi et al. (1994) used a dynamic knowledge base and active databases containing real-time information collected from the shop floor for an intelligent FMS real-time controller supported by a neural network for decision inference. The neural network was trained by an off-line simulation system.

Terpstra et al. (1994) proposed a prototype reactive scheduler, intended to handle a variety of uncertainties, for mixed batch\continuous plants with an expert system. The scheduler consists of three modules: Planner, Integer Scheduler (IS), and Non-Integer Scheduler (NIS). Planner generates all possible production paths to produce products required by production orders in a specific plan. IS reaches optimal production paths and

sequence using branch-and-bound method. NIS optimizes timing and flows within a path by non linear programming. This approach, however, did not show a mechanism to define schedule modification timing and modification area in a dynamic situation, which is critical in on-line scheduling. Moreover, it seems to be inflexible as far as the amount of time available for rescheduling and problem sizes are concerned, because the scheduler reschedules due to any small variation, which requires high computational capabilities for large size problems.

Dutta (1990) proposed a knowledge-based system to help automate the control activity at the scheduling level in FMS environment. The proposed system can react to machine failures, new job release and change in job priority. The variation in the processing time were not taken into consideration. The author assumed a batch size of one which is rarely the case in actual production systems.

Reynolds (1988) described a knowledge-based production scheduler called the master scheduling unit (MSU) used by Westinghouse Electric Corporation. The MSU generates production schedules that reliably fulfill the requirements of a given manufacturing facility and revises and maintains the production schedules in response to changing shop conditions. Each day, the MSU receives a new list of shop orders, information regarding incomplete jobs from the previous day, and the status of machines on the floor. The MSU then constructs a schedule for the day based on the information received. Rescheduling takes place when the priority of a job changes or a machine breakdown occur.

2.2.4 PETRI NETS

Wu (1999) introduced a Petri net model, called colored resource-oriented Petri net (CROPN). The concurrent resource conflict and the characteristics of the production processes necessary for deadlock control were modeled. Based on the developed model, necessary and sufficient conditions and a control law were presented for deadlock-free operation in FMS's. This control determines when a resource can be allocated to which job to avoid deadlock. The author mentioned that the control model can be used in real-time scheduling but did not take into consideration the effect of uncertainties. The control policy presented in this paper allows blocking of machines which results in bad performance of the system. In other words, the author did not consider scheduling optimization beside the deadlock avoidance.

Jeng et al. (1999) presented a heuristic search method based on Petri nets for scheduling flexible manufacturing systems with assembly (FMSA) by partially generating the reachability graph. The Petri nets structure was used to solve their state equations for solutions that constitute a part of the proposed heuristic function. Considering the dynamic information of nets such as concurrency and synchronization, the part of the heuristic function is adjusted since state equation solutions may overestimate the real cost. The adjustment is based on a lower bound of the real cost and on dynamically comparing the partial estimated cost and partial real cost during the search process. The authors did not consider the deadlock problem.

Masory (1994) introduced a framework for integrating simulation; control and real-time diagnostic of manufacturing systems based on adaptive augmented time Petri nets. The system consists of three main modules namely: 1- System Simulation for performance evaluation, 2- System Controller, and 3- Real-time control and diagnostics.

In his system he mentioned how to detect a fault in the system. but not how to develop a corrective action.

Tawegoum et al. (1994) developed a system for real-time monitoring of FMS based on Petri nets. The system consists of two main modules. The first module generates a schedule based on branch and bound method that minimizes transfer and waiting times. The second part is the piloting module, which modifies the jobs routing in case of perturbation. They detect the perturbations by comparing the scheduled dates with the actual observed dates. They did not present an example to test the performance of their system. In addition. they did not consider the details of causes of the perturbation.

Camurri et al. (1993) used Simulation of PN model with some priority rules to develop a sub-optimal schedule. They did not consider the effect of the different disturbances on the real-time execution of the Manufacturing system or develop a look a head window for real-time simulation.

2.3 PERFORMANCE EVALUATION OF MANUFACTURING FLEXIBILITY

Performance of the system can be evaluated at different levels of flexibility to enable the managers to set the amount of flexibility in order to get a reasonable performance. In the literature. there are some attempts to analyze the system performance with varying a certain type of flexibility.

Beach et al. (2000) tried to provide a comprehensive understanding of the manufacturing flexibility for academic and industrial practitioners of this field. A review of the literature was used to examine the issues related to manufacturing flexibility. Among the concepts that were discussed. the use of manufacturing flexibility as a

strategic objective, the relationship of flexibility with uncertainties, the use of taxonomies as a vehicle for furthering understanding of the types of flexibility, and the measurement of flexibility. Through this process of synthesis, the paper attempted to establish the extent to which knowledge of manufacturing flexibility has now progressed. Suggestions for future research topics in flexibility were also presented.

D'Souza and Williams (2000) presented an approach for general taxonomy of manufacturing flexibility dimensions. They proposed a theoretical construct of the manufacturing flexibility. Their model was based on only four dimensions: volume, variety, materials handling, and process. Operational measures of manufacturing flexibility dimensions were identified, from previous literature, and tested on a sample of 240 manufacturing firms. Results indicated good support for the proposed taxonomy.

Zukin and Dalcol (2000) presented a study of how consumer electronics companies in Brazil deal with the issue of manufacturing flexibility. Their main objective was to provide an understanding into how flexibility is being perceived. The paper explained how indicators were established to obtain an analytical structure with which to assess the managerial perception and the actual industrial use of flexibility. In addition, they showed the most relevant results of an investigation of 16 leading firms in Brazil. The findings were divided according to managerial perception of flexibility and effective utilization of flexibility in the organization. The paper concluded that firms in the consumer electronics industry in Brazil do not use flexibility practices in the same proportion that they perceive its importance.

Tsubone and Horikawa (1999) evaluated two types of flexibility, machine flexibility and routing flexibility, in terms of manufacturing performance. A simulation

based investigation was conducted to analyze the impact of these types of flexibility on the average flow time under various job flow pattern conditions, which characterize the shop nature from a random job shop to a flow shop, operation time variance, setup time, and shop load. The experimental results showed the trend of change of the average flow time due to these types of flexibility and which type is superior under what conditions.

Bateman et al. (1999) described a mathematical model for measuring mix response flexibility and its results were compared with a simulation. Experiments were performed on three system types: single machine system, simple multi-machine system, and complex multi-machine system. The response of the system were measured by the mean sensitivity to change. The model was found to give meaningful results for both single machines and multiple machine systems when compared with another simulation method.

Berry and Cooper (1999) studied the impact of product variety on manufacturing systems performance. They showed that adding product variety can have adverse cost and margin implications when marketing and manufacturing strategies were mis-aligned. They reported methods that can be used to measure product mix flexibility and manufacturing performance in terms of costs based on actual orders and production data. They have tested these methods in field research on high volume batch processes that are representative of many firms in process industries. The results showed that gaining competitive advantage through increased product variety requires a clear understanding of the process choice required to support the considered range of product volumes, and the cost and profitability trade-off involved.

Schneeweiss and Schneider (1999) introduced a measure of flexibility as the availability of a dynamic system in an uncertain environment. They extended their measure to incorporate a system's planning, forecasting, and implementation ability. In addition, the authors highlighted the design of flexibility as a hierarchical planning problem.

Mahmoodi et al. (1999) examined the effects of scheduling rules and routing flexibility on the performance of a constrained, random flexible manufacturing system (FMS). Other experimental factors considered were shop load, shop configuration, and system breakdowns. Their results indicated that, in the presence of total routing flexibility, the effects of shop load, system breakdowns, and scheduling rules were significant. In particular, when total routing flexibility exists, the choice of scheduling rules is not critical. In addition, They showed that the behavior of scheduling rules in a more constrained FMS environment (i.e., where system breakdowns occur and material handling capability is limited) was consistent with the findings of previous research conducted under less constrained environments. Finally, results indicated that the shop configuration factor had little or no impact on a system's flow-time performance.

Caprihan and Wadhwa (1997) presented a framework based on the Taguchi experimental design for studying the nature of the impact of varying levels of routing flexibility on FMS performance. Through simulation results, they showed that an increase in routing flexibility, when introduced at the cost of an associated penalty on operation processing time, was not always beneficial. They claimed that there could be an optimal flexibility level beyond which system performance deteriorates, as judged by the makespan measure of performance.

Benjaafer and Ramakrishnan (1996) introduced several representation and measurement schemes for sequencing flexibility. The relationship between flexibility and system performance was used to identify essential characteristics of flexibility measures.

Chen and Chung (1996) suggested some measures for machine and routing flexibility. Design of experiments was conducted to test the effect of machine flexibility and routing flexibility on system performance. The analysis was based on two measures: makespan and system utilization. Their conclusion agrees with that of Benjaafer (1994): FMS performance improvement resulting from an increase in machine and routing flexibility follows the Law of Diminishing Returns.

Benjaafer (1994) addressed the relationship between routing flexibility and manufacturing systems performance. He used the number of alternative machines of each operation as a measure for routing flexibility. Mathematical models, based on the queuing theory, were developed to estimate the average flow time at different routing flexibility levels. It was found that the effect of routing flexibility on performance is of the diminishing return kind. This means that while a small degree of flexibility is highly desirable, the additional benefits due to further increases are only marginal.

Gupta (1993) measured machine flexibility by the number of products (K) that can be processed by the same machine. A mathematical model was developed to obtain the optimal number of machines, the capacity of every machine, and the flexibility level (K) of every machine in order to manufacture a certain product mix with defined volumes. It was assumed that all machines have the same capacity and flexibility level.

Chen and Chung (1991) investigated the relationship between loading and routing decisions along with effect on FMS flexibility and productivity. They developed a

mathematical model for solving the loading problem in FMSs. The model maximizes the total number of operation assignments subject to technological and resource constraints. The output of this model presents feasible routes along which parts can flow through the system. Another mathematical model for routing decisions was developed; its input is the output of the loading model. For every job, the routing model selects the route that optimizes the makespan.

2.4 RESEARCH MOTIVATIONS

Deadlock problems can cause unnecessary costs like, long downtime that leads to under-utilization of some critical and expensive resources. In automated manufacturing systems such as FMSs, deadlocks should be prevented or the system controller should be designed to avoid deadlocks on real-time or equipped with deadlock resolution mechanism. Otherwise, deadlocks must be resolved by human in real-time. Such an occurrence in the system is undesirable since it prohibits the operation of an FMS in an automated mode. In general, deadlocks are difficult to predict. The earlier a deadlock or its possibility in the system is detected, the easier it can be resolved or prevented, and consequently, the lost production is reduced.

From the past research in the scheduling it can be concluded that the research is focused mainly on the optimization without considering the deadlock problem. In other words, the research of optimizing the scheduling of FMS with deadlock avoidance is limited. This research is aimed to resolve the deadlock problem with optimizing the schedule of FMS. The advantage of this approach of deadlock resolution that it does not

restrict the system flexibility, and in addition, it does not impose complexity on the system control.

To my knowledge, there is no attempt to study the effect of flexibility in systems exhibiting deadlocks. We were one of the first groups to study this issue [Abdallah et al. (1998)]. In order to achieve a contribution in this direction, some measures for the considered flexibility type should be used to guide the experiments. In addition, the factors that may affect the system performance in the presence of flexibility should be taken into consideration.

Because of the dynamic nature of the recent manufacturing systems, rescheduling is an essential part of their control. From the above literature review it can be concluded that most of the authors adopt the idea of rescheduling the whole jobs in operation in case of uncertainty occurrence. This concept leads to impractical response time. On the other hand there are some authors who tried to minimize the rescheduling response time by rescheduling a subset of jobs (which are affected by the uncertainty occurrence) of the original schedule rather than changing the whole schedule. Some of these works did not consider other factors such as the routing flexibility, material handling, and limited buffers sizes. Other authors claimed that they reschedule a subset of jobs but they did not mention how to determine this set of jobs. Another point can be concluded from the above literature that the problem of deadlock is ignored in the rescheduling area of research.

2.5 RESEARCH OBJECTIVES

It can be noted from the literature survey that there are two approaches addressing the scheduling problems in FMS. First, authors are working towards developing algorithms to find deadlock-free schedules without optimization. Second, authors are working towards developing algorithms to find optimal or near-optimal schedules assuming that the system will not exhibit deadlocks.

In this research, the work was divided into three parts. In the first part, a heuristic algorithm was developed to find an off-line optimal (or near -optimal) deadlock-free schedule in FMSs. An algorithm was developed in order to optimize the makespan. This algorithm was modified in order to optimize the average flow time. Furthermore, two truncation techniques were developed in the sake of improving the search algorithm performance. The truncation techniques are based on the structure of the Petri net mode and the concepts of scheduling. Three heuristic functions, for optimizing average flow time, were developed to reduce the complexity of the scheduling problem when optimizing the average flow time.

In the second part of the research, the performance of manufacturing systems that exhibit deadlocks was analyzed under different levels of routing flexibility using Petri nets. The effect of other factors such as system loading, probability of machine breakdowns, and processing time variability were considered. The factorial design of experiments was introduced and ANOVA method was used for analysis.

In the third part, a rescheduling algorithm was developed to deal with machine breakdowns in real-time. The functions of this algorithm are to react to disturbances, maintain the superiority of the original schedule, and guarantee the deadlock-freeness.

The developed algorithm takes into consideration two important features of FMSs namely: routing flexibility, and limited material handling facilities. In order to have a reasonable response time during real-time, the rescheduling algorithm should modify the affected subset of the original schedule rather than changing the whole schedule.

The research work of this thesis can be integrated in a system as shown in Figure 2.1. This system can be used in production scheduling and control of FMSs. The figure shows the components of the systems that were addressed in this thesis.

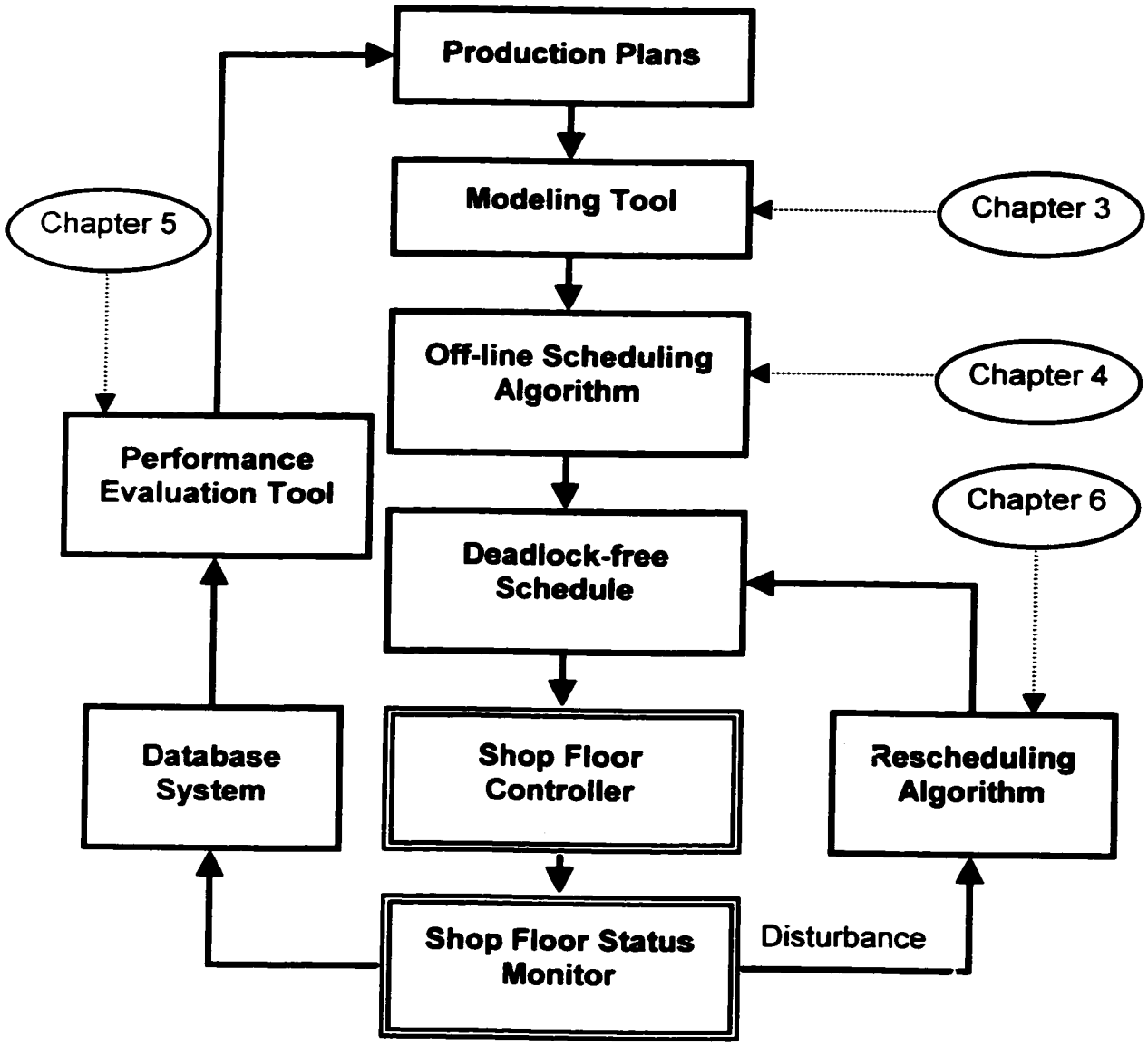


Figure 2.1: Production Scheduling and Control System for FMSs

CHAPTER 3

MODELING FMSs USING PETRI NETS

Concurrent and asynchronous events characterize modeling discrete event dynamic systems. The precedence constraints and the interactions between these events can be modeled easily using Petri Nets (PN). In addition, Petri Nets have a strong mathematical foundation, which enables them to do qualitative and quantitative analysis. For this reason, Petri Nets is one of the most suitable tools for analyzing the deadlock problem of FMSs. Petri Nets are comprehensive and efficient modeling tools that can be used in the following areas for manufacturing systems:

(a) Performance Evaluation: When time is added to the PN model, temporal measures can be analyzed.

(b) Optimization: by analyzing the reachability graph of the PN model the optimal solution path can be obtained.

(c) Real-time Control: the flow of tokens through the net provides the sequence of events to accomplish a certain production plan.

In this chapter, Petri Nets will be presented as powerful tools for modeling and optimizing the scheduling of manufacturing systems.

3.1 TIMED PETRI NETS

Petri Net theory was originally developed by Carl Adam Petri and presented in his doctoral dissertation in 1962. In a Petri Net, conditions are represented by places (circles) and events are represented by transitions (solid bars). The occurrence of a condition is

represented by a token (solid circle) in the place representing this condition. Thus, the marking of a PN indicates the current state of the system. The arcs represent the pre-conditions and post-conditions of an event occurrence.

Standard Petri Nets do not include the concept of time, they are used only for qualitative and a logical analysis of systems. The emergence of Timed Petri Nets (TPN) made it possible to perform quantitative analysis of systems. By associating a constant time delay with either places or transitions, Timed Places Petri Nets (TPPN) and Timed Transition Petri Nets (TTPN) were created. Stochastic Petri Nets (SPN) were proposed because probabilistic performance models allow the analyst to capture the essence of system behavior through probabilistic assumptions so that a detailed deterministic description of the system can be avoided. Generalized Stochastic Petri Nets (GSPN) allow transitions to be either timed or immediate. The state space generated by GSPN is smaller than that of the SPN. Some other extended Petri Nets are: Colored Petri Nets (CPN), which give more compact graphical representation of Petri Nets. Extended Stochastic Petri Nets, which is an effort to include non-exponential distribution in the analysis of SPN-based models, and DSPN, which are Petri Nets with deterministic and exponential firing times.

There are many mathematical definitions for TPN, the one mentioned in Zurawski and Zhou (1994) is used.

Definition: A TPN is PN = (P, T, I, O, M_0, γ, τ): where

1. $P = (p_1, p_2, p_3, \ldots, p_m)$ is a finite set of places,
2. $T = (t_1, t_2, t_3, \ldots, t_n)$ is a finite set of transitions. $P \cup T \neq \phi$ and $P \cap T = \phi$,

3. I: (PxT) $\rightarrow$ N is an input function that defines directed arcs from places to transitions. where N is a set of nonnegative integers.
4. O: (PxT) $\rightarrow$ N is an output function that defines directed arcs from transitions to places.
5. M_0: P $\rightarrow$ N is the initial marking.
6. γ: P $\rightarrow$ R^+ is the delay function. R^+ is the set of nonnegative real numbers. and
7. τ: T $\rightarrow$ R^+ is the firing time function.

The above definition applies for an ordinary TPN. and the time delays are associated with places and transitions. The firing rules for the ordinary TPN are:

1. At any time instance. a transition t becomes enabled if each input place p of t contains at least one available token.
2. Once transition t is enabled. it starts firing by removing one token from each input place. It completes firing after time delay $\tau(t)$. and deposits one token into each output place p. These tokens become available after time $\gamma(p)$.

3.2 USING PETRI NETS IN SCHEDULING

Place/Transition Petri Nets (P/T nets) are very easy to develop and understand. From the point of representing them in computing software. they do not need a lot of effort. Their structure and time analysis are well established in the literature. On the other side. their graphical representation of complex systems is unrealistic. Colored Petri Nets (CPN) can be used to overcome this problem. The disadvantage of colored Petri Nets is their need of complex data structure for representation.

The main concern of scheduling optimization is the state space of the problem. Both of P/T nets and CPN of the same system have the same state space. The complex representation of the colored Petri Nets needs more effort which may be reflected on the memory requirements and solution time. Therefore, it is recommended using P/T nets in scheduling optimization and CPN in the performance evaluation and control of the manufacturing systems.

The goal of scheduling is to assign operations to resources and define the start and finish time of each operation. Petri Nets can easily and concisely model the complex features of scheduling problems, such as material handling activities, limited buffer sizes, precedence relations and routing flexibility. The use of Petri Nets for the scheduling of manufacturing systems has been increasing. In the PN model of a scheduling problem, each resource and operation is represented by a place. The transitions represent the start and/or finish time of an event. The operation (processing or handling) time can be represented by assigning time to the corresponding place in the PN model. The distribution of the tokens over the PN places at any time represents the current system state. A token in place p becomes available after a duration which equals the time assigned to p. In the ordinary PN model, a transition t is called enabled if each input place p of t contains at least one available token.

In FMSs, each part type has a production sequence which is represented by a state machine (i.e. each transition has one input place and one output place) in the PN model as shown in Figure 3.1. The initial number of tokens in the first (Start) place of the state machine i represents the batch size of the part type i. The current marking of the last

(End) place in the state machine represents the number of finished parts. If the part has more than one route, a different state machine is used to represent each possible route.

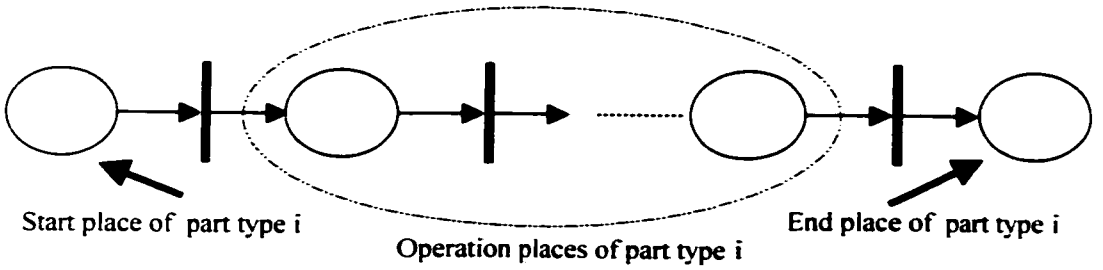


Figure 3.1: State machine of sequence j of part type i

Example:

Consider an FMS composed of three machines M1, M2, and M3, and a common buffer B which produces two part types, Part1 and Part2 [Chen and Jeng (1995)]. Each machine can process one part at a time. The available capacity of the intermediate buffer is two. The production sequences of Part1 and Part2 and the processing times are shown in Table 3.1. Each part type presents alternatives in its production sequence, i.e., routing flexibility. Figure 3.2 shows the timed Petri Net sub-models of Part1 and Part2. The timed Petri Net model of the whole system is the combination of the two sub-models (a) and (b) through the shared resources M1, M2, M3, and B.

Table 3.1: Production sequences alternatives and processing times

Part i	Alternative	Operation Number		
		1	2	3
		(Machine, Time)	(Machine, Time)	(Machine, Time)
Part1	1	(M2, 300)	(M1, 200)	------
	2	(M2, 300)	(M3 ,150)	------
Part2	1	(M1, 220)	(M3, 320)	------
	2	(M2, 350)	(M3, 320)	------
	3	(M1, 100)	(M2, 200)	(M3, 400)

3.3 OBTAINING AN OPTIMAL SCHEDULE

The Petri Net model is executed by firing one enabled transition at a time. For any situation where there is more than one enabled transition, the first in the set of enabled transitions is fired, while the others are stored as the set of alternative transitions at this stage. This process is repeated until the marking of the PN model reaches the final marking. At this point, the value of the criterion used for this sequence of fired transitions is recorded as the upper bound (ubv) of the search. Backtracking is initiated up to the stage where alternative transitions exist. A new branch is formed by firing one of the alternative transitions. The PN model is then executed in the manner described above until the final marking is reached again. If the new value of the evaluation criterion is better than the upper bound, the new value is stored as an upper bound and the new sequence is stored as the current minimum solution. This process is repeated until all possible paths have been explored.

Of course, this method guarantees obtaining an optimal solution, but the solution time will grow exponentially with the problem size. Therefore, heuristics are used to limit the search space. Using heuristics may not guarantee obtaining a global optimum solution, but a good or near-optimal solution can be reached in a reasonable time. Some of these heuristic functions and truncation techniques will be presented in section 3.4 and chapter 4.

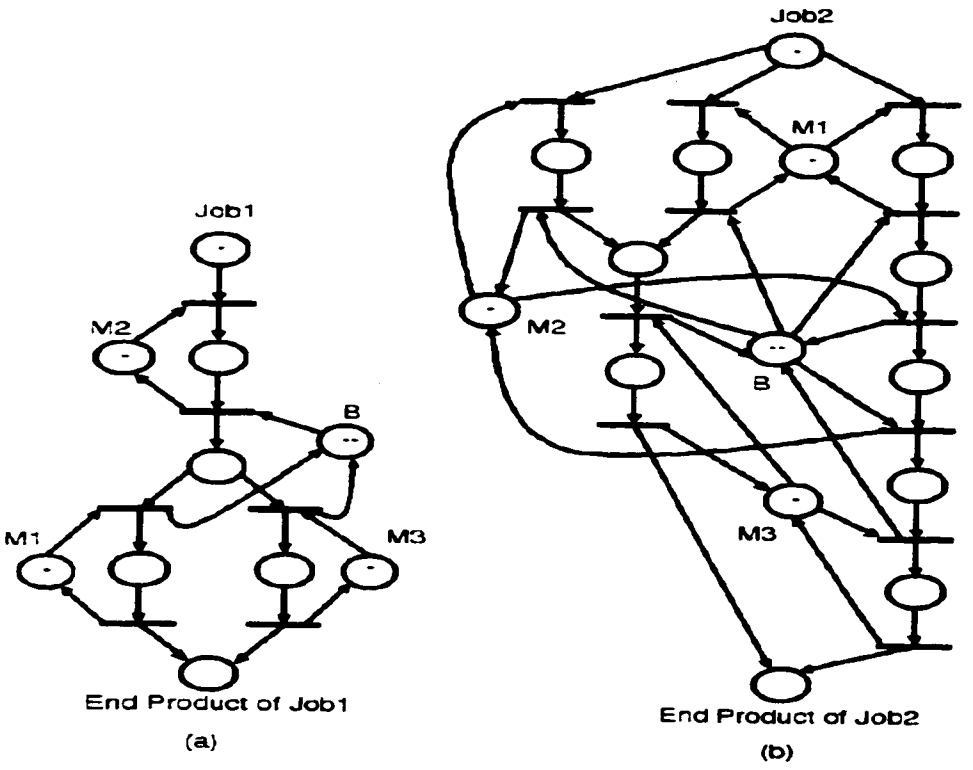


Figure 3.2: Petri Net Sub-models: (a) Sub-model of Part1, (b) Sub-model of Part2

3.4 TRUNCATION TECHNIQUES

The search space of the scheduling problem has been reduced significantly using the timed Petri Nets, although searching the whole reachability graph of the PN model belongs to the combinatorial optimization. Therefore, some truncation techniques have been developed [Shen et al. (1992), and Abdallah, ElMaraghy and ElMekkawy (1998)] to limit the search space. The truncation techniques developed in Shen et.al (1992) are for optimizing the makespan. No truncation techniques or heuristic functions have been developed for optimizing other performance criteria.

Look-ahead:

In the search for an optimal schedule, at every new marking of the PN, the current time is compared to the upperbound of the search. If it is less, then the search along that branch continues; otherwise, a backtracking step is performed since the current path will surely not result in the optimum solution.

Duplicate Marking:

When a transition is fired, it is checked if the obtained marking is repeated in the current minimal solution. If this occurs and the time required to reach this marking in the current search is less than the time of reaching it in the current minimal solution, the search along that branch continues; otherwise, a backtracking step is performed.

3.5 AUTOMATIC GENERATION OF PETRI NET MODEL

As mentioned in section 3, the graphical representation of complex systems using Petri Nets is unrealistic. In addition, random generation of test cases in scheduling is very important especially for testing the performance of an algorithm under changing some parameters. To overcome these two problems, the automatic generation of the Petri Net model from a production plan is needed. Hence, an algorithm is developed for generating the ordinary Petri Net models to serve the objectives of the rest of the thesis. In this algorithm, it is assumed that a part may have more than one route (sequence). All routes of the same part have the same number of operations. Each route includes the required machines, material handling facilities, and buffers. The following notation is used in the algorithm.

Notations:

a- Input (or randomly generated) parameters

- n: number of part types
- no_i: number of operations of part type i, i = 1,2,....,n
- ns_i: number of sequences of part type i, i = 1,2,....,n
- seq_matrix[i][j][k]: sequence matrix, that represents the different sequences of all part types, where: i = 1,2,....,n j = 1,2,....,ns_i k = 1,2,....,no_i
- nr: number of used resources (machines, robots, buffers, ..etc.)

b- Estimated parameters

- Sp[i]: a vector represents the start places, i = 1,2,....,n
- Ep[i]: a vector represents the end places, i = 1,2,....,n
- Op[j]: a vector that represents the operation places of the different sequences, where: j = 1,2,....,$\sum no_i \times ns_i$
- Rp[i]: a vector of resource places that represents the used resources, i = 1,2,....,nr
- Trans_matrix[i][j][k]: transitions matrix, that represents the transitions used within each sequence, where: i = 1,2,....,n j = 1,2,....,ns_i k = 1,2,....,no_i+1
- np: number of places of the Petri Net model
- nt: number of transitions of the Petri Net model
- Incid_matrix[i][j]: incidence matrix of the Petri Net model

Begin

0. i=0; w=0; kk=0; jj=0;
1. While(i<n)

1.1.Sp[i]=w; j=0;

1.2.While(j< ns_i)

1.2.1. k=0;

1.2.2. While(k< no_i)

1.2.2.1. w=w+1; Op[jj]=w; jj=jj+1;trans_matrix[i][j][k]=kk; kk=kk+1; k=k+1;

1.2.3. trans_matrix[i][j][k]=kk; kk=kk+1; j=j+1;

1.3.w=w+1; Ep[i]=w; w=w+1; i=i+1;

2. i=0;

3. While(i<nr)

3.1.Rp[i]=w; w=w+1; i=i+1;

4. np=w; nt=kk; i=0;

5. While(i<n)

5.1.k=Sp[i]; j=0;

5.2.While(j<ns_i)

5.2.1. w= trans_matrix[i][j][0];

5.2.2. Incid_matrix[k][w]=-1; j=j+1;

5.3.k=Ep[i]; j=0;

5.4.While(j<ns_i)

5.4.1. kk=no_i; w=trans_matrix[i][j][kk]; Incid_matrix[k][w]=1; j=j+1;

5.5.i=i+1;

6. jj=0; i=0;

7. While(i<n)

7.1.j=0;

7.2.While(j<ns_i)

7.2.1. k=0;

7.2.2. While(k< no_i)

7.2.2.1. kk=Op[jj]; jj=jj+1; w=trans_matrix[i][j][k]; Incid_matrix[kk][w]=1;

7.2.2.2. w=trans_matrix[i][j][k+1]; Incid_matrix[kk][w]=-1; k=k+1;

7.2.3. j=j+1;

7.3.i=i+1;

8. i=0;

9. While(i<n)

9.1.j=0;

9.2.While(j<ns_i)

9.2.1. k=0;

9.2.2. While(k< no_i)

9.2.2.1. jj=seq_matrix[i][j][k]-1; kk=Rp[jj]; w=trans_matrix[i][j][k+1];

9.2.2.2. Incid_matrix[kk][w]=-1;

9.2.2.3. w=trans_matrix[i][j][k+1]; Incid_matrix[kk][w]=1; k=k+1;

9.2.3. j=j+1;

9.3.i=i+1;

End

The set of places of the Petri Net model is classified into four groups.

1- Sp: the set of start places of the different part types.

2- Ep: the set of end places of the different part types,

3- Op: the set of operation places, and

4- Rp: the set of places representing the set of resources (machines, material handling facilities, and buffers).

The developed method of automating the generation of a Petri Net model consists of three main steps:

1- Estimate the total number of places of the Petri Net model, and define these places.

2- Estimate the total number of transitions of the Petri Net model, and define these transitions, and

3- Estimate the set of input and output transitions of every place.

Steps 0 to 4 of the algorithm estimate total number of (and define) the different places and transitions of the Petri Net model. The set of the input and output transitions of the four places sets are estimated using the remaining steps of the algorithm.

ILLUSTRATIVE EXAMPLE:

Consider an FMS composed of three machines M1, M2, and M3, and a robot R that produces two parts, Part1 and Part2. The robot is used for loading/unloading the parts to/from the machines. Table 3.2 shows the production plans for both part types.

Table 3.2: The production plan of the illustrative example

PART	SEQUENCE	OPERATIONS						
		1	2	3	4	5	6	7
Part1	1	R	M1	R	M2	R	--	--
	2	R	M2	R	M1	R	--	--
Part2	1	R	M3	R	M1	R	M2	R
	2	R	M3	R	M2	R	M1	R
	3	R	M2	R	M1	R	M3	R

Figures 3.3 and 3.4 show the Petri Net sub-models of Part1 and Part2 consequently. The combination of both models with the shared resources gives the overall Petri Net model for the system. The required resources are not shown in Figure 3.4 for simplifying the graph. The resources can be added in the same manner shown in Figure 3.3. The parameters estimated by the algorithm are shown in Table 3.4.

Table 3.3: The estimated parameters of the illustrative example

np = 39
nt = 36
Sp = {0,12}
Ep = {11,34}
Op = {1,2,3,4,5,6,7,8,9,10,13,14,15,16,17,18,19, 20,21,22,23,24,25,26,27,28,29,30,3,32,33}
Rp[i] = {35,36,3,38}
Trans_matrix[0][0] = {0,1,2,3,4,5}
Trans_matrix[0][1] = {6,7,8,9,10,11}
Trans_matrix[1][0] = {12,13,14,15,16,17,18,19}
Trans_matrix[1][1] = {20,21,22,23,24,25,26,27}
Trans_matrix[1][1] = {28,29,30,31,32,33,34,35}
The incidence matrix is estimated as it is defined in section 3.1, e.g. Incid_matrix[0][0] = -1, Incid_matrix[0][1] = -1, Incid_matrix[0][j] = 0, where j = 2,3,....,35

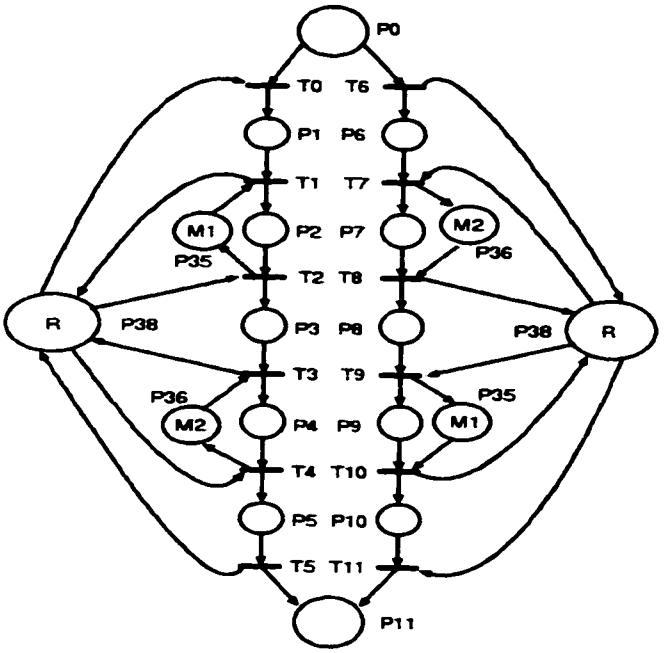


Figure 3.3: Petri Net sub-model of Part1

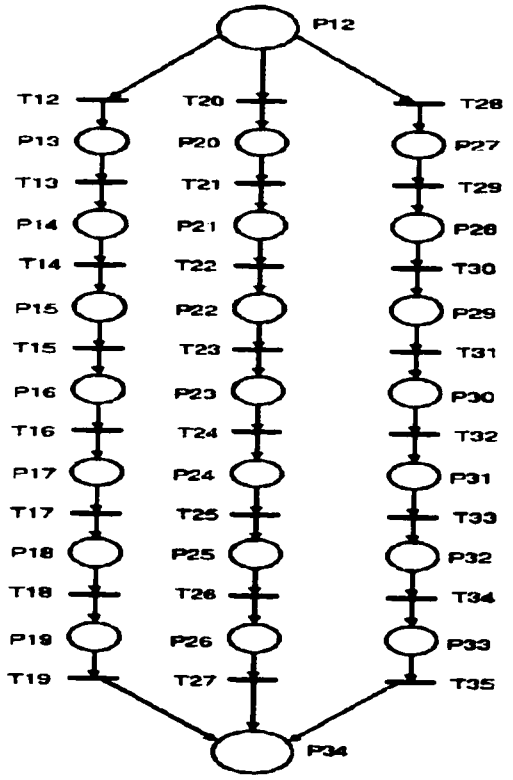


Figure 3.4: Petri Net sub-model of Part2 without the resource places

CHAPTER 4

HEURISTIC SEARCH FOR DEADLOCK-FREE OPTIMAL SCHEDULING IN FMSs

4.1 INTRODUCTION

Most of the scheduling problems are known to be NP-hard where the computation time required to obtain the global optimal schedule grows exponentially with the problem size [Pinedo (1995)]. Therefore, heuristic methods [Abdallah et al. (1998-a), Chen and Jeng (1995), and Xiong and Zhou (1997)] have been developed to tackle this exponential complexity. The heuristics solve complex problems effectively by reducing the number of evaluations and obtaining solutions within reasonable time. Petri nets are well suited for representing such FMS characteristics as precedence relations, concurrency, conflict, and synchronization. A powerful feature of Petri nets is their ability to detect the good behavior properties of the system such as deadlock-freeness.

Timed Petri nets have been used for scheduling problems [Carlier and Chretiene (1988)]. It is possible to generate an optimal schedule by computing the reachability graph (RG). However, enumerating the whole RG is an exponentially complex problem. Hence, heuristic search approaches were developed to generate only a portion of the RG using truncation techniques and dispatching rules methods [ElMekkawy, Abdallah, and ElMaraghy (1998), Chen and Jeng (1995), Lee and Dicesare (1994), Xiong and Zhou (1997), and Zhou et al. (1995)]. Chen and Jeng (1995) developed a heuristic algorithm based on the depth-first technique. Heuristic functions based on the state equation of the Petri net model were used. Lee and DiCesare (1994) presented a heuristic algorithm

based on A^* search technique. The effect of using four different heuristic functions on the search performance was studied. Xiong and Zhou (1997) developed a hybrid heuristic algorithm for optimal or near-optimal deadlock-free scheduling. The algorithm combines depth-first search and controllable backtracking. Zhou et al. (1995) developed a heuristic algorithm utilizing the branch-and-bound technique. They assumed an upper bound of the makespan, provided by the user, as an initial solution.

In this research, a search algorithm based on the depth-first principle and backtracking is developed. The algorithm optimizes the makespan. Two truncation techniques are developed and added to the search algorithm. The first truncation technique is based on the structure theory of the Petri nets. This technique deals with the deadlock problem and its objective is to avoid the deadlock situation efficiently during the search process. The second truncation is based on the Petri net structure and the concepts of delay and non-delay scheduling [Pinedo (1995)]. This technique provides an opportunity for trade-off between the solution quality and the search effort.

Most of the research work related to using Petri nets in scheduling focused on optimizing the makespan in spite of the importance of optimizing other measures of criteria such as average flow time. Optimizing the average flow time is equivalent to minimizing the mean completion time, mean lateness and mean waiting time [Ramaswamy, and Joshi, 1996]. Therefore, another version of algorithm was developed in order to optimize the average flow time. In addition, three heuristic functions, for optimizing average flow time, were developed to reduce the complexity of the scheduling problem when optimizing the average flow time. Each heuristic function was equipped with a parameter to provide a trade-off between the solution quality and the search effort.

The power of these heuristics is the ease of modifying them to deal with other measures of performance such as resources utilization, and due date measures.

The effects of extending the modified algorithm with the truncation techniques on the system and search performances were investigated. Then, the search algorithm, that optimizes the average flow time, was equipped with each one of the heuristic functions to guide the search during evaluating them.

4.2 SEARCH ALGORITHM

The search algorithm consists of two stages: Initialization and Backtracking. The Initialization step finds an initial solution and the Backtracking step finds an optimal (or near optimal) solution. The two stages are shown in Figures 4.1, and 4.2. The following notations are used in the search algorithm.

Notations:

- m_o: initial marking of the PN model.
- m_f: final marking of the PN model.
- m: current marking of the PN model.
- P_R_T: a vector represents the ready times of tokens in every place at time t.
- K: the expected maximum number of transitions to be fired to reach m_f from m_o.
- k: current search step number, k=1,2,3,....,K.
- *ubv*: value of the upper bound of makespan.
- *cv*: current value of the makespan.
- U_B_S: sequence of transitions which gives *ubv*.
- C_S: the current sequence of transitions.

- *Clock_time*: current clock time.
- Data[k]:

  - *m[k]*: marking of the PN model at the kth step.

  - *P_R_T[k]*: place ready time at the kth step.

  - *TE_set[k]*: list of enabled transitions based on P_R_T at the kth step.

  - *SE_set[k]*: list of enabled transitions without considering time at the kth step.

  - *E_L[k]*: list of completion times of operations in progress at kth step.

  - *Clock_time[k]*: the value of *Clock_time* at the kth step.

Initialization

Input: M_0, m_f, K, and S_N ;

Output: ubv and U-B-S ;

1. $m = M_0$;

2. k = 1 ;

3. Estimate Data[k] ;

4. If ($m = m_f$) go to (10) ;

5. If (SE-set[k] = ∅) go to (9) ;

6. If (TE-set[k] = ∅) go to (7) ;

   6.1. Select t from TE-set[k] ;

   6.2. Delete t from TE-set[k] and SE-set[k] ;

   6.3. If (Test0 = True) go to (9) ;

   6.4. Add t to U-B-S ;

6.5. Fire t and update m ;

6.6. k = k + 1 ;

6.7. update Data[k] :

6.8. Go to (4) ;

7. Clock-time = min { E-L[k] } ;

8. Update Data[k]; go to (6) ;

9. Do backtracking. go to (6) ;

10. ubv = Clock-time ; stop ;

Backtracking:

Input: ubv and U-B-S and S_N ;

Output: Optimal (or near optimal) firing transitions sequence and the minimal upper bound ;

1. Copy U-B-S to C-S ;

2. k = k -1 ;

3. If (k < 1) stop ;

4. m = m[k] ;

5. If (m = m_f) go to (11) ;

6. If (SE-set[k] = ∅) go to (10) ;

7. If (TE-set[k] = ∅) go to (8) :

7.1 select t from TE-set[k] :

7.2 Delete t from TE-set[k] and SE-set[k] :

7.3 If (Test0 = True) go to (10) :

7.4 C-S[k] = t :

7.5 Fire t and update m :

7.6 k = k +1 :

7.7 update Data[k] :

7.8 Go to (5) :

8. Clock-time = min {E-L[k] } ;

9. Update Data[k]; go to (7) ;

10. Do backtracking; go to (7) :

11. cv = Clock-time ;

12. If (cv < ubv) ;

12.1 ubv = cv ;

12.2 Copy C-S to U-B-S ;

13. Else Copy U-B-S to C-S;

14. Go to (2);

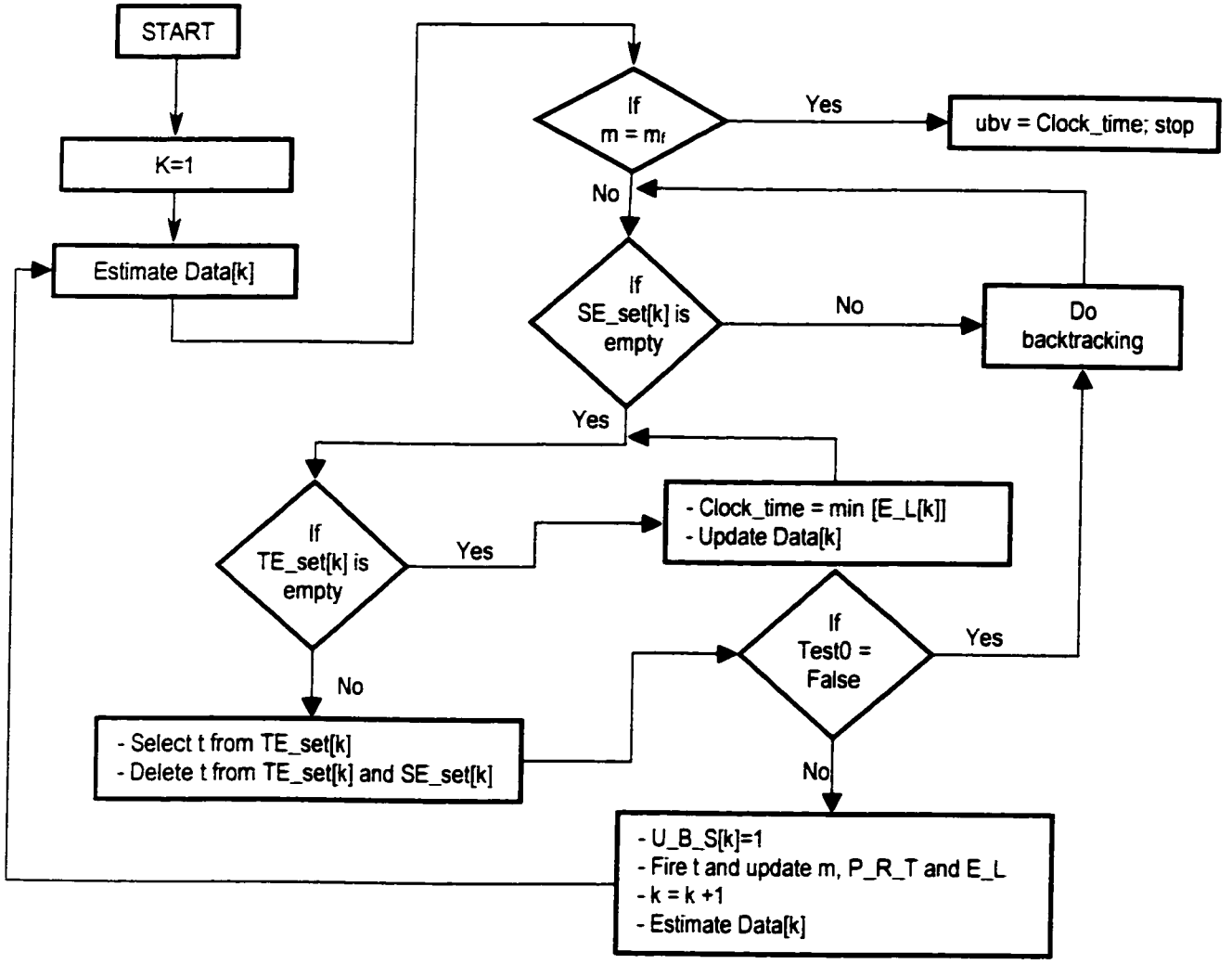


Figure 4.1: Initialization stage of the scheduling algorithm

Initialization: Starting from m_0 a sequence of transitions is found to reach m_f. Noting that *TE_set* is a subset of *SE_set*, in every search step, the following alternatives can occur:

1) *TE_set* is not empty;

2) *TE_set* is empty and *SE_set* is not empty; and

3) *SE_set* is empty.

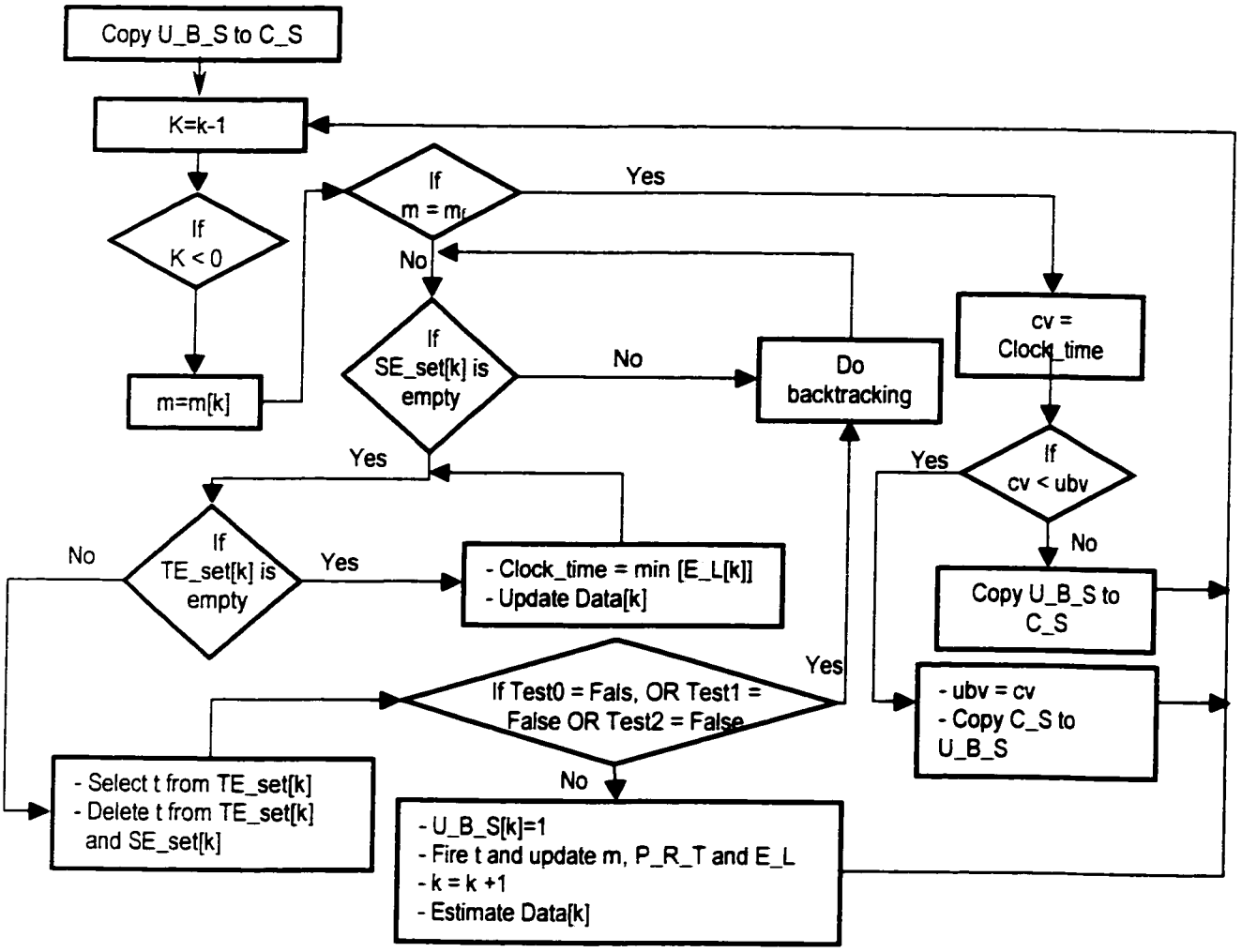


Figure 4.2: Backtracking stage of the scheduling algorithm

In the first case, a transition t is selected from *TE_set* to fire arbitrary (or by using a dispatching rule). The firing action is performed by updating *m, P_R_T, and E_L*. In the second case, some or all of the required resources, by waiting jobs, are still busy. Hence, Clock-time is advanced to the minimum completion time of the operations in progress. This time advance leads to nonempty *TE_set*, then a transition t is selected for firing. In

the last case, a deadlock situation is reached (Test0 = False). Therefore, backtracking is performed to find the closest step (including the current step after deleting t from its *TE_set* and *SE_set*) that has non-empty *SE_set*. If the *TE_set* of the found step is empty, *Clock_time* is advanced to the minimum completion time of the operations in progress. Then a transition t is selected for firing. This logic is followed until the final marking is reached. Then the sequence of transitions which represents the initial deadlock-free schedule is recorded in *U_B_S* and the upper bound of the makespan is recorded in *ubv*.

BACKTRACKING: Starting from the final marking step obtained in the Initialization stage, backtracking to the previous steps is conducted to improve the initial solution. The backtracking search steps are treated similar to the Initialization stage but with two basic differences. When generating a new branch, in addition to Test0, two truncation techniques [Chen and Luh (1992)] (Test1 and Test2) are used. Test1 compares the current time of each new marking to *ubv*. If the current time is less, then the search along this branch continues (Test1=True), otherwise, it is rejected (Test1=False). Test2 checks for duplicate markings while generating a new branch. If the time required to reach the duplicate state in the current search is greater than or equal to the time required to reach this state in the current minimal solution, the new branch is rejected (Test2=False); otherwise, the search along this branch continues (Test2=True). Every time the final marking is reached in a new branch, the obtained value of the makespan (*cv*) is compared with *ubv*. If the *cv* is less than *ubv*, the obtained sequence (*C_S*) is stored in *U_B_S*, and *ubv* is set to *cv*. This process is repeated until no more paths remain unexplored.

4.2.1 ILLUSTRATIVE EXAMPLES

The search algorithm presented here was implemented in C on a Pentium 32.0 MB RAM/166 MHz. Two examples from the literature [Chen and Jeng (1995), and Xiong and Zhou (1997)] were considered for illustration.

Example 1:

Consider the system presented in Chapter 3, when only one unit of every job type is considered. A makespan equal to 540 hours was obtained in less than 0.001 CPU seconds using the proposed search algorithm. The same makespan was obtained by [Chen and Jeng (1995)] in 0.017 CPU seconds (the computing system was not specified). The Gantt chart of the obtained schedule is as shown in Figure 4.3.

Example 2:

Consider an automated manufacturing system composed of three machines M1, M2 and M3, a robot R and a load/unload station [Xiong and Zhou (1997), and Ramaswamy and Joshi (1996)]. The robot is responsible for handling parts between machines and the load/unload station. Four part types were considered. Their production sequences are : J1: M1, M2 and M3, J2: M2, M1 and M3, J3: M1, M2 and M3, J4: M3, M2 and M1. The processing and transportation times are given in Table 4.1, where $O_{i,j,k}$ means the *jth* operation of the *ith* part being performed by machine k, L_i (resp. U_i) represents the loading (resp. unloading) time of the ith part from the load (resp. unload) station, and $R_{i,j}$ represents the transportation time of the *ith* part to its *jth* operation.

Figure 4.4: shows the timed Petri net sub-model of J1. The timed Petri net model of the whole system is the composition of the four sub-models (J1. J2. J3. and J4) through the shared resources M1. M2. M3. and R.

Table 4.1: Operation and transportation times (in hours) of example 2

Operation	Time	Transport	Time
$O_{1,1,1}$	40	L_1 . L_2	5
$O_{1,2,2}$	100	L_3	6
$O_{1,3,3}$	36	L_4	4
$O_{2,1,2}$	65	$R_{1,2}$	3
$O_{2,2,1}$	45	$R_{1,3}$	5
$O_{2,3,3}$	98	$R_{2,2}$. $R_{4,3}$	3
$O_{3,1,1}$	212	$R_{2,3}$	6
$O_{3,2,2}$	73	$R_{3,2}$	7
$O_{3,3,3}$	32	$R_{3,3}$	4
$O_{4,1,3}$	35	$R_{4,3}$	5
$O_{4,2,2}$	65	U_1 . U_2	4
$O_{4,3,1}$	55	U_3 . U_4	5

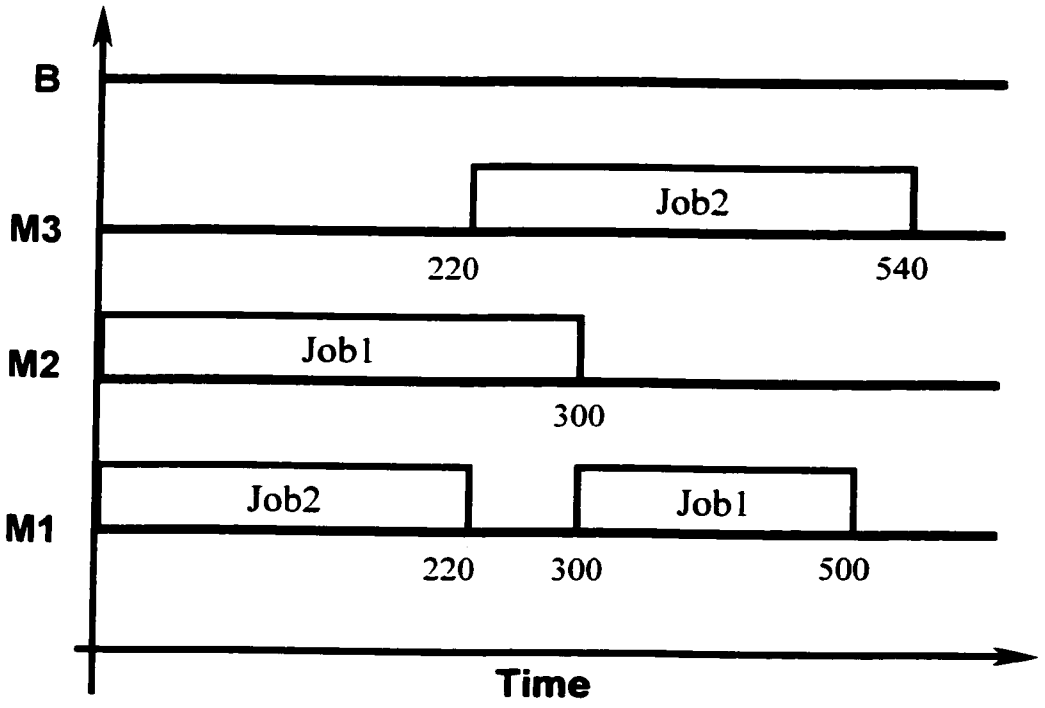


Figure 4.3: Gantt chart of the obtained schedule of example 1

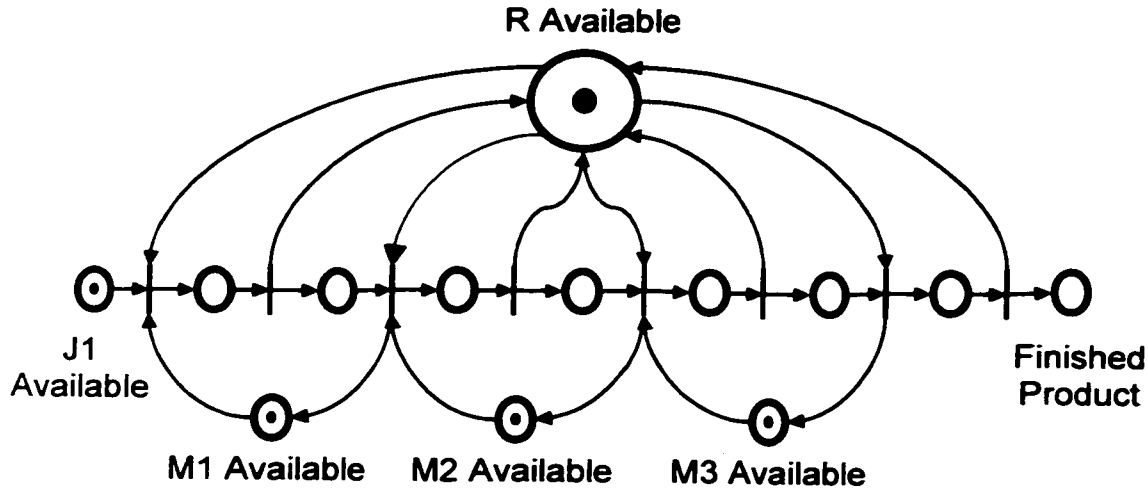


Figure 4.4: Petri net sub-model for J1 of example 2

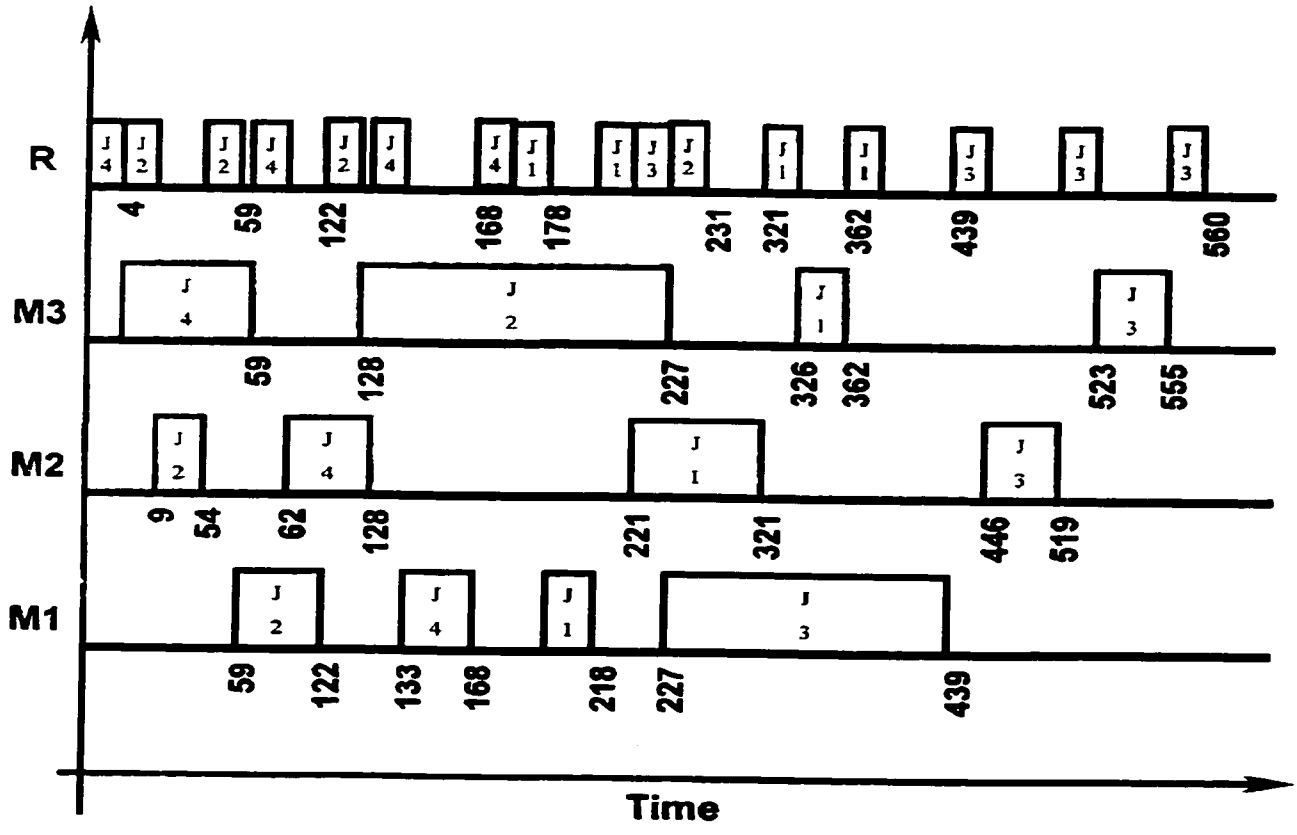


Figure 4.5: Gantt chart of the obtained schedule of example 2

When only one unit of every job type is considered, the computation time of the proposed search algorithm is 0.06 CPU seconds to obtain a minimal makespan equal to 560 hours. The same makespan was obtained in Xiong and Zhou (1997) in 0.41 CPU seconds on a SUN Sparc 20. The Gantt chart of the obtained schedule is as shown in Figure 4.5

4.3 TRUNCATION TECHNIQUES AND HEURISTIC FUNCTIONS

4.3.1 MINIMAL SIPHON CONCEPT

The effort of deadlock-free scheduling is divided into two parts. The first part is searching for optimal (or near-optimal) solution, and the second part is avoiding (by backtracking) deadlock states. Through my experience, it is noted that a significant time is usually spent in the second part. Therefore, new techniques should be developed to reduce this time. Working toward this direction, a new truncation technique is developed. The truncation technique is based on the minimal siphon property of the Petri net model. A minimal siphon is a set of places that can be computed by checking some structure conditions of the Petri net model [Abdallah, ElMaraghy, and ElMekkawy (1997)]. A Petri net model can contain more than one minimal siphon. An empty siphon at a reachable marking is equivalent to a partial deadlock in the FMS (i.e., a part of the system is deadlocked). Furthermore, a partial deadlock leads necessarily to a global deadlock (i.e., dead-marking) [Abdallah and ElMaraghy (1998-b)].

The minimal siphons of the Petri net model are computed using the algorithm developed in Abdallah, ElMaraghy, and ElMekkawy (1997) after implementing it using C language. Test0 is originally checking if a deadlock situation is reached, to do a backtracking action. Instead, Test0 was modified by checking if a minimal siphon became empty to do a backtracking action. This modification is expected to save wasted time of searching a deadlocked search path.

4.3.2 USER CONTROL FACTOR

Through application to several case studies, it was shown that the developed search algorithm has a reasonable response time. However, because the search effort may be high for large FMSs, relaxing the optimality condition becomes an economic necessity [Pearl (1984)]. Hence, some reasonable balance between the quality of the solution and the search effort should be found. The basic problem in handling a semi-optimization task is to devise algorithms that guarantee bounds on both the search effort and the extent to which the optimization objective is compromised. A more ambitious task would be to equip such an algorithm with a set of adjustable parameters so that the user can meet changes in emphasis between the cost and performance by controlling the trade-off between the quality of the solution and the amount of search effort. Hence, a user control factor (UCF) is defined and added to the search algorithm in the backtracking stage. The explored search space will depend on the value of UCF ($0 \leq UCF \leq 1$).

Definition:

Let *UCF* be equal to x. Let RN be a random number ($0 \leq RN \leq 1$). If $RN < x$, then backtracking is performed to a step with *TE_set* $\neq \phi$. Otherwise, backtracking is performed to a step with *SE_set* $\neq \phi$.

Example:

Consider the Petri net model shown in figure 4.6 It can be noted that *TE_set* = $\{T_1\}$ and *SE_set* = $\{T_1, T_3\}$. It can be shown that during the backtracking stage, exploring all transition alternatives of the *SE_set* will increase the computation time but may provide better solution quality. Conversely, exploring only transitions of *TE_set* will take a shorter computation time, but may reduce the solution quality. Thus, the user can compromise between the solution quality and computation time by fixing the appropriate *UCF*.

It can be noted that delay schedules [Pinedo (1995)] are allowed here when *UCF* does not equal 1. Indeed, exploring all nodes of *SE_set* means that the delay schedules are taken into consideration, while exploring only nodes of *TE_set* means that only non-delay schedules are taken into consideration. A backtracking-routine was added to the backtracking stage of the search, which utilizes the *UCF* concept as shown in figure 4.7

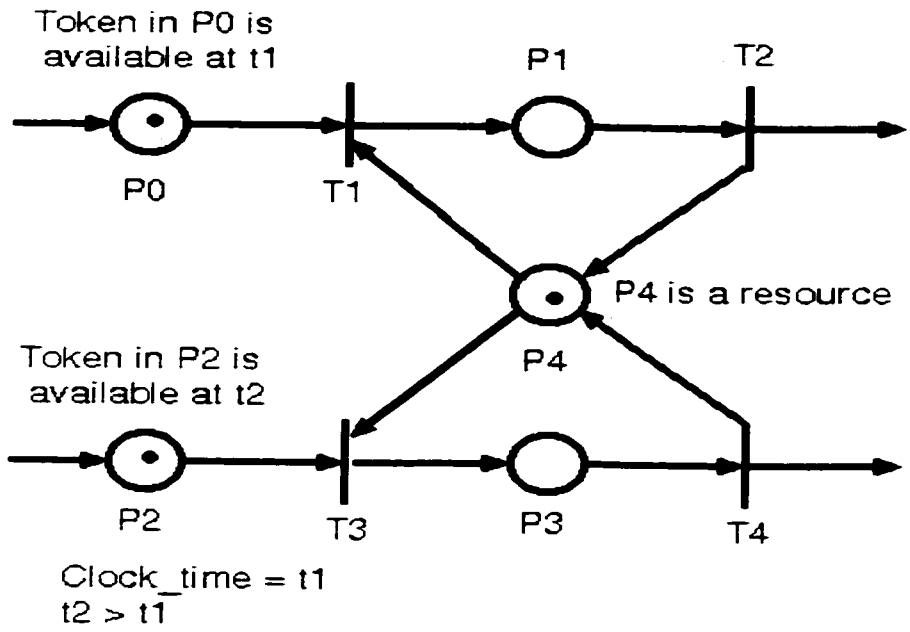


Figure 4.6: A part of a Petri net model

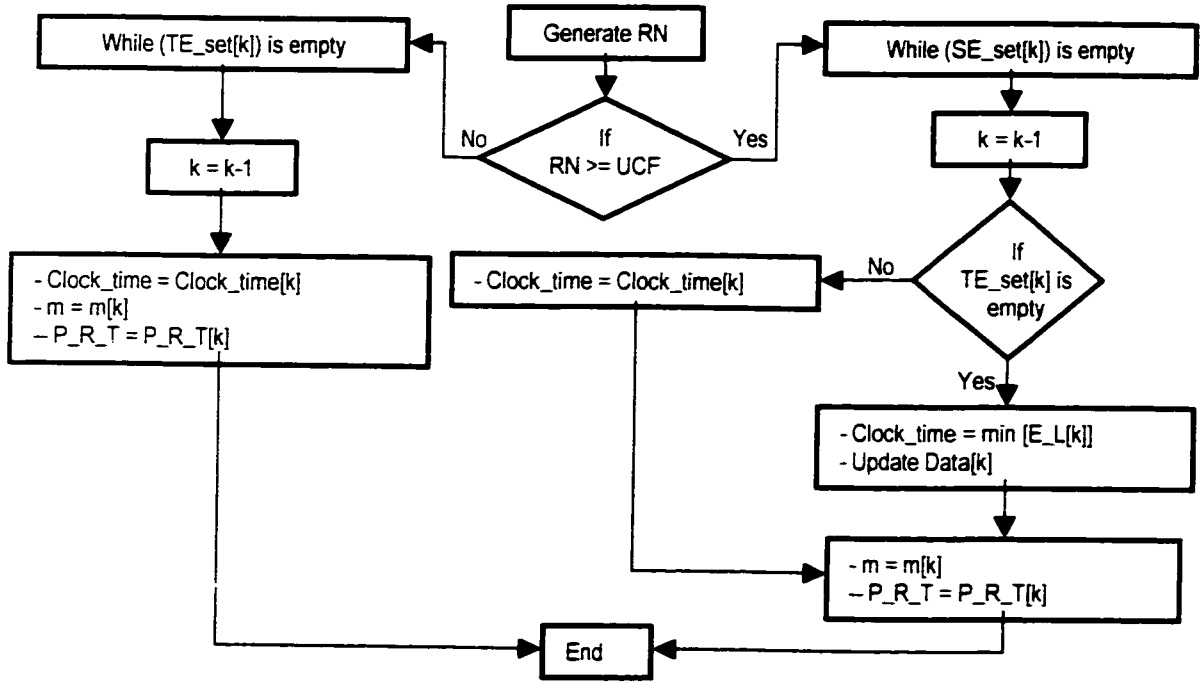


Figure 4.7: Backtracking-routine

4.3.3 AVERAGE FLOW TIME OPTMIZATION

The search algorithm, presented in section 4.2, was developed to optimize the makespan. This algorithm was modified to include three new heuristic functions for optimizing the average flow time. The modified algorithm is presented in Appendix 1. The search algorithm consists of two stages: Initialization and Backtracking. In the Backtracking stage, Test1 compares the current estimated average flow time (explained in section 3.1) of each new marking to *ubv*. If the current estimated average flow time is less than ubv, the search along this branch continues (Test1=True); otherwise, it is rejected (Test1=False). Test2 checks for duplicate markings while generating a new branch. If the estimated average flow time of the duplicate state in the current search is greater than or equal to the average flow time of this state in the current minimal solution, the new branch is rejected (Test2=False); otherwise, the search along this branch continues (Test2=True). Every time the final marking is reached in a new branch, the obtained value of the average flow time (*cv*) is compared with *ubv*. If the value of *cv* is less than *ubv*, the obtained sequence (*C_S*) is stored in *U_B_S*, and *ubv* is set to *cv*. This process is repeated until all paths have been explored.

The developed search algorithm has been used to test three heuristic functions. As mentioned above, Test1 and Test2 rely on estimating the total flow time from the current marking (search step k) to the final marking. The total flow time at any search step (*TFT*(k)) consists of the total flow time for the scheduled parts (*TFT*1(k)), and the estimated total flow time for the unscheduled parts (*TFT*2(k)), i.e. *TFT*(k) = *TFT*1(k) + *TFT*2(k).

Let K = the expected maximum number of transitions to be fired to reach final

marking from the initial marking.

k = current search step number. k=0.1. 2..... and $\leq K$.

n_1 = number of parts scheduled at search step k:

n_2 = number of unscheduled parts at search step k:

$T(k)$ = time at search step k.

rpt_i = remaining processing time of the ith part. i = 1.2.3. ...., n_2; and

rno_i = remaining number of operations of the ith part. i = 1.2.3. ...., n_2: and

$aowt_i$ = average operation waiting time of the ith part. i = 1.2.3. ...., n_2; and

$\alpha = k/K$, $0.0 \leq \alpha \leq 1.0$;

MF = modification factor.

The following three heuristic functions are used to estimate the term $TFT2(k)$:

(1) Remaining Processing Time (RPT):

$$TFT2(k) = \sum_{i}^{n_2} \{T(k) + rpt_i(\alpha + (1-\alpha)MF)\} \qquad (4.1)$$

The flow time lower bound of the ith unscheduled part at the k^{th} search step can be estimated by adding the remaining processing time of this part (rpt_i) to the time at the k^{th} search step ($T(k)$). The deviation between the estimated lower bound and the actual value of the flow time will be significant, if the difference between the number of unscheduled parts and the available number of resources is high. Hence, the compensation factor $(\alpha+(1-\alpha)MF)$ is used as shown in Equation 4.1.

The α factor gives a dynamic nature to the compensation factor. Hence, the effect of the compensation factor diminishes as the search goes deeper in the RG. The *MF* is the static part of the compensation factor, which is defined by the user.

(2) Average Operation Waiting Time (*AOWT*):

$$TFT2(k) = \sum_{i}^{n2} \{T(k) + rpt_i + rno_i\, aowt_i\, MF\} \qquad (4.2)$$

Using the knowledge of the upper bound schedule, the average waiting time per operation for every part type can be estimated. The average waiting time per operation and the remaining number of operations are estimated based on the maximum number of operations in case the part has more than one route. The flow time is estimated by adding the waiting time multiplied by *MF* to the remaining processing time of every unscheduled part.

(3) Scheduling with Dispatching Rules (*SDR*):

A dispatching rule, such as first come first served (FCFS) and shortest processing time (SPT) [10], may be applied to sequence the unscheduled part at any search step *k* using simulation of the PN model. A central buffer with infinite capacity is used in order to avoid the occurrence of deadlocks during simulation. It is assumed that every part acquires one resource only at a time and all resources always release the part to the central buffer.

After sequencing the unscheduled part at search step k, by using a dispatching rule, their total flow time is estimated (*rpft*). The value of *rpft* is a lower bound. In other words, this value will be less than or equal to the actual value if the same search path has

been used. Therefore, the compensation factor $(\alpha+(1-\alpha)MF)$ is used as shown in Equation 4.3.

$$TFT2(k) = stft(k)\,(\alpha + (1-\alpha)MF) \qquad (4.3)$$

Setting *MF* to one in *RPT* and *SDR* functions (or to zero in *AOWT* function), respectively, provides more opportunity for all nodes to be explored. On the other hand, increasing the value of the *MF* decreases this opportunity. Therefore, there should be a tradeoff between exploring more nodes in the RG in order to obtain a higher solution quality, and limiting the search effort in order to obtain a solution in a reasonable time.

In comparing the developed heuristic functions with those developed by Lee and DiCesare (1994), it should be noted that Lee and DiCesare (1994) used the A^* algorithm as a search technique and considered the makespan as an optimization criterion. In this research, the depth-first heuristic with backtracking was used as a search technique and average flow time as an optimization criterion. The Remaining Processing Time (*RPT*) function might have some similarity with the third heuristic function of Lee and DiCesare (1994). In their function they estimated a heuristic value for each marking, at each search step, in order to select the most promising marking for further exploration. Their heuristic function was a summation of the minimum remaining processing time of the places that have a token under the marking and the negative value of the marking depth in the reachability graph. In the *RPT* function, the remaining processing times of unscheduled parts, at each search step, are modified by multiplying them by the modification factor (see Equation 4.1). As mentioned earlier, the α factor gives a dynamic nature to the

compensation factor. Hence. the effect of the compensation factor diminishes as the search goes deeper in the RG.

The other two heuristic functions. Average Operation Waiting Time (*AOWT*) and Scheduling with Dispatching Rules (*SDR*), have not been used before in the literature. especially in the area of deadlock-free scheduling using Petri Nets. In addition. The developed heuristics can be easily modified to deal with other measures such as machines utilization and due date related measures. Hence. they are based on estimating the completion time of every unscheduled part. In other words. an expected schedule is assumed (for the unscheduled parts) at every search step. Therefore. any other measure. at the current search path. can be used (instead of the average flow time) for comparison with the upper bound value that has been obtained during the initial solution stage or a previous search path.

4.4 EXPERIMENTAL RESULTS

The search algorithm presented in this paper was implemented in C on a 32 MB RAM/166 MHz Pentium PC. The efficiency of the developed algorithm is illustrated by randomly generated test cases using two sets of specifications that are presented in Appendix 2.

The random generation of test cases in scheduling is important especially for testing the performance of an algorithm while changing some parameters. but it usually results in a huge number of test cases. It is impractical to manually develop the Petri Net models of the generated test cases. Hence. a routine has been developed for automatically

generating them [Ezpeleta and Colom (1997). Sun and Fu (1994).and Xue et a. (1998)]. This routine is presented in Chapter 3.

4.4.1 MINIMAL SIPHON CONCEPT

The number of work centers. number of part types. number of operations of each part type. and number of sequential processes of each part type have been varied to obtain systems with different numbers of siphons. The rest of the parameters was kept the same as defined in data set 1 (see Appendix 2). These variations are summarized Table 4.2.

The algorithm. that optimizes the average flow time. was applied to generated cases with and without the siphon concept in order to study the effect of this concept on the solution CPU time. Five randomly generated cases have been performed for each siphon interval. which resulted in 50 runs (25 with siphon concept and 25 without siphon concept). Figure 4.8 summarizes the results. The percentage difference in CPU time was calculated as: ((Solution CPU time without siphons - Solution CPU time with siphons)/ Solution CPU time without siphons) x 100.

Table 4.2: Parameters variation

Number of siphons	# of work centers	# of part types	# of operations	# of processes
1-5	2 and 3	10-20	1-3	1-3
6-10	3	20-50	1-3	1-3
11-15	4	50	1-3	1-3
16-20	5	20-50	1-3	1-3
21-25	6	50-100	1-4	1-4

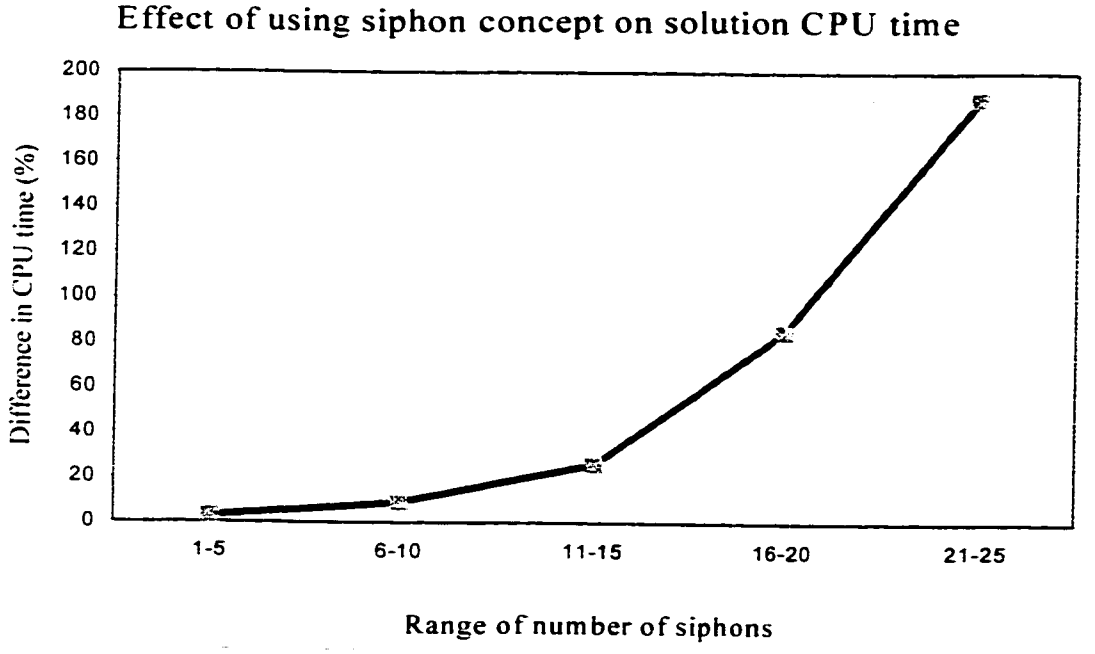


Figure 4.8: Effect of siphon concept on the CPU time

From the above results, the following conclusions can be drawn about the efficiency and effectiveness of using the siphon concept:

(1) The siphon concept enhances the ability to develop deadlock-free schedules of systems with high number of deadlocks (i.e., uncontrolled siphons). The high number of uncontrolled siphons results in a high probability of reaching a marking with an empty siphon. Therefore, unnecessary branches are not generated and the search space is dramatically reduced.

(2) No significant usefulness was found in using the siphon concept for systems with a small number of uncontrolled siphons. The low number of uncontrolled siphons results in a low probability of reaching a marking with an empty siphon. Executing Test0 at each reachable marking is time-consuming compared with generating the whole branches in the reachability graph.

(3) The developed scheduling algorithm assumes the availability of a set of uncontrolled minimal siphons. This set is computed using an extended efficient search algorithm. developed for S^3PR nets [Abdallah et al. (1997)]. For a system with 25 siphons. the algorithm takes 24.06 seconds on a Pentium 32.0MB RAM/166 MHZ computer.

(4) The worst case complexity of the scheduling algorithm is exponential. but the use of siphon concept reduces the CPU time and enhances the ability to develop deadlock-free schedules for flexible manufacturing systems with high number of deadlocks.

4.4.2 USER CONTROL FACTOR

The specifications of data set 1 (see Appendix 2) were used in generating the test cases. Siphon concept was used. All the test cases have 11 minimal siphons. Three *UCF* values have been used (0.0. 0.5. 1.0) to study the effect of the *UCF* on the CPU time. Five randomly generated cases have been performed for each number of part types interval and each *UCF* value. which results in 60 runs. Figure 4.9 shows the results.

It can be noted that the variation of the *UCF* from 0 to 1 reduces the CPU time dramatically. particularly for higher number of part types (e.g. the CPU time was reduced from 2178.2 seconds to 285.1 seconds at 100-150 part types). In addition. no solution was obtained with test cases of more than 150 part types using *UCF*=0.0. On the other hand. it is noticed that a high value of the *UCF* does not affect the solution quality. i.e., the average flow time does not change when the *UCF* is increased. Therefore. unforced idleness in those scheduling problems (i.e., delay schedules) does not lead to better average flow time.

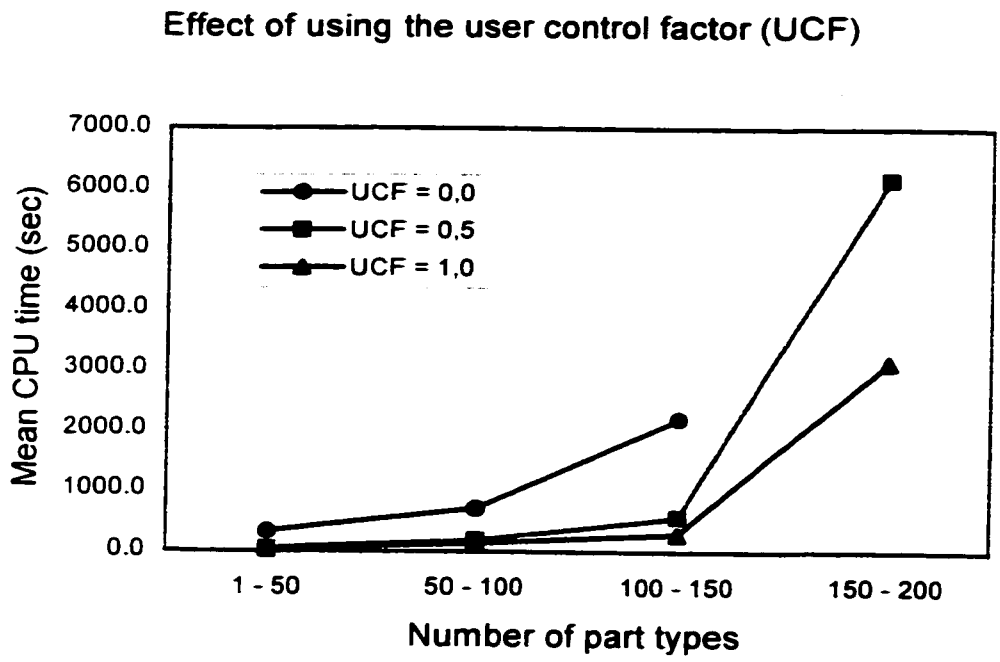


Figure 4.9: *UCF* effect on CPU time

4.4.3 HEURISTIC FUNCTIONS

The data set 2, presented in Appendix 2, were used to test the performance of the above three heuristic functions. Some preliminary experiments were performed and those *MF* values that had remarkable effect on the average flow time or solution CPU time were used for further experiments. The shortest processing time (SPT) dispatching rule was used for Scheduling with Dispatching Rules (*SDR*) function. The mean values of the average flow time and CPU time for the generated test cases are shown in Figure 4.10.

It can be noted from Figure 4.10 that the results obtained using the three heuristic functions exhibit the same trend. The average flow time decreases and the solution time increases as the *MF* value increases. In other words, A low value of *MF* results in better

average flow time but higher solution time since it provides more opportunity for exploring higher number of search nodes. Table 4.3 summarizes percentage of difference in average flow time and solution time at the highest and lowest value of *MF* using the three heuristic functions. The difference in both of average flow time and solution time becomes more significant (for example the average percentage difference in average flow time and CPU time, at AOWT, changes from 6.3 to 19.4 and from 54 to 138 respectively) as the number of part types increases but goes down at the last interval of number of part types. This might be justified by the significant reduction in the search space, with using the highest value of *MF*, as the number of part types increases up to a certain point then the effect of the *MF* factor starts to diminish. Using the lowest value of *MF*, with the three functions, results in no solution for test cases that have more than 150 part types. That is because more nodes are explored which leads to search space explosion.

The performance of the three heuristic functions is compared in Figure 4.11. The Average Operation Waiting Time (*AOWT*) function outperforms the other two functions with respect to the obtained average flow time and solution time. The efficiency of the *AOWT* function can be attributed to the use of the information obtained during the previous search path to reduce the time wasted in searching unpromising solution paths. On the other hand, the Dispatching Rules (*SDR*) function always results in the highest average flow time and solution time. This might be explained by the need for longer computations that are repeated in each search step. In addition, the *SDR* function assumes the existence of a central buffer with infinite capacity in order to estimate the average flow time for the unscheduled parts.

The capabilities of the modified algorithm (see Appendix 2) has been tested by using the *AOWT* function with *MF*=0.1. In addition to the efficiency of the developed heuristic function (*AOWT*), the proposed method can handle test cases with up to 612 operations (an operation is a part of a job and is defined as the total time (setup and processing) required by the operation on a certain resource). Jeng and Chen (1998), reported in their paper that "their method is unable to deal with more than 300 operations" but did not specify the used computer. A test case of 612 operations was solved using the above computing machine (Pentium 32MB RAM/166 MHZ computer) in 9756 seconds. The same test case was solved using a faster computing machine (Intel Celeron 64.0MB RAM/433 MHZ) in 9324 seconds. Larger test cases (with 647 operations and up) were fed to the faster computer for more than 20 hours without obtaining a solution.

Example2, presented earlier [Ramaswamy and Joshi (1996)], has been solved using the *AOWT* function. Table 4.4 presents the comparison between the results obtained using the proposed method and those reported in Ramaswamy and Joshi (1996). The efficiency of the *AOWT* function is demonstrated by obtaining the same solution as that obtained using the mathematical model of Ramaswamy and Joshi (1996) in a very comparable solution time as shown in Table 4.4.

The developed method helped in handling larger systems [compared to Jeng and Chen (1998)]. In addition, the use of the *MF* parameter in the heuristic functions provided a trade-off between the solution quality and solution time. The resulting increase (average of 14%) in the obtained average flow time is less significant than the reduction in the solution time (average of 71%).

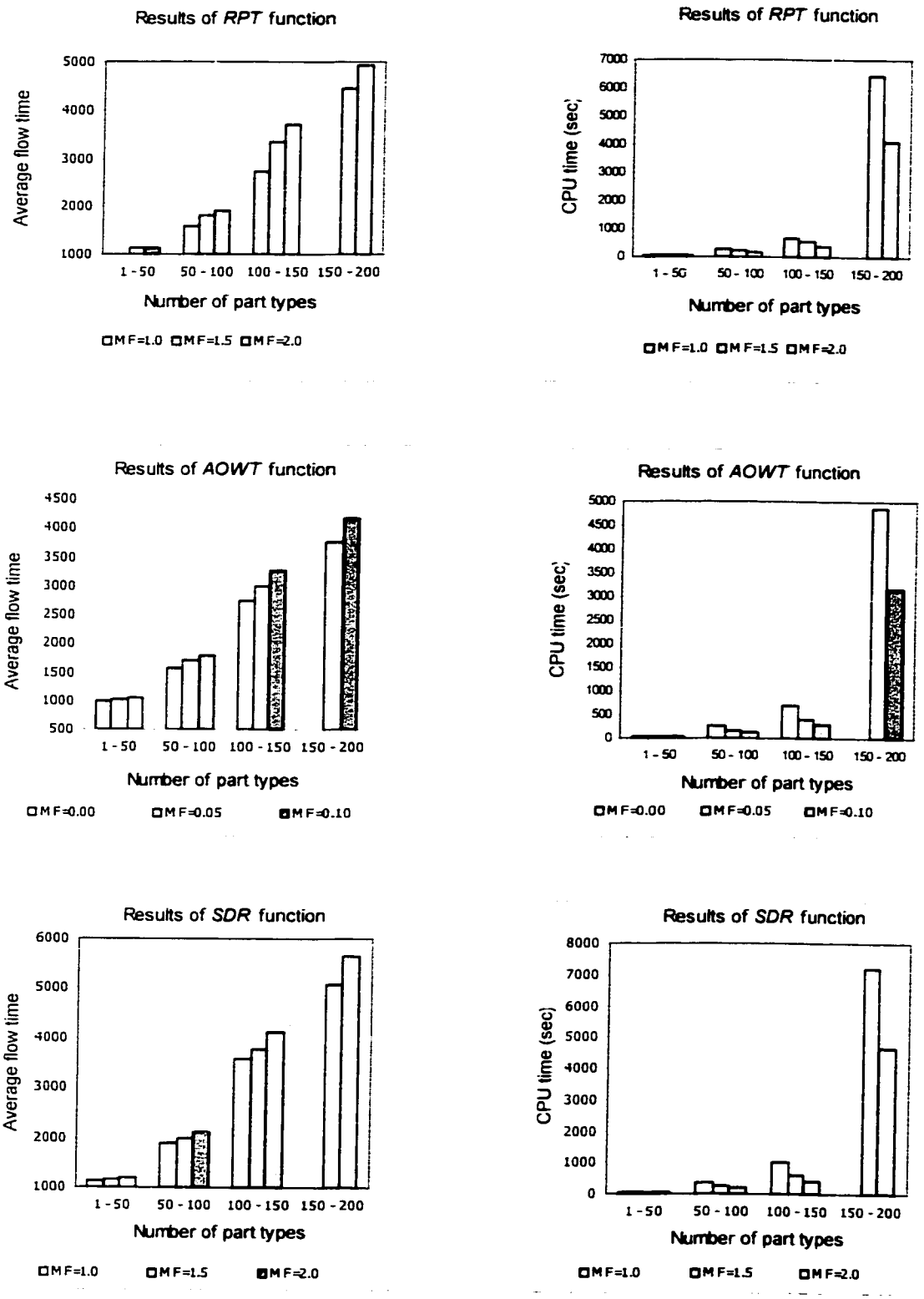


Figure 4.10: Scheduling results of experiments using the three heuristic functions

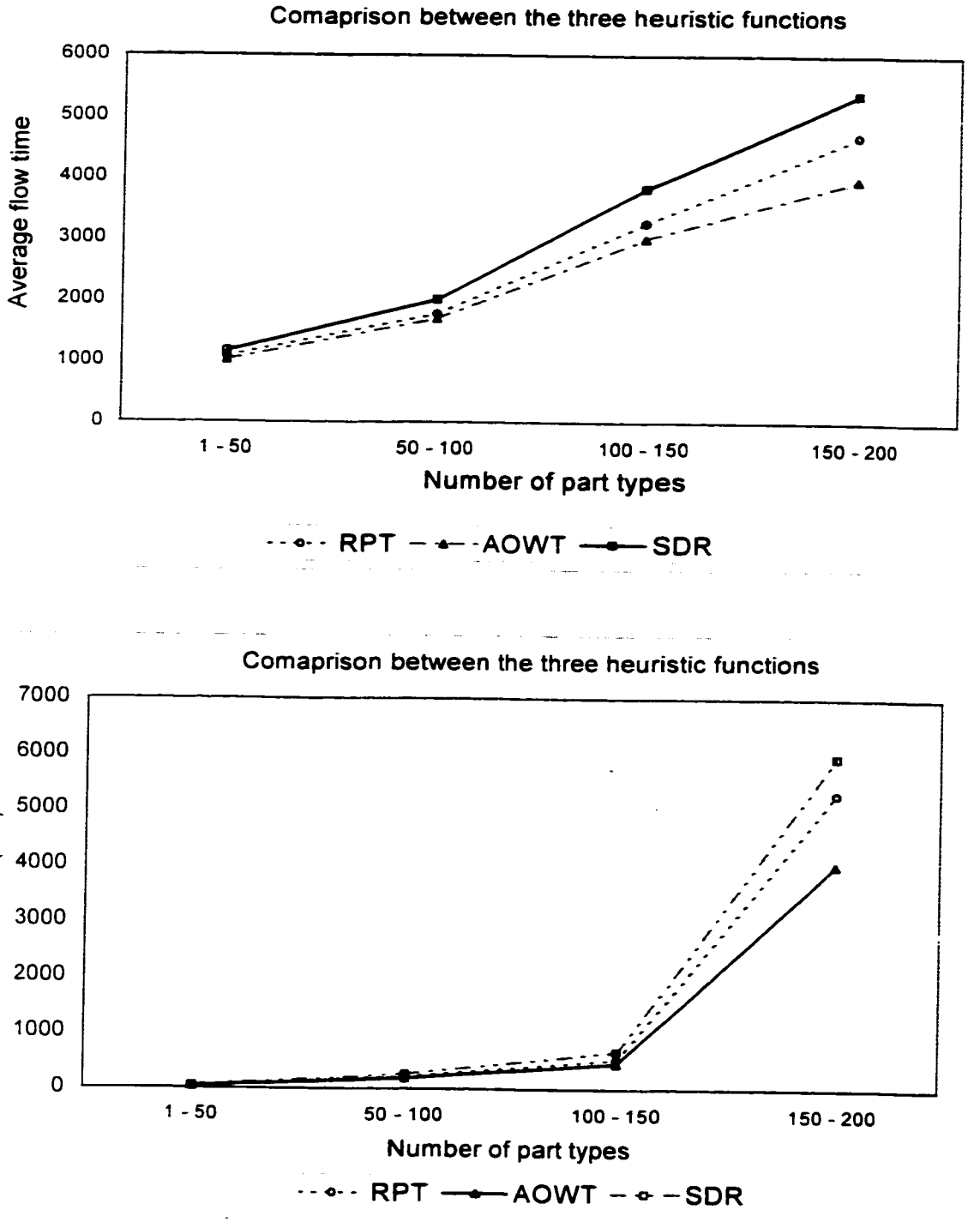


Figure 4.11: Comparison between the three heuristic functions

Table 4.3: Percentage of difference in average flow time and solution *CPU* time at the highest and lowest value of *MF* using the three heuristic functions.

Heuristic function	% Difference of average flow time[1]				% Difference of solution *CPU* time[2]			
	Number of part types				Number of part types			
	1-50	50-100	100-150	150-200	1-50	50-100	100-150	150-200
RPT	14.8	21.1	35.0	11.3	30.3	47.1	83.5	56.2
AOWT	6.3	13.2	19.4	11.1	54.8	80.2	138.6	54.7
SDR	4.0	11.3	14.9	10.3	51.7	76.1	126.7	53.6

[1] (Average flow time at the highest *MF* value-Average flow time at the lowest *MF* value)/Average flow time at the lowest *MF* value *100.

[2] (Solution time at the lowest *MF* value-Solution time at the highest *MF* value)/Solution time at the highest *MF* value *100.

Table 4.4: Comparison of results obtained using the proposed method and Ramaswamy and Joshi (1996) method.

Method		Average flow time	CPU time (sec)	Computing machine
Ramaswamy and Joshi (1996) method	Mathematical model only	332.5	67.0	IBM ES/3090-600S
	Mathematical model with heuristic	336.2	0.71	
Proposed method		332.5	0.06	32 MB Ram/166MHz PC Pentium

4.5 CONCLUSION

An efficient scheduling algorithm for finding an optimal or near-optimal deadlock-free schedule (minimal makespan or average flow time) in FMSs has been presented. It is shown how structure analysis of Petri nets can be used to develop an efficient truncation technique for the proposed scheduling algorithm. Another efficient truncation technique, that is based on the structure of the PN model and the concepts of delay and non-delay scheduling, was also proposed to enhance the ability to develop

deadlock-free schedule for large problems. The algorithm is based on the depth-first backtracking search technique. The efficiency of the proposed approach is due to the following facts:

- The powerful ability of Petri nets to detect good behavior properties of the system such as deadlock-freeness, in addition to their strong modeling capability.
- The size of the reachability graph is effectively reduced by using the truncation technique based on the siphon concept. Indeed, the siphon truncation technique enhances the ability to develop deadlock-free schedules, of systems with high number of deadlocks (i.e., uncontrolled siphons), which cannot be obtained by standard scheduling Petri net approaches.
- The search effort was dramatically reduced using an efficient heuristic based on an appropriate *UCF* value (e.g. the CPU time was reduced from 2178.2 seconds to 285.1 seconds at 100-150 part types when UCF changed from 0 to 1).
- Although the worst-case complexity of the algorithm is exponential, it is shown, through application to several case studies, that it has reasonable responsiveness and the termination of the algorithm is guaranteed.

Three heuristic functions for the optimization of average flow time in flexible manufacturing systems scheduling have been presented. These are the Remaining Processing Time (*RPT*), Average Operation Waiting Time (*AOWT*), and Dispatching Rules (*SDR*). A performance evaluation of the developed heuristics has been conducted using randomly generated test cases. The developed heuristics can be easily modified to include other factors such as machines utilization and due date related measures. Since

they are based on estimating the completion time of every unscheduled part. Each heuristic function is equipped with a parameter (*MF*) to provide a trade-off between the solution quality and the search effort. This approach facilitates solving large problems by relaxing the optimality condition in order to make the search effort reasonable. It is recommended to use low values of *MF* especially in the absence of constraints on solution time and memory capacity.

The Average Operation Waiting Time (*AOWT*) heuristic outperformed the Remaining Processing Time (*RPT*) and Dispatching Rules (*SDR*) functions with respect to the obtained average flow time and the solution CPU time. The increase in the modification factor (*MF*) dramatically decreases (average of 71%) the required solution time without sacrificing the solution quality (average of 14%). In addition, the newly developed heuristics helped in scheduling larger problems compared to those reported in the literature.

It can be seen that Petri nets are more suitable to deal with the deadlock-free scheduling problem than other scheduling approaches such as mathematical programming. In fact, the deadlock states are explicitly defined in the Petri net framework. Thus no equation is needed to describe deadlock avoidance constraints to derive deadlock-free schedules as done using mathematical programming techniques.

The results presented in this research constitute a step towards tackling efficiently the deadlock-free scheduling problem in FMSs and the development of efficient deadlock-free schedules for large FMSs. The research in this direction is still an open area. Therefore, future research and recommendations are presented in Chapter 7.

CHAPTER 5

PERFORMANCE EVALUATION OF MANUFACTURING SYSTEMS EXHIBITING DEADLOCKS

5.1. INTRODUCTION

The flexibility of manufacturing systems is required nowadays to deal with external and internal uncertainties. External uncertainties (market changes) put a lot of pressure on the organization's management. Manufacturing systems should have a reasonable degree of flexibility in order to respond to these uncertainties. These uncertainties may include the requirements to change the products' mix, production volumes, and technical features of the products. Moreover, flexibility helps in coping with internal uncertainties such as machine breakdowns, and shortage of materials. The response time to these uncertainties must be fast and cost effective.

Toni and Tonchia (1998) summarized the conditions that determine mostly the request for flexibility as follows:

1- The demand variability (random or seasonal);

2- Shorter life cycle of products and technologies;

3- Wider products range;

4- Increased customization; and

5- Shorter delivery time.

Flexibility attracted recent attention from researchers to better understand and clarify its concept. As a result, many definitions emerged in the literature. Most of the definitions, which are related to the flexibility of manufacturing systems, are based on the

notion of adaptability to uncertainties. Other definitions are related to other disciplines that are of no direct interest in this research (see Toni and Tonchia (1998)). Mandelbaum (1978) defined flexibility as "the ability to respond effectively to changing circumstances". Zelenovich (1982) defined manufacturing flexibility as "the ability of manufacturing systems to adapt to changes in environmental conditions and in the process requirements". Slack (1987) defined flexibility as "the capacity of a system to assume different positions or to assume a certain number of different states". A comprehensive definition is presented in Toni and Tonchia (1998). They defined flexibility as "the ability to change or react with little penalty in time, effort, cost, or performance". This definition takes into consideration the time, cost, and performance factors especially when assessing the value of the flexibility of manufacturing systems in the current competitive and dynamic market.

As mentioned earlier, manufacturing flexibility is essential to adapt to market and internal uncertainties. However, to convince production managers of its importance and to better understand its concept, there should be quantitative measures of it. Hence, manufacturing organizations could consider it one of their objectives. Another advantage of measuring flexibility is the ability to compare different alternatives of manufacturing policies. This research direction attracted the attention of many researchers, and still represents a challenge for further research.

The performance of systems that exhibit deadlocks will be analyzed, in the following sections, under different levels of routing flexibility using Petri nets. The effect of other factors such as system loading, machine failure rate and processing time

variability will be taken into consideration. The factorial design of the experiments will be introduced and the ANOVA method will be used for the analysis.

Section 5.2 introduces the different classifications and measuring schemes of flexibility found in the literature. Some modeling and measuring approaches of routing flexibility are described in section 5.3. Designs of experiments and results analysis are introduced in section 5.4. Finally, section 5.5 is devoted to conclusions and future research.

5.2. FLEXIBILITY CLASSIFICATIONS

There are many classification schemes of manufacturing flexibility in the literature. Buzacott (1982) based his classification on the uncertainty type with which the manufacturing system adapts itself. He defined two types of flexibility, job flexibility and machine flexibility. Job flexibility was defined as the ability of the system to cope with changes in the jobs to be processed. Machine flexibility was defined as the ability of the manufacturing system to cope with machine disturbances (e.g. breakdowns).

Slack (1987), in his observations on flexibility, claimed that managers identify four main types of flexibility:

- Product flexibility: the ability to introduce novel products, or to modify existing ones.
- Mix flexibility: the ability to change the range of products made within a given time period.
- Volume flexibility: the ability to change the level of aggregate output.
- Delivery flexibility: the ability to change planned or assumed delivery dates.

Sethi and Sethi (1990) defined flexibility as "the ability of the manufacturing system to configure itself to produce efficiently different products of acceptable quality". They mentioned the following types of flexibility:

1- Machine flexibility:

Machine flexibility refers to the various types of operations that the machine can perform without requiring a prohibitive effort in switching from one operation to another. Chen and Chung (1996) claimed that machine flexibility is fundamental for process, product and operation flexibility.

In order to measure machine flexibility, Sethi and Sethi (1990) specified two aspects. The first can be measured by the number of different operations that a machine can perform without requiring more than a specified amount of effort [Brill and Mandelbaum (1989), and Nagarur (1992)]. The second can be measured by the effort (time and/or cost) required in switching from one operation to another [Son and Park (1987), and Chandra and Tombak (1992)].

Machine flexibility allows small batch sizes (lower inventory cost), higher utilization, ability to produce complex parts, shorter lead times for new product introduction, and improved product quality [Gupta and Somers, 1992). In order to achieve machine flexibility, the following factors should be taken into consideration:

- Numerical control;
- Number of axes;
- Easily accessible programs;
- Efficient part-loading and tool-changing devices;

- Size of the tool magazine;
- Sufficient pallets and fixtures;
- Automatic chip removal;
- Adaptive control to optimize metal removal;
- Diagnostic software; and
- Integration with CAD/CAM.

2- Material handling flexibility:

The flexibility of a material handling system is its ability to move different part types efficiently for proper position and processing through the manufacturing facility it serves.

Several researchers have suggested different criteria for measuring material handling flexibility. Chatterjee et al. (1987) defined a universal material handling system that can link every machine to every other machine. Then the material handling flexibility of a given system was expressed by the ratio of the number of paths that the system can support to the number of paths supported by the universal system. Stecke and Browne (1985) ranked the following systems in order of increasing flexibility: belt conveyor, powered roller conveyer, power-and-free conveyers, monorails, towlines, and automated guided vehicle systems. Newman (1986) suggested using general-purpose fixtures to increase material handling flexibility. Material handling flexibility increases machine utilization and reduces the work in process (WIP). To achieve the material handling flexibility the following factors should be presented:

– Automated guided vehicles, robots and computer control;

- General-purpose fixtures;
- Automatic tool changers and multi-axis robots; and
- Robots with flexible grippers and intelligent interface (sensors, and vision).

3- Operation Flexibility:

Operation flexibility of a part refers to its ability to be produced in different ways. Sethi and Sethi (1990) considered it as a part property. Sethi and Sethi (1990), Chaterjee (1987), and Browne et al. (1984) had the same definition of operation flexibility. According to their definition, operation flexibility constitutes both of sequencing flexibility and flexible processing mentioned in Lin and Solberg (1991).

Operation flexibility can be measured by the number of different processing plans of part fabrication. It contributes to various system flexibility types, especially to the routing flexibility. Therefore, it increases machine utilization and improves scheduling performance. The following factors should be taken into consideration in order to achieve the operation flexibility:

- Efficient design that allows the part surfaces to be easily accessible;
- Using modular parts for assembling the part; and
- Generating alternative process plans using computer-aided process planning.

4- Process Flexibility:

Process flexibility of a manufacturing system relates to the set of part types that the system can produce without major setups. It is called part mix flexibility in Melcher and Booth (1987).

Many methods of measuring process flexibility are suggested in the literature. Most of them are based on the idea of measuring the number of part types that the system can produce without major changeover. Process flexibility reduces the batch size and inventory cost. Process flexibility is related to the following types of flexibility:

- Machine flexibility;
- Operation flexibility; and
- Material handling flexibility.

5- Product Flexibility:

Product flexibility is the ease with which new parts can be added or substituted for existing parts. The main difference between product flexibility and process flexibility is that, no additional setup is added due to process flexibility but additional setup is allowed in product flexibility. Product flexibility can be measured by estimating the time and cost required to introduce a new product into a current part mix. Product flexibility contributes to fast response to market changes. The following factors should be incorporated to achieve the product flexibility:

- Machine flexibility;
- Operation flexibility;
- Material handling flexibility;
- Efficient CAD/CAM interface;
- Computer aided process planning (CAPP);
- Group technology (GT);
- Rapid exchange of tools and dies; and

- Flexible fixtures.

6- Routing Flexibility:

Routing flexibility of a manufacturing system is its ability to produce a part by alternate routes through the system. Chen and Chung (1996) measured routing flexibility by the relative ratio of the total number of feasible routes of all part types within a set to the total number of part types of this set.

Routing flexibility allows for efficient scheduling of parts via improved balance of machine loads, reduction in WIP, and absorption of negative effects of machines breakdowns, late receipts of tools, or occurrence of part defects on system performance. It helps meet customer's due dates and facilitate capacity expansion. The following factors should be implemented in order to achieve routing flexibility:

- Multipurpose machines;
- Similar machines pooling into groups;
- Flexible fixture devices;
- Efficient tool management system;
- Flexible labor;
- Real-time scheduling or rescheduling;
- Flexible material handling; and
- Design features of the parts (to allow the same operation to be manufactured by different methods).

7- Volume Flexibility:

Volume flexibility of a manufacturing system is its ability to be operated profitably at different overall output levels. Gerwin (1987) measures it by the ratio of average volume fluctuation over a given period of time to the production capacity limit. Volume flexibility maintains market share in uncertain demand level. Volume flexibility can be achieved by considering the following factors:

- Highly automated flexible manufacturing system (FMS); and
- Efficient subcontracting network.

8- Expansion Flexibility:

Expansion flexibility of a manufacturing system is the ease with which its capacity and capability can be increased when needed. Expansion flexibility is different from volume flexibility in its concern with capacity only (maximum level of production rate). Expansion flexibility is a growth strategy while volume flexibility is a survival strategy. Carter (1986) suggested measuring expansion flexibility by the overall effort and cost needed to add a given amount of capacity. Expansion flexibility can result in reducing implementation time and cost (direct and indirect cost of interruption) for new products, variations of existing products, or added capacity. To achieve the expansion flexibility the following factors should be taken into consideration:

- Small production units;
- Modular manufacturing system;
- Having multipurpose machinery;
- Flexible material handling system;

- High level of automation: and
- Infrastructure that supports growth.

9 - Program Flexibility:

Program flexibility is the ability of a system to run virtually unattended for a long enough period. Sethi and Sethi (1990) suggest measuring it by the expected percentage of uptime during the second and third shifts.

10 - Production Flexibility:

Production flexibility is the universe of part types that the manufacturing system can produce without adding major capital equipment. Production flexibility can be measured by the size of the universe of parts the system is capable of producing.

Performance of the system can be evaluated at different levels of flexibility to enable the managers to set the amount of flexibility in order to get a reasonable performance. In the literature, there are some attempts to analyze the system performance by varying a certain type of flexibility.

To my knowledge, there has been no attempt, in published literature, to study the effect of flexibility in systems exhibiting deadlocks. We were one of the first groups to study this issue [Abdallah et al. (1998)]. In this research, some measures of the considered flexibility type are used to guide the design of experiments. In addition, the factors that may affect the system performance in the presence of flexibility are taken into consideration. The performance of manufacturing systems that exhibit deadlocks are

analyzed under different levels of routing flexibility using Petri nets. The effect of other factors such as system loading, machine failure rate and processing time variability are considered. The factorial design of experiments is introduced and ANOVA method is used for analysis.

5.3. MODELING ROUTING FLEXIBILITY

There are three types of flexibility that can be related to part manufacturing. Lin and Solberg (1991) considered them as different levels of routing flexibility:

- No routing flexibility: Each part must be processed according to a single sequence of operations that must be conducted at predetermined unique machines.
- Fixed sequencing: The operations must be performed in fixed sequence (e.g. drilling - milling - turning); however, there could be more than one machine capable of performing any given operation.
- Flexible sequencing: Certain operations can be performed in arbitrary order; however, the operations are fixed, that is, there is only one-way to achieve a given machined feature (e.g. using a cutter of a certain dimension to machine a certain feature).
- Flexible processing: At this level, neither the operation nor the sequencing is fixed. Therefore, it is possible to produce the same manufacturing features with alternative operations, or sequence of operations.

There are many modeling schemes of these flexibility types reported in the literature. Hancock (1989) suggested the use of a product process network where part manufacturing options are modeled using a directed network with nodes representing machine processes and arcs representing flow precedence constraints. Each complete

path through the network represents a feasible process plan. However, the process network does not explicitly distinguish between operations, or manufacturing features, and machines [Benjaafar and Ramakrishnan (1996)].

Lin and Solberg (1991) introduced a similar part process network based on AND-OR digraphs. All possible part operations are represented by nodes and all precedence relations between operations by directed arcs. Each node carries information about the processing requirements of the corresponding operation. Nodes are classified as being AND nodes, OR nodes and virtual AND/OR nodes. An AND node in the graph means that all of its children must be performed. An OR node means that exactly one of its children operations is to be performed. A virtual AND-OR node has zero processing time and is used to handle cases when both AND arcs and OR arcs are needed from a node. This representation does not explicitly differentiate between feasible operations and available machines and may, at least in theory, produce process plans for which no machines exist in the system.

Bengaafer (1995) introduced a set theoretic model that simultaneously captures operation, sequencing, and processing flexibility. A part is described by a set structure $\Sigma = (O, S, M, G_S, G_M)$ where O is the set of all feasible operations associated with the part, S is the set of all ordered subsets of operations with each subset representing a sequence of operations, M is the set of machines in the manufacturing system, G_S is an operation successor generator G_S: $S \rightarrow O^*$ that maps an operation sequence to the set of operations consisting of all its possible immediate successors, and G_M is a machine generator G_M: $O \rightarrow M^*$ that maps an operation in O to a subset of machines in M representing all those that carry out the operation.

Petri nets are comprehensive and efficient modeling tools that can be used in many areas of manufacturing systems such as performance evaluation, optimization, and real-time control. Therefore, Petri Nets will be introduced as a tool for modeling and representing the routing flexibility. Two levels of routing flexibility [Lin and Solberg (1991)] will be introduced in this section. The first one is the case of a part that has multiple fixed linear sequences. If this level has an order within Lin and Solberg's (1991) classification of routing flexibility levels, it should be put right after the "No routing flexibility" level. This type of routing flexibility might improve the quality of the generated off-line schedule. On the other hand, it has no effect on the system performance where there is uncertainty. Many authors explored routing optimization by determining the best route for each part type through the system while generating schedules off-line [Chang et al. (1984), Wittrock (1988)].

Example 1:

Consider an FMS composed of three machines M1, M2, and M3, and a robot R that produces two parts, Job1 and Job2. The robot is used for loading/unloading the parts to/from the machines. Table 3.2 shows the production plans for both part types.

Table 5.1: The production plan of the illustrative example.

JOB	SEQUENCE	OPERATIONS						
		1	2	3	4	5	6	7
Job1	1	R	M1	R	M2	R	--	--
	2	R	M2	R	M1	R	--	--
Job2	1	R	M3	R	M1	R	M2	R
	2	R	M3	R	M2	R	M1	R
	3	R	M2	R	M1	R	M3	R

Figures 5.1 and 5.2 show the Petri net sub-models of Job1 and Job2 respectively. The combination of both models with the shared resources represents the overall Petri net model for the system. The required resources are not shown in Figure 5.2 but can be added in the same manner shown in Figure 5.1.

As shown in Figure 5.1, Job1 has two fixed routes. Each route is represented with a state machine (e.g. P0, T0, P1, T1, P2, T2, P3, T3, P4, T4, P5, T5, P11 represents the first route). The same can be noted from Figure 5.2 for Job2.

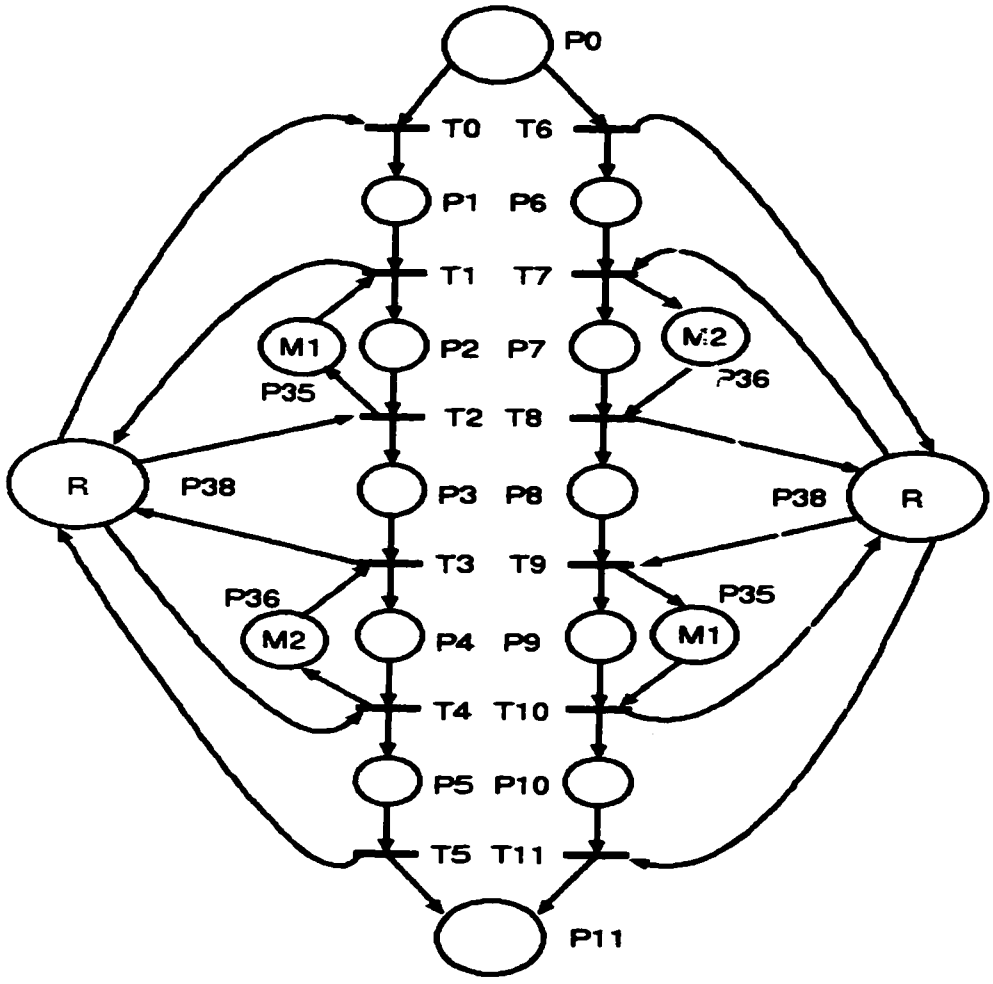


Figure 5.1: Petri net sub-model of Job1

The second level is the case of a part that has fixed sequencing. This level mentioned in Lin and Solberg's (1991) classification of routing flexibility is higher than the former one (i.e. Example 1). A manufacturing system with this level of routing flexibility is expected to generate higher quality of production schedules and better performance when a disturbance occurrs. Wilhelm and Shin (1985) compared the results of using fixed sequencing process plans with the performance achieved by using no routing flexibility process plan for an FMS through simulation. Their results showed that the system with routing flexibility can reduce flow time and increase machine utilization.

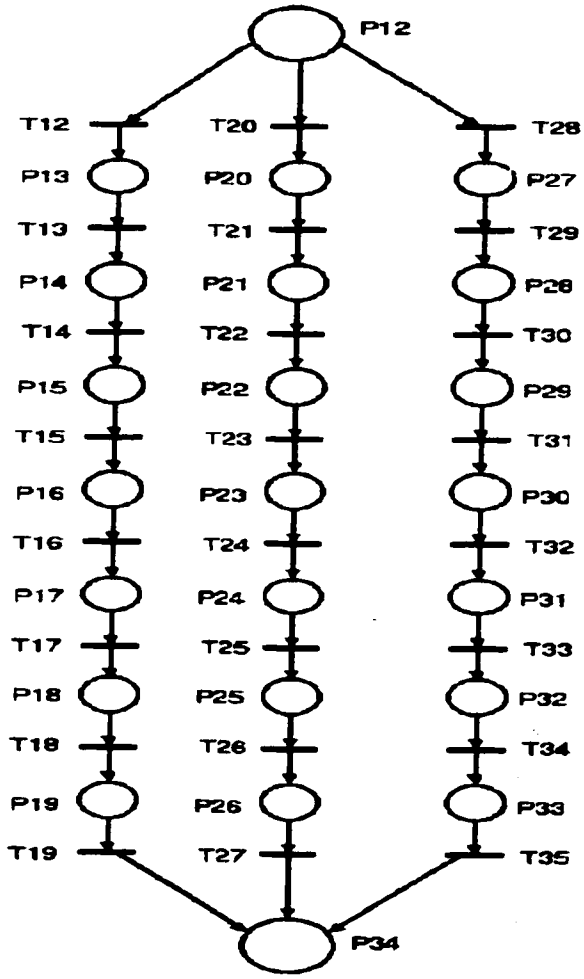


Figure 5.2: Petri net sub-model of Job2

Example 2:

Consider a part type that requires three operations on three machines. Table 5.2 shows the operations sequence of this part type. The Petri Net model of this part is presented in Figure 5.3. The resource places are not shown in this Petri Net model for the sake of simplifying its visual appearance. As it can be noted from Figure 5.3. the first operation has two alternative machines. Therefore. it is represented by two operation places. P2 and P3. Following the same logic. the Petri Net model for this part type was developed.

Table 5.2: Operations sequence

Parts	Operations		
	O1	O2	O3
1	M2 M3	M1 M2	M1 M3

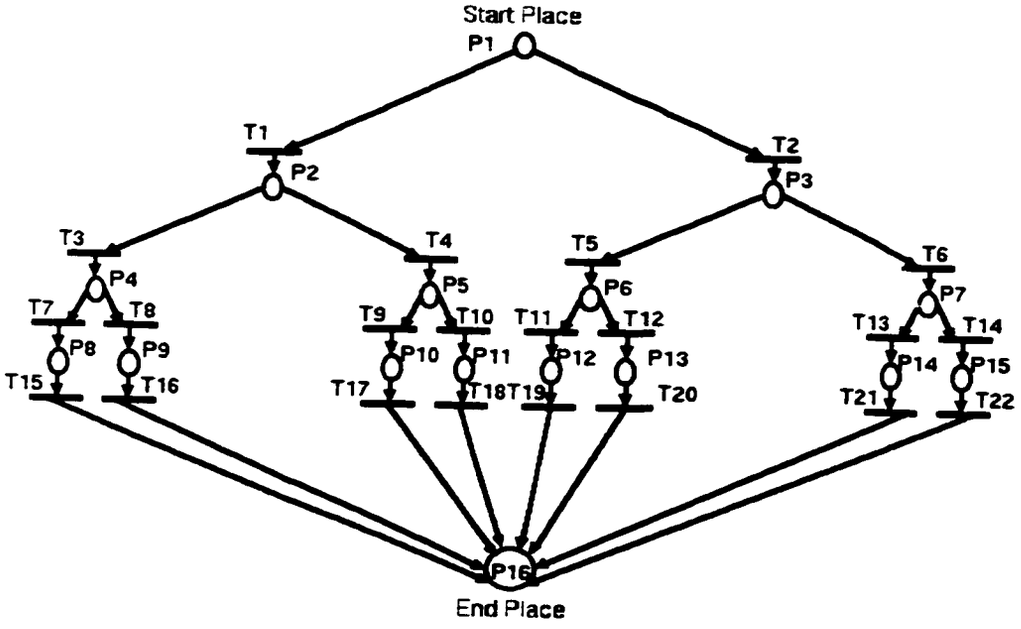


Figure 5.3: Petri Net model for a part with fixed sequencing

5.3.1 MEASURING ROUTING FLEXIBILITY

It is impossible to control something that can not be measured. Therefore, several attempts have been made to understand and measure flexibility qualitatively and quantitatively. The qualitative trials were based on technological and business issues that may affect manufacturing flexibility [Carter 1986, and Browne et al. 1984). On the other hand, the quantitative measures were based on mathematical modeling [Brill and Mandelbaum 1989, Son and Park 1987, Nagraur 1992).

The main concern of this research is routing flexibility. Therefore, some of the models and approaches used to measure it will be presented in this section. Sarker et al. (1994) observed three methods of measuring routing flexibility:

(a) Based on the number of routes available for the processing of a part type;

(b) Based on the efficiency of each route; and

(c) Using availability/utilization of resources.

Nagraur (1992) defined the measure of routing flexibility as a proportion of all potential routes that are available. The measure used by Sinha and Wei (1992) was based on the average number of alternative routes available for processing each part. Chen and Chung (1996) followed the same scheme of measuring routing flexibility. Their measure for routing flexibility, RF, is given by:

$$RF = \frac{\sum_{i}^{H} r_i}{H} \qquad (5.1)$$

Where: H = number of part types; and

r_i = number of feasible routes that part type i can flow through the system.

Bernardo and Mohamed (1992) defined two measures of routing flexibility: i) actual routing flexibility, which is a measure of the number of existing production routes that could be used, and ii) potential routing flexibility, which is the total available routes to make given part. Gerwin (1987) proposed a rerouting flexibility measure that evaluates the efficiency with which alternative routings can be assigned in case of machine stoppage. This efficiency was based on the drop in production rate when machine stoppage occurs.

5.4. EXPERIMENTAL DESIGN OF ROUTING FLEXIBILITY EVALUATION

Most of the flexibility types are dependent. Therefore, improving one flexibility type may improve other flexibility types. For example, if machines have high flexibility this leads to higher routing flexibility. In addition, the increase in routing flexibility may give a better chance for expansion flexibility, and product mix flexibility. In the scheduling of flexible manufacturing systems, routing flexibility is one of the most important system features that should be taken into consideration. Previous research has proven that routing flexibility improves the performance of systems that do not exhibit deadlocks.

The major objective of the experiments conducted in this research is to evaluate the effect of routing flexibility on the quality of obtained schedules in systems that exhibit deadlocks. The literature has been carefully reviewed in order to select the

configuration of the FMS. In the following paragraphs, some of the reviewed papers will be presented.

Chen and Chung (1991, 1996) used an FMS that consists of 6 machines and produces 3 part types. Each part type requires 1 to 6 operations. The batch size of each part type is generated from a uniform distribution with a range of 16 to 24 units.

Caprihan and Wadhwa (1997) used a hypothetical FMS that comprises 6 machines. They claimed that this size was considered to occur most frequently [Shanker and Tzen (1985)]. The FMS produces 6 part types. Each part type requires between 4 and 6 operations with average equal to 5.

Lin and Solberg (1991) used an FMS that consists of three milling machines, two drilling machines, load/unload station, a central buffer and a set of automated guided vehicles (AGVs). Only one part type with different process plans was used in the experimentation.

Schmidt and Ranta (1989) found in their survey that 46% of all FMSs include a total of 2-4 NC-machines. Figure 5.4 shows the FMS distribution relative to the total number of NC-machines in each system. In addition, they found that three fourth of the FMSs produce parts by batches with less than 100 units each, and one third of FMSs produce by batches of no more than 10 units a batch. Another indicator studied in Schmidt and Ranta's (1989) survey is product variants. They found that 30% of the FMSs produce no more than 10 different part types. Figures 5.5 and 5.6 show the FMS distribution relative to the average batch sizes and number of product variants.

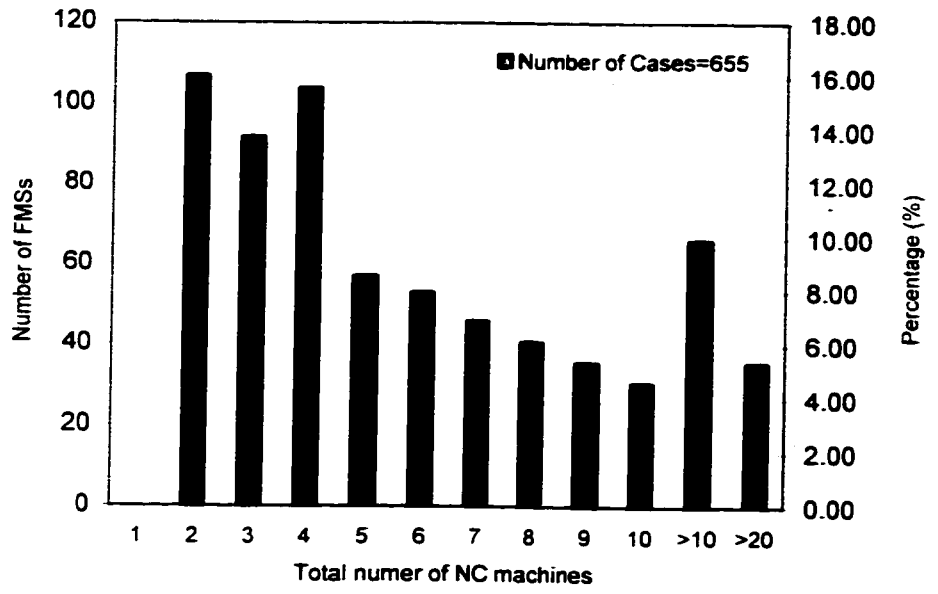


Figure 5.4: FMS distribution relative to the total number of NC-machines in the system

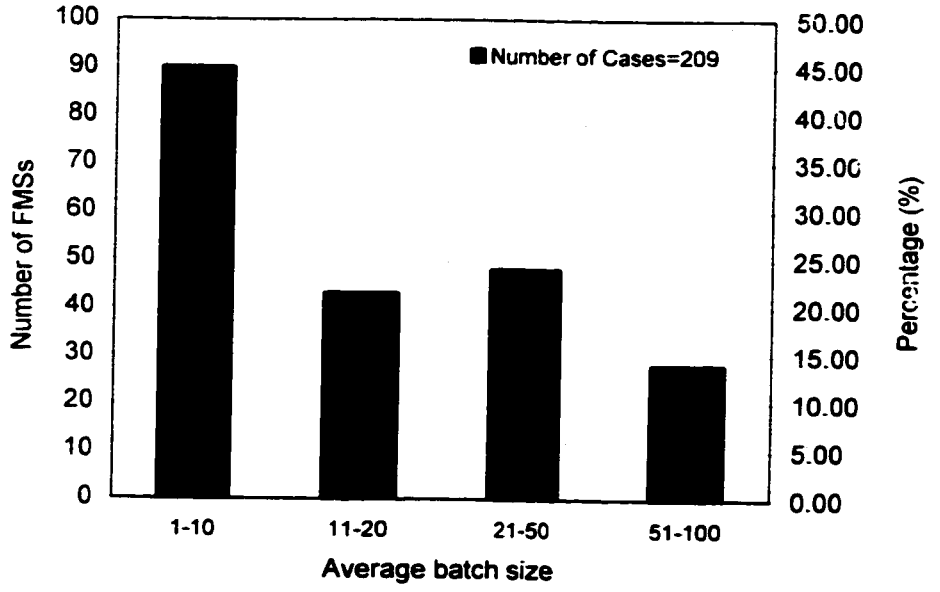


Figure 5.5: FMS distribution relative to the average batch sizes

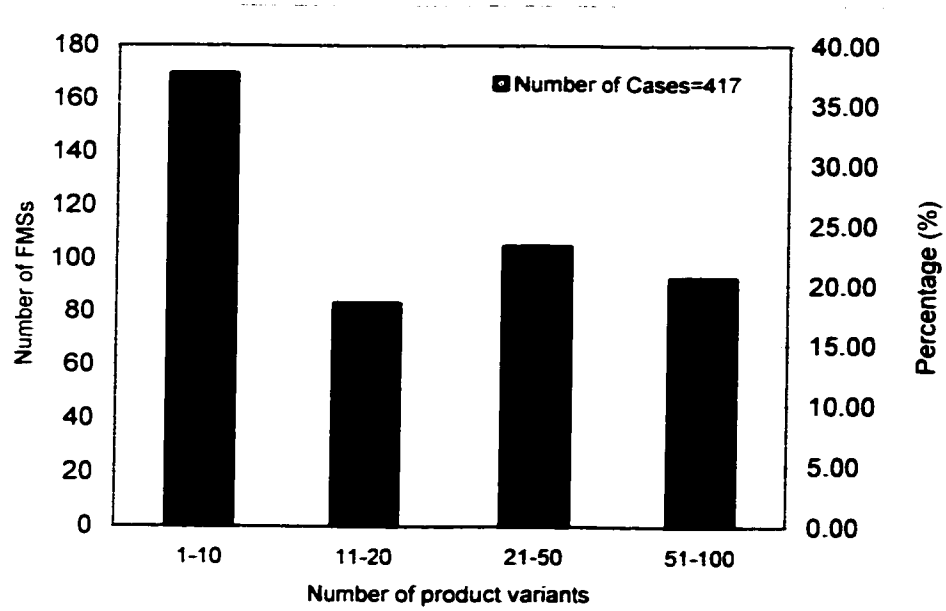


Figure 5.6: FMS distribution relative to the number of product variants

A hypothetical FMS consists of Five machines and produces 5 part types was used for experiments. The detailed description of this system is presented in data set 3, Appendix 2. This system configuration is used to generate the test cases for the different combination of levels of the considered factors. In addition to the routing flexibility factor, three other factors were taken into consideration. These are system loading, machine failure rate, and processing time variability. The full factorial design is shown in Table 5.3. Five randomly generated test cases were used for each combination level of the considered factors. Hence, 360 runs were performed and analyzed using the SPSS software. The algorithm presented in Chapter 4 was used for optimizing the average flow time. The effect of routing flexibility and its interaction with other factors were the main concern of this study. Routing flexibility was measured using equation 1 [Chen and

Chung,1996]. The system loading is measured by the total number of processed parts. It is assumed that all part types have the same batch size. The value of mean time between machine failures (*MTBF*) is used to represent system reliability. The *MTBF*, in these experiments, is related to the operation average processing time (*apt*).

The adopted measure of routing flexibility [Chen and Chung,1996] in this research represents the average number of routes per part. The number of routes of a part is measure by multiplying the number of alternative machines (machines that can perform an operation) of each of its operations. For example, if a part has 5 operations and each operation can be performed on two machines, then the total number of alternative routes of this part is 32 (i.e. 2^5). The system that was used for the experiments consists of five machines and produces five part types. A typical randomly generated case of the experiments could have number of operations per part type as of 2,3,4,5, and 6 over the five part types. Assuming that every operation can be performed on any of the five machines. Then a maximum routing flexibility (RF) factor value equals 3905 will be obtained as demonstrated in Table 5.4. Case 2, of Table 5.4, represents a very optimistic condition where a 75% of the operations (15 out of 20 operations) has an alternative machine. On the other hand, Case 3 represents a very pessimistic state where each operation can be performed on only one machine. Therefore, in the experiments, the levels of the Routing Flexibility factor were divided into four intervals that lie in the interval of (1 to 14). In the literature, the same measure were used in two papers [Chen and Chung,1996, and Caprihan and Wadhwa, 1997]. Chen and Chung (1996) used four levels of the routing flexibility with RF values of 1, 4, 10, and 16 respectively. While

Caprihan and Wadhwa (1997) used five levels of the same factor with RF values of 1, 2, 3, 4, and 5 respectively.

MTBF of a machine is defined as the mean value of the times between consecutive failures. The MTBF can be computed as the ratio of the total cumulative observed time to the total number of failures [Smith, 1985]. In the literature, the Exponential, Erlang, Weibull, and other distributions are used to model the time between failures. Lie et al. (1987) mentioned more than 50 papers used the Exponential distribution in modeling MTBF. Erlang-4 distribution was used in Burton et al. (1989), and Banerjee and Flynn (1987). Lie et al. (1987) mentioned that some other distributions are also used such as, Normal, Log-Normal, and Weibull.

. In real-life situations, the machine failure mode can be identified and consequently the appropriate distribution can be defined using historical data. The concern of this study is not the manufacturing system reliability and maintainability but the scheduling quality with the existence of machine breakdowns. In most cases, the scheduling of a system is generated to cover time horizon of one week to three months (depending on the system type and nature). Hence, the selected distribution type that represents the machine failures will not have a considerable effect. In addition, the objective of this part is to find a mechanism that generates a number of breakdowns with the corresponding time-spans between them over the generated schedule makespan. In this part, the uniform distribution was selected to represent the machine breakdowns.

The fourth factor, processing time variability (PTV), represents the variation in the required processing time of the considered part mix. Stecke, (1981), and Rovito and Dvorsky, (1984) reported that the operation processing times range from 2 to 45 minutes

when they considered both systems of Sundstrand/Caterpillar DNC line and Cincinnati Milacrom FMS respectively. Shanker and Tzen. (1985) used a uniform distribution of interval (6. 30) as the operation processing time distribution. Chen and Chung.1996 used the uniform distribution of two intervals (16. 24) and (2. 38) as a Low level and High level of the factor Operation Processing Time in his design of experiments. In this research. processing time with low variation is assumed to be uniform distribution of interval (6. 10). and with high variation of interval (4. 26).

The mean number of minimal siphons at each level of routing flexibility is presented in Figure 5.7. Increasing the routing flexibility. for the same system. increases the interaction between the resources. which is reflected by an increase in the number of minimal siphons. Therefore. it is expected that increasing the routing flexibility will increase the probability of the deadlock occurrence. which could negatively affect the system performance.

Table 5.3: The considered factors and their levels.

1- Routing Flexibility (RF):	
1- RF1:	RF = 1
2- RF2:	RF = 3-5
3- RF3:	RF = 8-10
4- RF4:	RF = 12-14
2- Machine Failure Rate (MFR):	
1- MFR1:	MTBF = Uniform(6 x apt. 8 x apt)
2- MFR2:	MTBF = Uniform(10 x apt, 12 x apt)
3- MFR3:	MTBF = Uniform(16 x apt. 20 x apt)
3- System Loading (SL):	
1- SL1:	50 units
2- SL2:	100 units
3- SL3:	150 units
4- Processing Time Variability (PTV):	
1- PTV1:	Uniform(6,10)
2- PTV2:	Uniform(4,26)

Table 5.4: Explanation of the routing flexibility measure.

Part number		**1**	**2**	**3**	**4**	**5**
Number of operations		**2**	**3**	**4**	**5**	**6**
Case 1	Number of routes	5^2	5^3	5^4	5^5	5^6
	Maximum *RF*	$(5^2+5^3+5^4+5^5+5^6)/5=3905$				
Case 2	Number of routes	2x3	2x3x2	2x3x2x2	2x3x2x1 x1	2x3x2x1x 1x1
	RF	(6+12+24+12+12)/5 = 13.2				
Case 3	Number of routes	1	1	1	1	1
	RF	1				

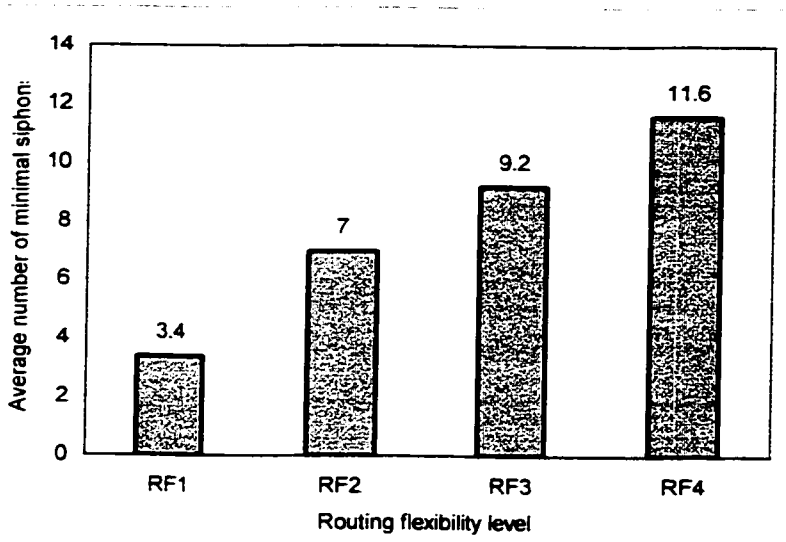


Figure 5.7: Effect of routing flexibility on number of minimal siphons

5.4.1. AUTOMATIC GENERATION OF PETRI NET MODELS

An algorithm for automatically generating the Petri Net models of systems with multiple fixed linear sequences is presented in Chapter 3. In this section another

algorithm for automatically generating the Petri Net models of systems with fixed sequencing is introduced. The case of fixed sequencing, type of routing flexibility, is considered in the experiments. The operations must be performed in fixed sequence; however; there could be more than one machine capable of performing any given operation. An algorithm has been developed to automatically generate the Petri Net model from a given production plan. The following notations were used in the developed algorithm:

N: number of part types;

M: number of resources (machines, buffers, or material handling facilities);

O_i: number of operations of part type i;

Alt_{ij}: set of alternative resource that can perform operation j of part type i;

ro_{ij}: number of elements of the set Alt_{ij} ((i.e.$ro_{ij} = ||Alt_{ij}||$);

Np: total number of places of the Petri Net model;

Nt: total number of transitions of the Petri Net model;

Sp_i: start place of part type i;

Ep_i: end place of part type i;

Opr_i: set of operation places of part type i;

Rp: set of resource places;

IT(p): input transition set of a place p;

OT(p): output transition set of a place p;

The developed algorithm for automating the generation of a Petri net model consists of the following steps:

1- Estimate the total number of places and the total number of transitions of the Petri Net model,

2- Identify the set of input and output transitions of every place.

The places of the Petri Net model are divided into four types: start places, end places, operation places and resource places. Every part type has one start place and one end place that represent the start and end of part processing respectively. An operation place represents the processing of a certain operation of a part type by one of the identified alternative resources. Every resource is represented by one place.

Step 1:

The operation places of the Petri Net sub-model for every part type can be divided into levels as shown in Figure 5.8. The number of levels of the i^{th} part type equals O_i. The number of operation places at each level can be estimated as follows:

$$N_{ij} = \prod_{k=1}^{k=j} ro_{ik} \tag{5.2}$$

Where N_{ij} is the number of operation places at level j of part type i

$$NP = \left[\sum_{i=1}^{i=N} \left\{ 2 + \sum_{j=1}^{j=O_i} N_{ij} \right\} \right] + M \tag{5.3}$$

In the same manner the transitions of the Petri Net sub-model for every part type can be divided into levels, as shown in Figure 5.8. The number of levels of the i^{th} part type equals (O_i+1). It can be noted that every level of operation places is preceded by a corresponding level of transitions with equal number of elements. The last level of

operation places is preceeded by a corresponding level of transitions with equal number of elements.

$$NT = \sum_{i=1}^{i=N}\left[\left\{\sum_{j=1}^{O_i} N_{ij}\right\} + \prod_{k=1}^{O_i} ro_{ik}\right] \quad \textbf{(5.4)}$$

Step 2:

Let i = 1:

2.1 Start place of the i^{th} part type:

$IT(Sp_i)= \phi$;

$OT(Sp_i)$=set of transitions at the 1^{st} level of the i^{th} part type.

2.2 End place of the i^{th} part type:

$IT(Ep_i)$=Set of transitions at the $(O_i+1)^{th}$ level of the i^{th} part type.

$OT(Ep_i) = \phi$.

2.3 Operation places of the i^{th} part type:

For the j^{th} operation place in the k^{th} level of the i^{th} part type, $j=1,2....N_{ik}$ and $k=1,2,...,O_i$:

$IT(j)$={j^{th} transition at the k^{th} level of the i^{th} part type}.

Divide the set of transitions at level $(k+1)^{th}$ into N_{ik} equal-sized sub_sets.Then:

$OT(j)$={the j^{th} sub_set}.

2.4 Resource places:

1. $\forall r: r \in Rp$; do::

1.1. for the k^{th} level of operation places of part type i; k=1.2.... O_i : do:

1.1.1. if ($r \in Alt_{ik}$): do:

1.1.1.1. pr = the position of r within Alt_{ik}

1.1.1.2. $OT(r)=OT(r)+pr^{th}\ T+(pr+ro_{ik})^{th}\ T+(pr+2\ ro_{ik})^{th}\ T+\ldots+(pr+k\ ro_{ik})^{th}\ T$: such that $(pr+k\ ro_{ik})\leq N_{ik}$, where $T \in$ set of transitions at k^{th} level

1.1.1.3. divide the set of transitions at the $(k+1)^{th}$ level into N_{ik} equal-sized sub_sets

1.1.1.4. $IT(r)=IT(r)+pr^{th}$ sub_set$+(pr+ro_{ik})^{th}+$sub_set$+(pr+2\ ro_{ik})^{th}$ sub_set$+\ldots+(pr+k\ ro_{ik})^{th}$ sub_set: such that $(pr+k\ ro_{ik})\leq N_{ik}$

2.5 If (i > N) stop; else i=i+1; go to 2.1;

Illustrative Example

The system presented and modeled in example 2 will be used to demonstrate the developed algorithm. The Petri Net model of the part with the required operation places levels and transitions levels is shown in Figure 5.8.

Sp: P1

Ep: P16

Operation places:

Level #1: {P2,P3}

Level #2: {P4,P5,P6,P7}

Level #3: {P8,P9,P10,P11,P12,P13,P14,P15}

Resource places:

M1: P17

M2: P18

M1: P19

Transitions:

Level #1: {T1,T2}

Level #2: {T3,T4,T5,T6}

Level #3: {T7,T8,T9,T10,T11,T12,T13,T14}

Level #4: {T15,T16,T17,T18,T19,T20,T21,T22}

Step 1:

$N_{11} = 2$

$N_{12} = 2x2 = 4$

$N_{13} = 2x2x2 = 8$

$Np = [\{2+ 2+4+8\}]+3 = 19$

$Nt = [\{2+4+8\}+2x2x2) = 22$

Step 2:

$i = 1;$

2.1 Start place of the i^{th} part type:

$IT(P1) = \phi;$

OT(P1) = { T1,T2};

2.2 End place of the i^{th} part type:

IT(P1) = {T15,T16,T17,T18,T19,T20,T21,T22};

OT(P1) = ϕ;

2.3 Operation places of the i^{th} part type:

Let k = 2 and j = 3:

IT(P6) = {T5} ;

N_{12} = 4, the set of transitions at $(2+1)^{rd}$ level are divided into {(T7,T8),(T9,T10),(T11,T12),(T13,T14)}, then:

OT(P6) = { (T11,T12) };

The rest of the operation places can be treated in the same manner.

2.4 Resource places:

1. let r = P18 (P18 represents M2);

1.1. let k = 2;

1.1.1. True (where Alt_{12} = {M1,M2});

1.1.1.1. pr = 2 (where M2 is number 2 in the Alt_{12}), N_{12} = 4;

1.1.1.2. OT(P18) = OT(P18) + 2^{nd} T+$(2+2)^{th}$ T, T$\in$set of transitions at 2^{nd} level,

OT(P18) = OT(P18) +{T4,T6},

1.1.1.3 N_{12} = 4, the set of transitions at $(2+1)^{rd}$ level are divided into {(T7,T8),(T9,T10),(T11,T12),(T13,T14)}, then:

1.1.1.4 OT(P18) = OT(P18)+ 2^{nd} sub_set+$(2+2)^{th}$ sub_set.

OT(P18) = OT(P18)+{(T9.T10)}+{(T13.T14)}

These steps should be performed for all resource places at the different operation places' levels.

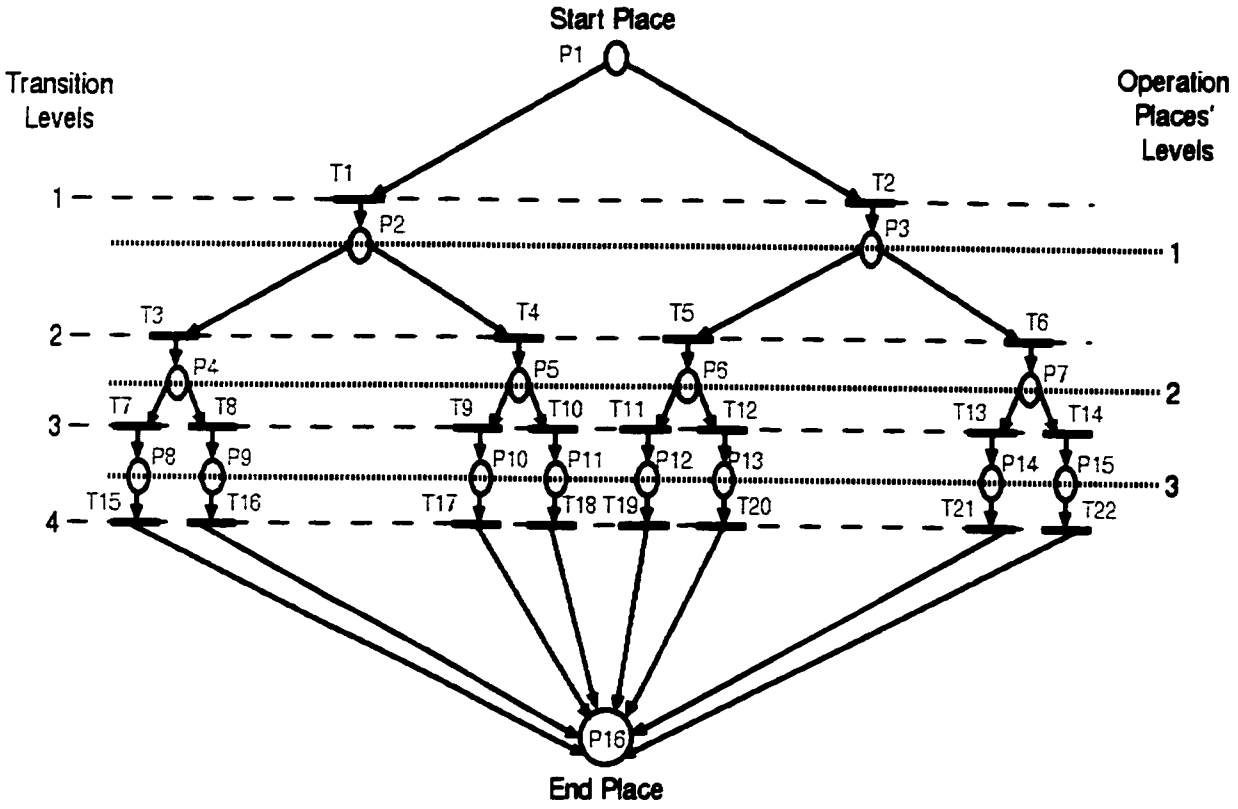


Figure 5.8: The Petri Net model of example 2 with the required operation places' levels and transitions' levels

5.4.2. RESULTS ANALYSIS

The algorithm for automatically generating Petri Net models, presented in section 5.4.1, was used in modeling the randomly generated test cases. The algorithm presented in Chapter 4 was used for optimizing the average flow time of the generated test cases. A

summary of the ANOVA output, by SPSS software, is shown in Table 5.5 (the detailed results of the ANOVA are presented in the Appendix 3). It can be noted from the last column of Table 5.5 that the main effect of all factors is significant at 5% level of significance. In addition, there are significant two-way interaction effects between each pair of routing flexibility (RF), system loading (SL), and processing time variability (PTV) factors at 5% level of significance. Figure 5.9 summaries the mean of the average flow time at the different levels of the routing flexibility (RF) factor. The mean values of the average flow time at the different levels of RF with the different levels of the other three factors are depicted in Figure 5.10, 5.11 and 5.12 respectively.

From Table 5.4, it can be noted that the third part type represents the average that can be generated in a test case. This part has 4 operations with an average processing time of 11.5 (the average of the two intervals of the Processing Time Variability factor, see Table 5.3). In order to estimate the lower bound of the average flow time of N unit (based on the system loading, see table 5.3) of this part, it is assumed that any operation of this part can be performed on any available machine. A simulation model was developed using Witness software in order to estimate the average flow time at different values of system loading factor (50, 100, and 150 units). A mean value of the average flow time was estimated and plotted as shown in Figure 5.9.

Improvement in the system performance was observed with increasing the routing flexibility level (15%, 33%, and 23% when changing the routing flexibility RF1-RF2, RF2-RF3, and RF3-RF4, respectively). Further analysis of the mean of the average flow time at the different levels of routing flexibility (RF) factor was performed. A summary of the analysis is presented in Table 5.6. It was found that all changes in the mean of the

average flow time (from one level to the next) resulted in a significant effect for the first group at 10% level of significance and 5% for the other two groups. These results contradict the hypothesis that "increasing routing flexibility complicates the Petri net model and creates new deadlocks, which could negatively affect the system performance". This is can be explained as in traditional manufacturing systems, the higher the routing flexibility the higher the chance to balance the load among the resource the better the obtained average flow time.

The greatest improvement in system performance (33%) was observed when routing flexibility increased from "RF2" to "RF3". On the other hand, this improvement dropped to 23% when routing flexibility changed from "RF3" to "RF4".

Most manufacturing flexibility types are correlated. In other words, routing flexibility needs machine flexibility, material handling flexibility and software flexibility. Therefore, the benefits of increasing routing flexibility should be justified by taking into consideration the resulting system improvements as well as the corresponding additional cost. This can be achieved by comparing the marginal benefits due to incremental increase in flexibility with resulting increase in costs.

The interaction effect between routing flexibility (RF) and machine failure rate (MFR) factors is not significant, as can be noted from Table 5.5 and Figure 5.10. The parallelism of the three lines in Figure 5.10 indicates that the main effect of the routing flexibility (RF) factor at the different levels of the machine failure rate (MFR) factor is almost the same. The concern of this study is the off-line scheduling performance rather than real-time scheduling. It is expected that this interaction effect might be significant if

real-time scheduling is considered. Therefore, this issue should attract more attention with real-time scheduling in flexible manufacturing systems.

As stated previously, the routing flexibility (RF) factor has a significant interaction effect with both system loading (SL) and processing time variability (PTV) factors. This can be noted from Figures 5.11 and 5.12. At high system loading, the chance of utilizing the available routing flexibility is higher. Therefore, the effect of routing flexibility at high level of system loading should be greater than the effect at low level of system loading. That justifies the significant interaction effect between the routing flexibility (RF) factor and system loading (SL) factor. In the same sense, the significant interaction effect between the routing flexibility (RF) factor and processing time variability (PTV) factor can be justified. Hence, at high level of processing time variability, the scheduling optimization algorithm finds a much better chance to obtain schedules with higher quality at high levels of routing flexibility than it does at low levels. The highest effect of the RF factor is at the "PTV2" level of the PTV factor. The reduction in mean average flow time was (36.4%) with a change in the RF from "RF2" to "RF3". It is very interesting to note that the low processing time variation significantly reduces the average flow time. Therefore, it is recommended to consider this factor at part grouping and part process plans generation stages.

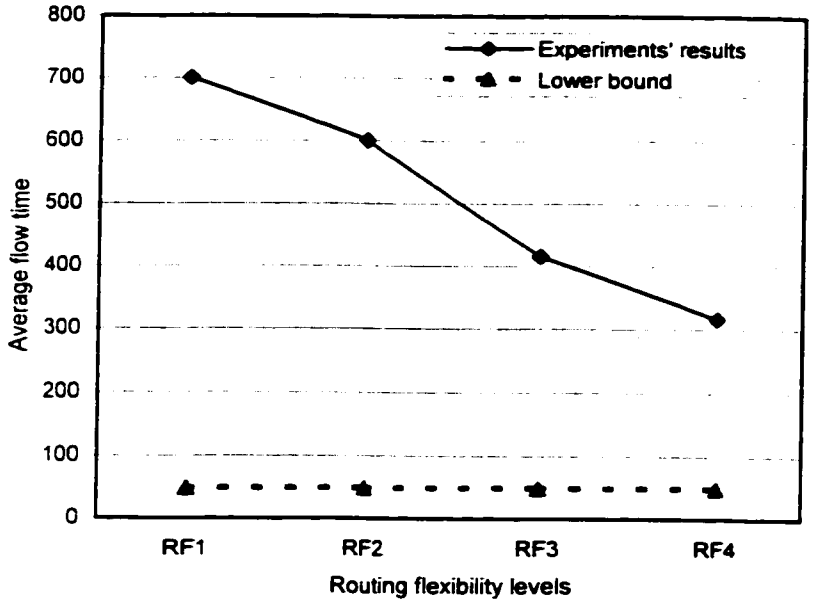


Figure 5.9: Summary of the average flow time at the different levels of the routing flexibility (RF)

Table 5.5: Summary of the ANOVA Results.

Source of Variation	F	Sig of F
Routing flexibility (RF)	176.01	0.000
Machine failure rate (MFR)	10.40	0.000
System loading (SL)	504.25	0.000
Processing time variability (PTV)	559.15	0.000
Routing flexibility (RF) BY Machine failure rate (MFR)	0.79	0.576
Routing flexibility (RF) BY System loading (SL)	16.18	0.000
Routing flexibility (RF) BY Processing time variability (PTV)	31.44	0.000
Machine failure rate (MFR) BY System loading (SL)	0.95	0.433
Machine failure rate (MFR) BY Processing time variability (PTV)	0.82	0.442
System loading (SL) BY Processing time variability (PTV)	47.36	0.000

X BY Y: Interaction effect between X factor and Y factor.
F: The estimated F value by SPSS software [Montgomery, 1976].
Sig. of F: The calculated significance of the estimated F value by SPSS software. If it is less than the assumed significance level, the effect is considered significant at this level.

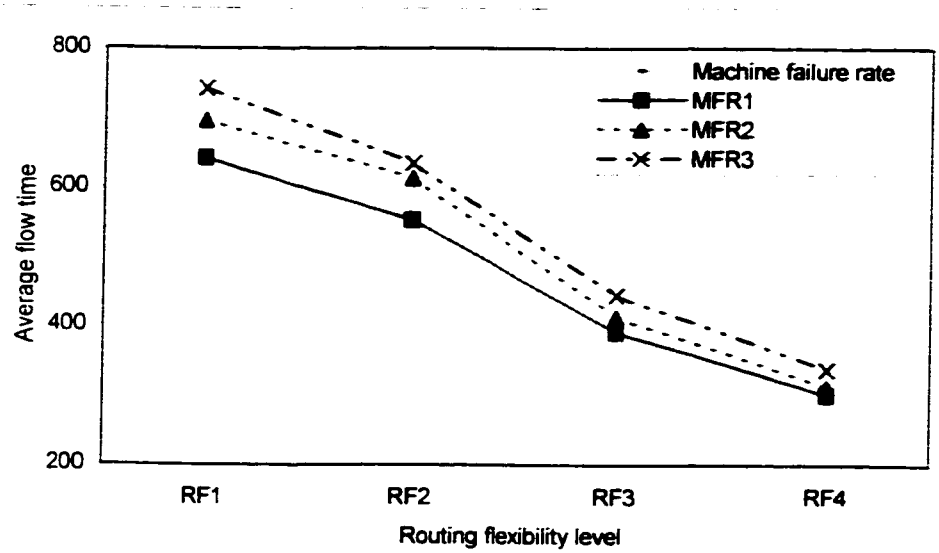


Figure 5.10: The interaction between routing flexibility factor (RF) and machine failure rate factor (MFR)

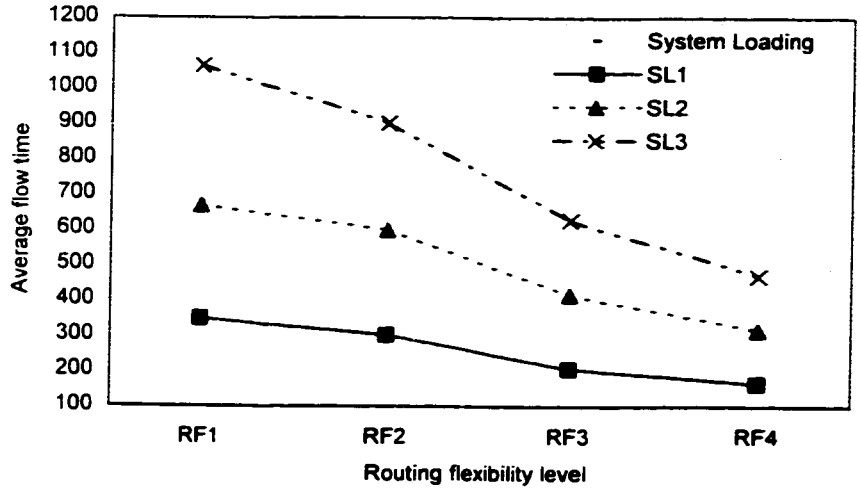


Figure 5.11: The interaction between routing flexibility factor (RF) and system loading factor (SL)

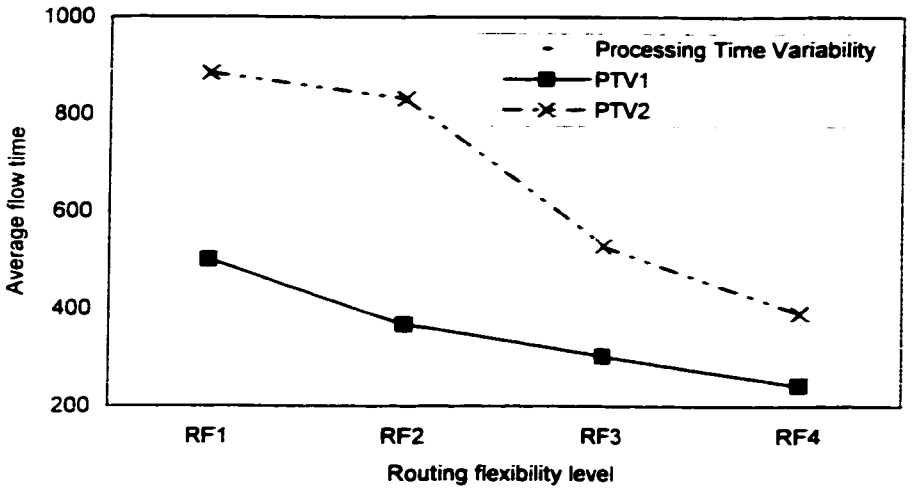


Figure 5.12: The interaction between routing flexibility factor (RF) and processing time variability factor (PTV)

Table 5.6: One Way ANOVA summary of comparison of means of the average flow time.

Comparison of means between of the average flow time	Sig of F
RF1 with RF2 levels	0.0841
RF2 with RF3 levels	0.0001
RF3 with RF4 levels	0.0011

Table 5.7: Percentage of improvement in system performance with changing the levels of routing flexibility at the different levels of SL and PTV factors.

Level Change of RF	Mean	System Loading			Processing Time Variability	
		SL1	SL2	SL3	PTV1	PTV2
RF1- RF2	14.33	13.36	10.42	15.31	26.64	5.92
RF2- RF3	30.72	31.60	30.69	30.45	17.76	36.44
RF3- RF4	23.88	20.54	23.79	25.04	19.93	26.13
Average		21.83	21.63	23.60	21.44	22.83

5.5. CONCLUSION

It was thought that routing flexibility would complicate the Petri net model and create new deadlocks, which in turn could negatively affect the system performance. The results showed that increasing routing flexibility improves the system performance (average flow time) in systems exhibiting deadlocks.

Although increasing the routing flexibility to "RF2", "RF3", or "RF4" improved the system performance significantly, the highest gain was obtained when it was changed from "RF2" to "RF3". Another important point that must be addressed is that, in order to gain the benefits of routing flexibility, manufacturing systems must have flexible fixturing devices, a tool management system and an efficient real-time scheduling controller. Therefore, the benefits of increasing routing flexibility should be justified by taking into consideration the resulting system improvements and the corresponding additional cost. This can be achieved by comparing the marginal benefits due to incremental increase with resulting increase in costs.

It is recommended to have routing flexibility whatever the level of machine failure rate , system loading, or processing time variability. Even though the routing flexibility has its highest effect if the system is at the highest levels of system loading and processing time variability.

Routing flexibility depends on machine flexibility, labor flexibility, material handling flexibility and operation flexibility. Therefore, the relationship between these types of flexibility needs to be firmly established.

The main concern of this study was the effect of routing flexibility and its interaction with other factors on the quality of off-line scheduling in systems exhibiting

deadlocks. A further study of these effects on the performance of these types of manufacturing systems in real-time will be reported in Chapter 6.

CHAPTER 6

DEADLOCK-FREE RESCHEDULING IN FMSs

Production scheduling and control of discrete manufacturing systems have been extensively researched over the past years and they continue to attract the interest of both academic researchers and practitioners. The generation of production schedules is becoming a necessity in today's complex manufacturing environment. Once an initial schedule has been implemented, it is almost immediately subjected to new production conditions, demands and constrains. Uncertainty (disturbance) in the production environment results in deviations from the generated schedules. This makes rescheduling essential. In this research, a rescheduling algorithm is developed to deal with machine breakdowns as a source of disturbance.

6.1 INTRODUCTION

The continuous customer demands for a variety of products, faster production rates, and higher delivery commitments are a challenge for most of modern industries. The flexibility to manufacture a wide range of products in a short time has been achieved at the expense of manufacturing efficiency. The performance of a production system greatly depends on the availability of efficient rescheduling scenario in order to deal with disturbances in real-time. These disturbances make previously generated schedules invalid.

A variety of conditions, demands and constraints necessitate revisions to an existing schedule. Some of the common disturbances that initiate rescheduling are [Shyu and Adiga, 1993]:

(a) machine breakdown;

(b) rush order arrival;

(c) shortage of materials;

(d) quality problems;

(e) over or underestimation of process time;

(f) order cancellation;

(g) due date changes; and

(h) being behind or ahead of the current schedule.

There are many approaches for real-time scheduling in manufacturing systems encountered in the literature. Among those, dynamic scheduling using simulation, dynamic scheduling using Petri nets, expert systems, Hybrid approaches, and heuristic rescheduling approach [Harmonosky and Robohn (1991), Basnet and Mize (1994), Shukla and Chen (1996), Szelke and Kerr (1994), and . Gonzalez (1995)]. Chapter 2 presented a literature review of these approaches. The point of interest of this research is rescheduling in flexible manufacturing systems.

Initial generated schedules are usually very fragile due to uncertainties that could happen in real world. Therefore, rescheduling is an assential part of any control system of a manufacturing environment. Rescheduling can be done manually by revising the original schedule. This method is not practical for complex systems especially in automated manufacturing systems. In addition, rescheduling can be done using computers

methods. The most popular computer methods are electronic Gantt chart, simulation, expert systems, and hybrid approaches.

Lin (2000) presented an approach that contains an alternative dynamic controller with a manufacturing system capability database. An initial schedule is generated whenever an unexpected perturbation is identified from the shop floor. A real-time procedure with a conflict-resolution mechanism then drives the control engine for alternative machines. Jain and El Maraghy (1997) developed local rescheduling algorithms that generate a new schedule without re-evaluating all tasks in the original schedule. Li et al. (1993) proposed a rescheduling algorithm based on the construction of a scheduling binary tree and a net change concept adopted from MRP systems. Yamamoto and Nof (1985) used a regeneration method in developing their rescheduling systems. This method involves rescheduling the entire set of operations, including those unaffected by the disturbance (uncertainty).

In this research, a rescheduling algorithm is developed to deal with machine breakdowns as a source of disturbance. The developed rescheduling algorithm guarantees a deadlock-free new schedule. The existence of alternative routes, availability of material handling facilities, and the limitations of buffer capacities are taken into consideration. The response time is an essential factor when performing rescheduling. Therefore, the developed algorithm modifies the affected portion of the original schedule rather than rescheduling the whole jobs. In addition to the response time advantage of local rescheduling, it provides stability to the original schedule.

In the literature, the alternative routing was taken into consideration with the assumption of infinite capacity of material handling and buffers. This assumption is not

realistic in most cases of flexible manufacturing systems. In addition, the limitation on both materials handling facilities and buffers sizes plays a crucial role in introducing deadlocks to manufacturing systems. Therefore, the developed rescheduling algorithm, in this research, takes into consideration the limitation on both material handling facilities and buffers sizes.

If the manufacturing system exhibits deadlocks, the rescheduling task becomes a very delicate process. In case of a disturbance, the scheduler should consider achieving the highest possible system performance in the lowest response time with guaranteeing a new deadlock-free schedule. In the following sections, the developed algorithm will be presented and tested.

6.2 DEADLOCK-FREE RESCHEDULING

The rescheduling algorithm is triggered by a disturbance occurrence in real-time. The function of the rescheduling algorithm is traditionally to update the original schedule due to this disturbance. Another function of the rescheduling algorithm is to guarantee that the new schedule is deadlock-free in case the manufacturing system exhibits deadlocks. The original schedule is defined by a Gantt chart. Every part type has a specific number of units that need to be produced. Every unit of a part type is considered as a job that may need more than one machine to be completed. An operation is a part of a job and is defined as the total time (setup and processing) required by the operation on a certain resource. The time spent on a machine, handling facility, or buffer is considered an operation. The Petri Net model of the original system must be available in advance as

an input of the rescheduling algorithm. The rescheduling algorithm is developed based on the following assumptions:

1- An initial deadlock-free schedule is available.

2- There is no pre-emption or job splitting on the alternative resource.

3- Machine breakdowns are the only source of disturbance.

4- Setup time of any operation is only resource dependent.

5- Each resource can process only one operation at any time.

6- Tools are always available at alternative machines.

The initial deadlock-free schedule can be obtained using the algorithm presented in chapter 4. No pre-emption is a valid scenario in the control of some practical cases while the pre-emption can be applied in other cases.

Upon a machine breakdown occurrence, the alternative machines that can perform the interrupted operation will be estimated. Routing the interrupted operation to one of these alternatives (if any) will be investigated. If it is decided to route the interrupted operation to one of the alternatives, the original process plan of the following operations (including material handling facilities, buffers and machines allocation) of the same job of the interrupted operation will become obsolete. Therefore, a new process plan for these following operations will be defined and the schedule will be updated.

6.2.1 RESCHEDULING ALGORITHM

It is assumed that an initial deadlock-free schedule is available in advance. The schedule is fed to the developed rescheduling algorithm as a Gantt chart in case of machine breakdown. The algorithm comprises three parts. The first part is the *Control*

Routine that receives the input data upon machine breakdown, and takes the necessary actions in order to modify the original schedule. The second part is the *Updating Routine* that finds the set of affected operations due to the disturbance, and updates the Gantt chart. The third part is the *Verification Routine* that checks whether the new schedule is deadlock-free. The following notations are used in the developed algorithm.

NOTATIONS:

- $O_{i,j}$: operation j of job i.
- $O_{k,w}$: next operation on resource r after operation $O_{i,j}$.
- $pt_{i,j}$: processing time of $O_{i,j}$.
- st_r: setup time of *DAO* on machine r: $r \in$ *Alt_route_set*.
- N_r: Number of units of reasource *r*.
- $s_{i,j}$: planned starting time of $O_{i,j}$.
- $e_{i,j}$: planned ending time of $O_{i,j}$.
- *dt*: time of disturbance.
- *edt*: the expected down time of the broken machine.
- DAO: directly affected operation. The operation that is interrupted due to the machine breakdown or scheduled on the broken machine during its downtime;
- *Alt_route_set*: the set of machines which can serve *DAO*.
- ewt_r: expected waiting time of *DAO* to start processing on machine *r*: $r \in$ *Alt_route_set*.
- ept_r: expected processing time of *DAO* on machine *r*: $r \in$ *Alt_route_set*.
- ast_r: additional setup time of *DAO* on machine *r*: $r \in$ *Alt_route_set*.
- est_r: expected starting time of *DAO* on *r* ($est_r = dt + ewt_r$).

- ect_r: The expected completion time of *DAO* by machine *r* (ect_r=est_r+ept_r+ st_r÷ast_r).
- *Affected_operations*: Set of affected operations due to the disturbance.
- $OT_{i,j}$: Type of operation $O_{i,j}$: $O_{i,j} \in$ *Affected_operations*. $OT_{i,j}$ has the mapping: 0→*DAO*. 1→following operation of the same job of *DAO*. and 2→any other operation.
- $RT_{i,j}$: ending time of previous operation on the same resource of operation $O_{i,j}$: $O_{i,j} \in$ *Affected_operations*.
- $JT_{i,j}$: ending time of previous operation of the same job of operation $O_{i,j}$: $O_{i,j} \in$ *Affected_operations*.
- *RD*: binary variable represents the routing decision.
- *FT*: binary variable represents the result of the deadlock existence test with the mapping: 0→the schedule has a deadlock. and 1→the schedule is deadlock-free;
- *M_Affected_operations*: Set of affected operations due to the disturbance that aremodified.

Control Routine:

1- Find the operation ($O_{i,j}$) that is interrupted due to the machine breakdown. Let $DAO=O_{i,j}$. Add *DAO* to *Affected_operations* and *M_Affected_operations*.

2- Evaluate the *Alt_route_set* of *DAO*.

3- If (*Alt_route_set* = ϕ) go to step (9).

4- Find the machine r: r $\in$ *Alt_route_set*, with the minimum *ect*.

Alt_route_set=Alt_route_set-r.

5- If ($ect_r \geq e_{DAO}$+*edt*) go to step (3)

6- Let RD =1 and use the *Updating routine*.

7- Use the *Verification routine*. If (the new schedule is NOT deadlock-free) then restore the data and go to step (3).

8- If(there is other scheduled operations on the broken machine during its downtime), then let DAO = the operation with the earliest starting time, and go to step (2). otherwise stop.

9- Let RD = 0 and use the *Updating routine*.

Updating Routine

1. Begin routine.
2. While (*Affected_operations* $\neq \phi$)
   2.1. Select an operation ($O_{i,j}$) from *Affected_operations* that has the earliest starting time ;
   2.2. If (RD = 1 AND $OT_{i,j} = 0$):
      2.2.1. Let r be the resource of $O_{i,j}$.
      2.2.2. $s_{i,j} = est_r$;
      2.2.3. $e_{i,j} = ect_r$;
   2.3. If(RD = 0 AND $OT_{i,j} = 0$) ;
      2.3.1. $e_{i,j} = e_{i,j} + edt$;
   2.4. If (RD =1 AND $OT_{i,j}$ =1) ;
      2.4.1. Evaluate the *Alt_route_set* of $O_{i,j}$;
      2.4.2. Find the resource r: r $\in$ *Alt_route_set*, with the minimum *ect* ;
      2.4.3. Let r be the resource of $O_{i,j}$;

2.4.4. $s_{i,j} = est_r$;

2.4.5. $e_{i,j} = ect_r$;

2.5. If $(OT_{i,j} = 2)$:

2.5.1. If$(RT_{i,j} > JT_{i,j})$

- $s_{i,j} = RT_{i,j}$;

2.5.2. Else

- $s_{i,j} = JT_{i,j}$;

2.5.3. $e_{i,j} = s_{i,j} + pt_{i,j} + st_{i,j}$;

2.6. If ($O_{i,j}$ is NOT the last operation of its job AND $s_{I,j} > s_{i,j+1}$) ;

2.6.1. If ($O_{i,j+1} \notin$ *Affected_operations*) ;

- Add $O_{i,j+1}$ to *Affected_operations* ; *M_Affected_operations*

2.6.2. If ($O_{i,j+1} \notin$ *Affected_operations*) ;

- Add $O_{i,j+1}$ to *M_Affected_operations* ;

2.6.3. $OT_{i,j+1} = 2$;

2.6.4. $JT_{i,j+1} = e_{I,j}$;

2.7. If ($O_{i,j}$ is NOT the last operation of the resource r AND there is an operation $O_{k,w}$ scheduled on r with $e_{I,j} > s_{k,w} > s_{I,j}$)

2.7.1. If ($O_{k,w} \notin$ *Affected_operations*) ;

- Add $O_{k,w}$ to *Affected_operations* ;

2.7.2. If ($O_{i,j+1} \notin$ *Affected_operations*) ;

- Add $O_{i,j+1}$ to *M_Affected_operations* ;

2.7.3. $OT_{k,w} = 2$;

2.7.4. $RT_{k,w} = e_{I,j}$;

2.8. If ($OT_{i,j}$≠0 AND $O_{i,j}$ is NOT the first operation of its job AND $s_{i,j} > e_{i,j-1}$) :

2.8.1. $e_{i,j-1} = s_{i,j}$:

2.8.2. If ($O_{i,j-1}$ is NOT the last operation of the resource r AND there is an operation $O_{l,m}$ scheduled on r with $e_{i,j-1} > s_{l,m} > s_{i,j-1}$)

- If ($O_{l,m} \notin$ *Affected_operations*) ;
  - Add $O_{l,m}$ to *Affected_operations* ;
- If ($O_{i,j+1} \notin$ *Affected_operations*) ;
  - Add $O_{i,j+1}$ to *M_Affected_operations* ;
- $OT_{l,m} = 2$;
- $RT_{l,m} = e_{i,j-1}$;

2.9. *Affected_operations* = *Affected_operations* - $O_{i,j}$;

2.10. End while ;

3. End routine.

Verification Routine:

1. Begin routine;
2. Let FT=1 ;
3. While (*M_Affected_operations* ≠ ϕ)
4. Select an operation $O_{i,j}$ from *M_Affected_operations* and remove it from *M_Affected_operations* ;
5. Let $T = s_{i,j}$;
6. Let the marking of each resource place r be N_r and every operation place is zero ;

7. Find the set of operations that are currently served or starting at T. Let the marking of their places equal 1 and of their resources equal N_r-1 ;
8. Find the set of operations that are ending at T. Let the marking of their places equal 0 and of their resources r equal N_r+1;
9. If (there is an empty siphon at T), let FT=0 and stop;
10. If ($T \neq e_{i,j}$), let $T = e_{i,j}$ and go to step (6);
11. End while ;
12. End routine.

The *Control* routine is the main part of the algorithm. It is initiated by the machine breakdowns. It has an access to the current schedule data (as Gantt chart), and the PN model of the system. The interrupted operation (*DAO*), due to machine breakdown, is found and the set of alternative machines (*Alt_route_set*) that can perform it is evaluated. If the *Alt_route_set* is empty, the interrupted operation will be kept on the broken machine and the schedule will be modified based on the expected machine downtime (*edt*). In case of nonempty set of *Alt_route_set*, the expected completion time (ect_r) of *DAO* on every member of *Alt_route_set* is estimated. The value of ect_r can be estimated using the following formula:

$$ect_r = est_r + ept_r + st_r + ast_r \quad (1)$$

Where:

$$est_r = dt + ewt_r \quad (2)$$

If *DAO* is routed to an alternative machine, it will compete with other operations that are originally scheduled on the same machine. Therefore, a dispatching rule can be

used to find the order of processing *DAO* on the alternative machine and estimate the expected waiting time (ewt_r) to start processing. Another point that should be considered is the expected processing time (ept_r) of *DAO* on the alternative machine. There are two possibilities. First, *DAO* resumes processing from the breakdown point of time and which may need an additional setup time (ast_r). In other words, the required processing time on the alternative machine will be considered as the remaining processing time of *DAO* after the machine breakdown. The value of ast_r could be a percentage of st_r. Second, *DAO* acquires the total planned processing time on the alternative machine.

After estimating the value of ect_r of each alternative machine, the machine r ($r \in$ *Alt_route_set*) with minimum *ect* is selected to test routing *DAO* to it. If the difference between *dt* and ect_r is less than *mdt*, the *Updating* routine is called to do further testing on routing *DAO* to *r*. The *Updating* routine routes *DAO* to *r* and modifies the required part of the schedule. To ensure that the modified part of the schedule is deadlock-free, the *Verification* routine is called. If the new part of the schedule is not deadlock-free, another alternative machine is selected to test routing *DAO* to it. The same scenario is repeated until *DAO* is routed to another machine or kept on the broken machine, and the schedule is updated based on the value of *mdt*. The steps of both *Updating* and *Verification* routines will be declared in the following paragraphs.

The function of the *Updating* routine is to find the operations affected by disturbance and modify the necessary portion of the schedule. The *Updating* routine receives the decision regarding routing the *DAO* to an alternative machine (*RD*=1) or keeping it on the broken machine (*RD=0*) from the *Control* routine. If routing *DAO* to an alternative machine is investigated, the *DAO* will be assigned to the alternative machine

with defining the starting and ending times (est_r, and ect_r, respectively). This is performed in step (2.2) of the algorithm.

The set of affected operations is divided into three types: (1) the *DAO* operation ($OT_{i,j}$=0), (2) a following operation of the same job of *DAO* ($OT_{i,j}$=1), and (3) any other operation ($OT_{i,j}$=2). If the *DAO* is routed to an alternative machine, the remaining part of its job original route becomes obsolete. Therefore, a new route for the second type operations ($OT_{i,j}$=1) will be found and they will be inserted within the schedule of the new route's resources. These actions are performed in step (2.4) of the algorithm.

The third type of affected operations ($OT_{i,j}$=2) could be a result of: (1) a change in the ending time of the same job previous operation, (2) a change in the ending time of the same resource previous operation, or (3) a change in the ending time of both of the same job and same resource's previous operation. Therefore, every affected operation has two other attributes that have values of the new ending times of both the same job and same resource previous operations ($JT_{i,j}$, and $RT_{i,j}$ respectively). The new starting time of the third type of affected operation is the maximum of { $JT_{i,j}$, and $RT_{i,j}$}.

The *Updating* routine modifies the necessary portion of the schedule by finding the affected operations by disturbance. Figure 6.1 shows how a change in an operation might affect other operations. Any currently modified operation ($O_{i,j}$) might affect three other operations directly and one operation indirectly as follows:

(1) The next operation ($O_{k,w}$) on the same resource of $O_{i,j}$.

(2) The next operation ($O_{i,j+1}$) of the same job of $O_{i,j}$.

(3) If the starting time of $O_{i,j}$ has increased to surpass the ending time of the previous operation of the same job ($O_{i,j-1}$), the ending time of $O_{i,j-1}$ has to be changed to be equal to the new starting time of $O_{i,j}$.

(4) If the third case occurs, this might affect the next operation ($O_{l,m}$) on the same resource of operation $O_{i,j-1}$.

The third type of affected operations ($O_{i,j-1}$) has to be considered in the case of having limited resources. Because the previous resource of a job can not release it unless the next required resource of the same job becomes available. Ignoring this fact will result in deadlock schedule. It is necessary to add both $O_{i,j-1}$ and $O_{l,m}$ to the affected operations but it is not necessary to consider the effect of $O_{i,j-1}$ on other operations.

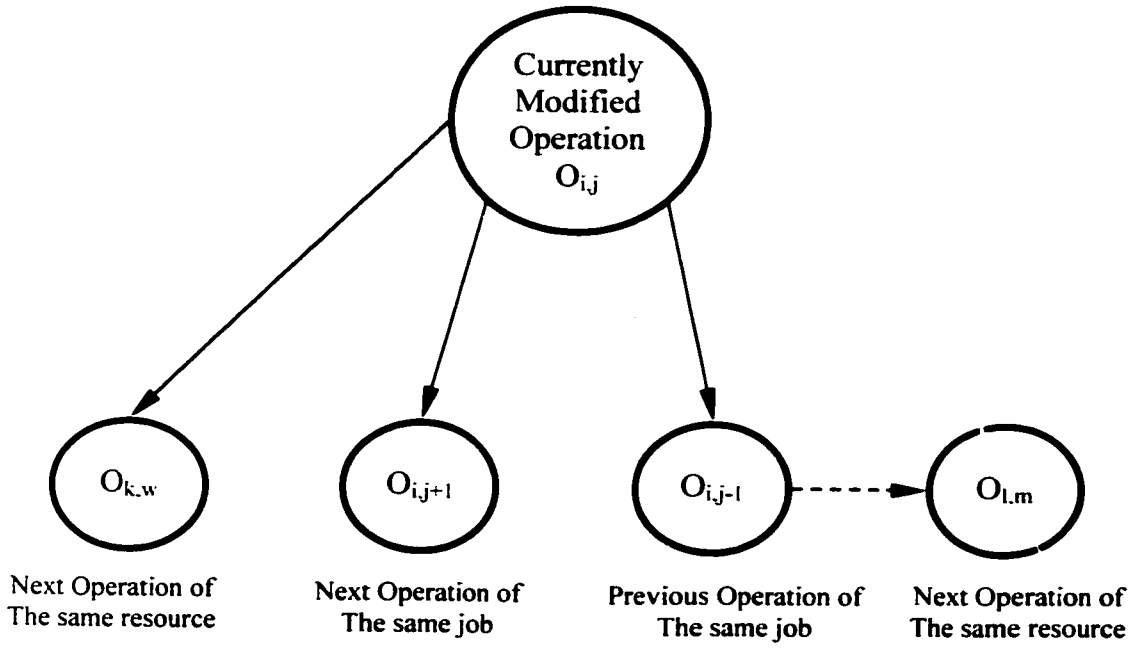


Figure 6.1: Finding the affected operations due to the disturbance

The *Verification* routine receives the set of all affected operations after modifying the schedule (*M_Affected_operations*) due to the disturbance. The marking of the Petri Net model is estimated at the start and end times of every operation ($O_{i,j}$) of *M_Affected_operations*. If any minimal siphon is empty at $T=s_{i,j}$ or $T= e_{i,j}$, the routine will stop with the message: "the new schedule has a deadlock (FT=0)". Otherwise, the routine will continue until all operations of *M_Affected_operations* are checked.

6.2.2 ILLUSTRATIVE EXAMPLE

Consider a manufacturing system composed of six machines (M1, M2, M3, M4, M5 and M6), two robots (R1, and R2), two intermediate buffers (B1, and B2), and two loading and unloading stations. The parts come from the loading stations directly to the required first machine. The machines can unload themselves to the unloading stations. Buffer B1 is an intermediate buffer for machines M1, M2 and M3 with a capacity of one. Buffer B2 is an intermediate buffer for machines M4, M5 and M6 with a capacity of one. Robots are used to unload the parts from the machines to the intermediate buffers. The system is producing two part types, Part1 and Part2. The production sequences of Part1 and Part2 and the processing and setup times are shown in Table 6.1. Figure 6.2 shows a schematic representation of the flexible manufacturing system.

The timed Petri Net models for Part1 and Part2 are shown in Figure 6.3 (a) and (b). The processing time and setup time of every operation are shown beside each corresponding operation place. The shown Petri Net models do not include the resources places for the sake of simplifying the graphical representation. In order to make the Petri Net more readable, the resource name is written beside the operation place that requires

it. For example, operation place P1 represents processing the first operation of Part1 on M1 with setup time of 10 and processing time of 100.

The minimal siphons of the above system were estimated and are presented in Table 6.2. The physical representation of each siphon is demonstrated in Figure 6.4. For example, if the Petri net model encounters the third siphon empty in one of its states, then the corresponding physical system's state will be case (c) in Figure 6.4.

Considering five units for each part type, the heuristic algorithm presented in chapter 4 is used to optimize the average flow time. An average flow time of 279.0 is obtained. The Gantt chart of the deadlock-free schedule is shown in Figure 6.5. Each bar shown in the chart represents the actual time of the resource occupied by the operation. The rescheduling algorithm was applied to the generated schedule in order to deal with the three disturbances presented in Table 6.3. It is assumed that: (1) Part1 has higher priority than Part2, and (2) in case of routing the interrupted operation to an alternative machine, the operation will resume processing with its original remaining processing time and with an additional setup time (*ast*) equal to 10% of the required setup time on the alternative machine.

The first disturbance represents a breakdown of machine M5 at Time=65 and expected downtime equal to 50. The interrupted is $O_{1,1,0}$. The alternative machines that can perform this operation are M4 and M6. The expected start and completion times of $O_{1,1,0}$ on M4 and M6 are (65,111) and (340,386) respectively. These are estimated using equations (1) and (2). By comparing the minimum value of the expected completion time (111) with the expected completion time if the part kept on the broken machine (150), it is decided to test routing operation $O_{1,1,0}$ to M4. The necessary steps of modifying the

schedule are performed as mentioned in the rescheduling algorithm. The new part of the schedule is tested for feasibility and deadlock-freeness. It is found that the new schedule is deadlock-free.

The algorithm detected operation $O_{1.3.0}$ as a directly affected operation due to the first disturbance. Therefore, the expected completion time in both cases of routing $O_{1.3.0}$ to an alternative machine (M4 and M6) and keeping it on the broken machine is compared. It is decided to keep $O_{1.1.0}$ on the broken machine. The necessary steps of modifying the schedule are performed. The new schedule is shown in Figure 6.6.

Table 6.1: Production sequences and processing and setup times.

Part	Alternative	Operation Number			
		1	2	3	4
		(r,*st*,*pt*)	(r,*st*,*pt*)	(r,*st*,*pt*)	(r,*st*,*pt*)
Part1	1	(M1,10,100) (M2,10,80)	(R1,0,10)	B2	(M4,10,80) (M5,10,120) (M6,10,100)
	2	(M3,10,50)	(R2,0,10)		
Part2	1	(M4,10,110) (M5,10,80)	(R1,0,10)	B1	(M1,10,70) (M2,10,120) (M3,10,100)
	2	(M6,10,50)	(R2,0,10)		

Table 6.2: The minimal siphons of the illustrative example.

Siphon Number	Set of places
1	R1,B1,M1,M2,M3,B2,M4,M5,M6,R2,P18,19,P20,P7,P8,P9
2	R1,B1,M1,M2,M3,R2,P18,P19,P20,P5,P4
3	R1,B2 M4,M5,M6,R2,P15,P16,P7,P8,P9

Table 6.3: Scheduled disturbances applied to the generated schedule.

Parameter	Disturbance number		
	1	2	3
Time of disturbance (*dt*)	65	100	170
Broken machine	M5	M2	M4
Expected downtime (*edt*)	50	120	30
Additional setup time (ast_r)	0.1x st_r	0.1x st_r	0.1x st_r

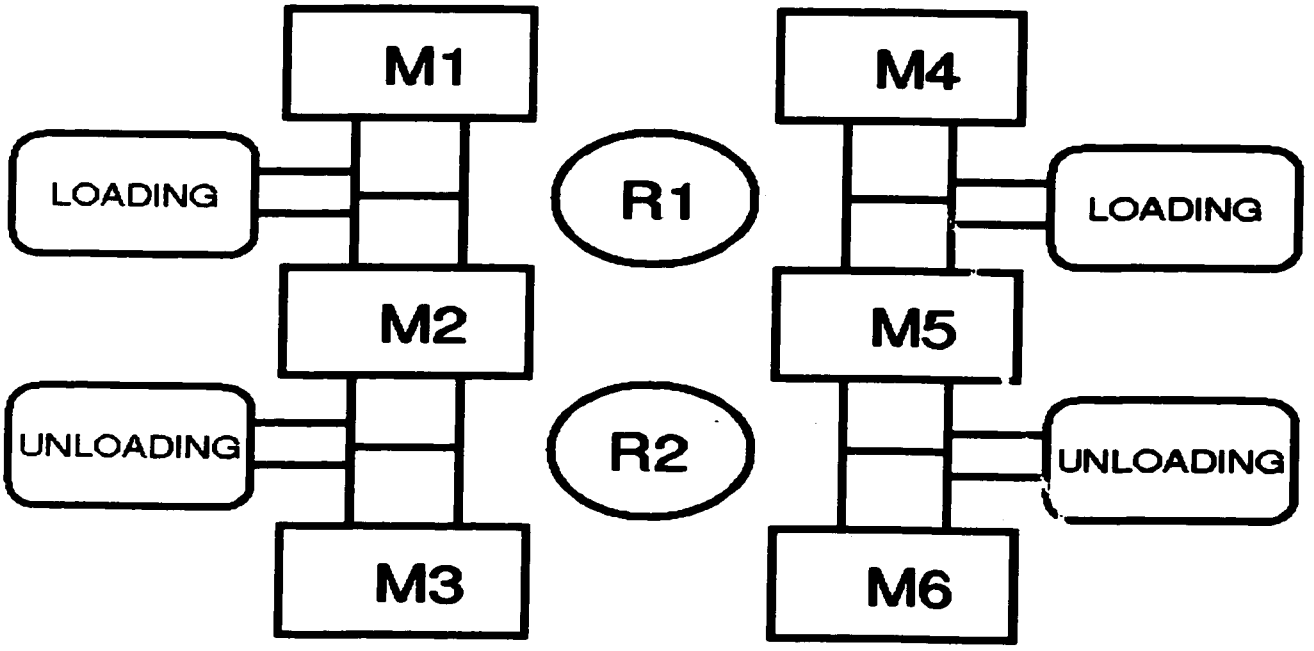


Figure 6.2: Flexible manufacturing system configuration

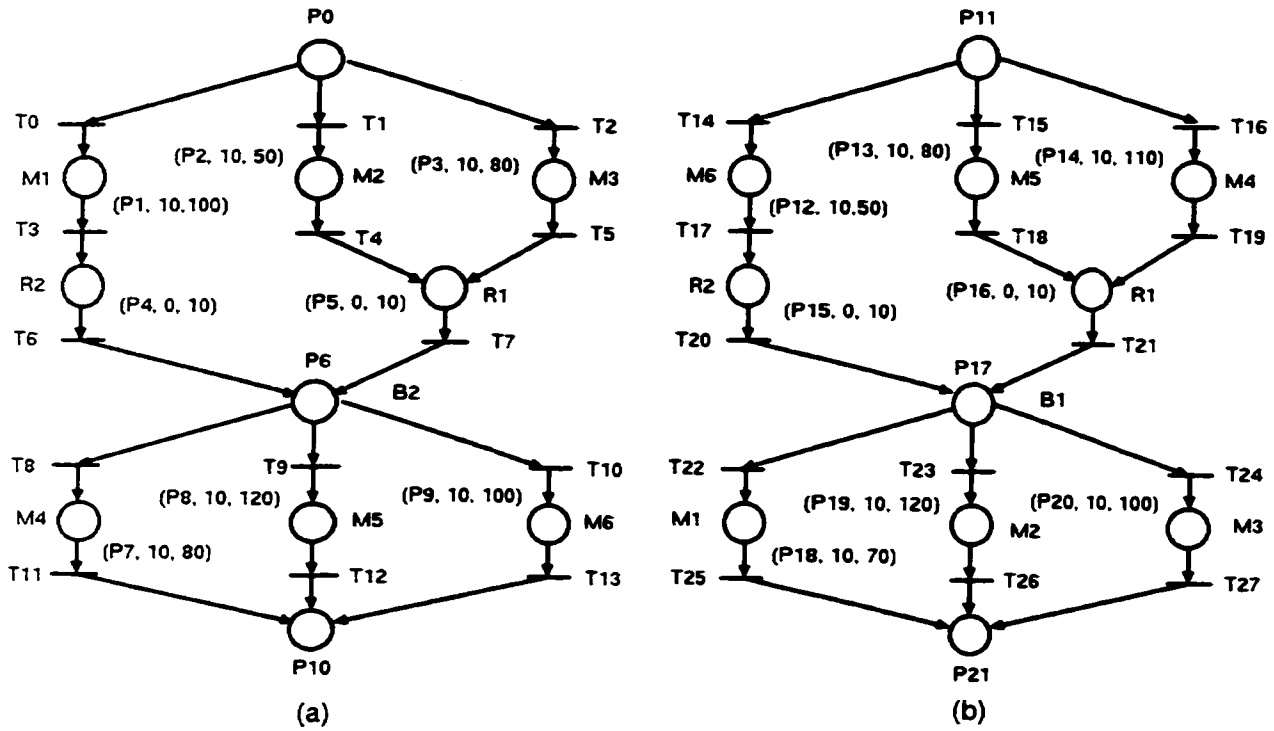


Figure 6.3: Petri Net model of the FMS

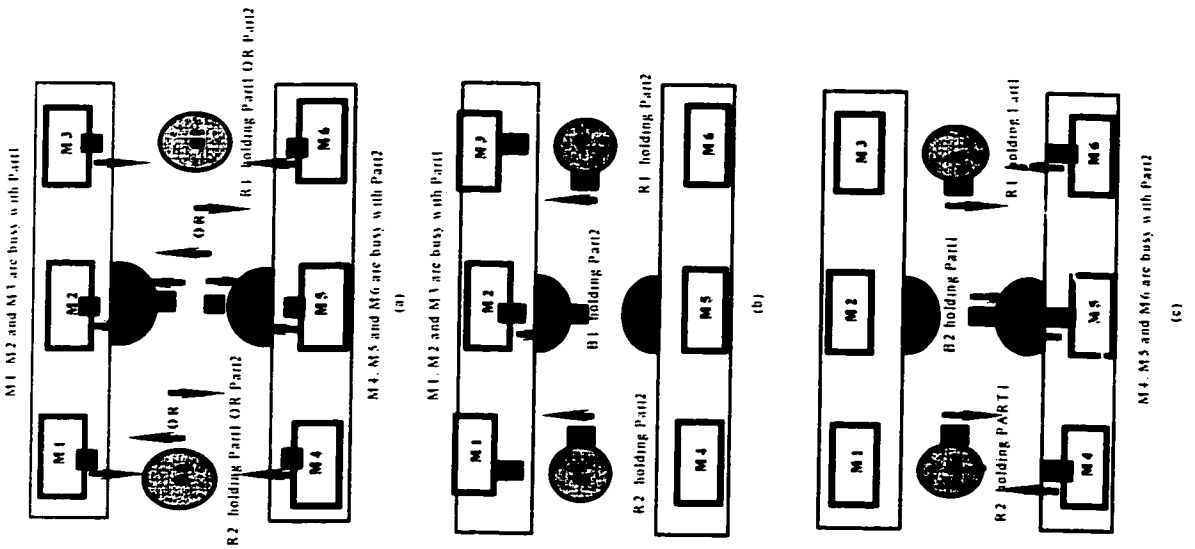


Figure 6.4: Deadlock states of the system of the illustrative example

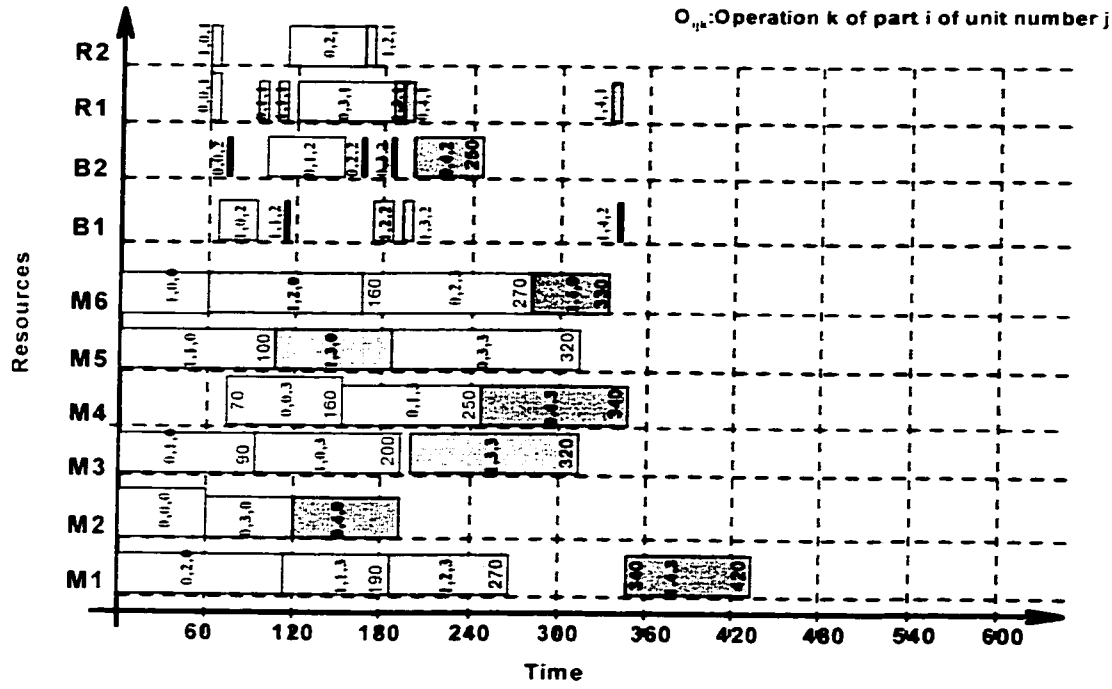


Figure 6.5: The generated deadlock-free schedule

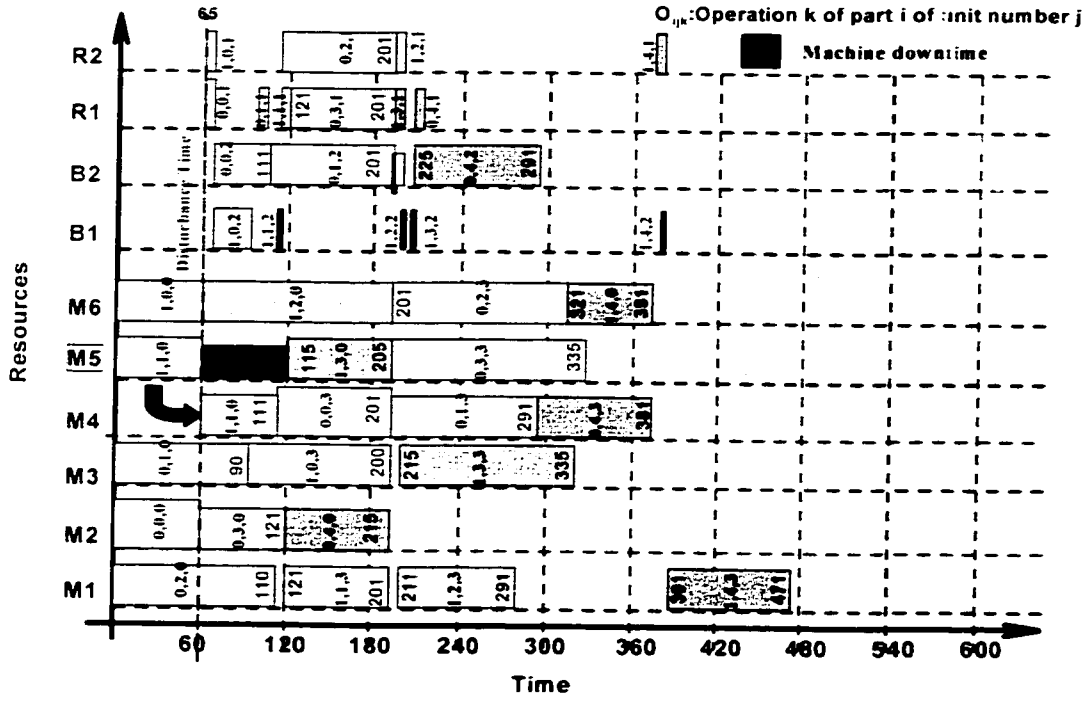


Figure 6.6: Disturbance #1: The modified schedule

The second disturbance represents a breakdown of machine M2 at Time=100 and expected downtime equal to 120. The interrupted operation is $O_{0,3,0}$. The alternative machines that can perform this operation are M1 and M3. The expected starting and completion times of $O_{0,3,0}$ on M1 and M3 are (110,141) and (200,231), respectively. The expected completion time of $O_{0,3,0}$ on the broken machine is 240. Therefore, it is decided to test routing it to M1. The new schedule was checked by the *Verification* routine. It was found that the new schedule is deadlock-free. Another directly affected operation is found ($O_{0,4,0}$). The expected starting time and completion of $O_{0,4,0}$ on M1 and M3 are (160,281) and (200,301) respectively. After comparing the minimum completion time of $O_{0,4,0}$ on M1 (281) with the expected completion time on the broken machine (280), the operation is kept on the broken machine (M2). The modified schedule is shown in Figure 6.7.

In the third disturbance, M4 is broken down at T=170 with an expected downtime of 30. From the schedule presented in Figure 6.5, it can be noted that the only directly affected operation is $O_{0,1,3}$. The alternative machines that can perform this operation are M5 and M6 with expected starting and completion times (320,411) and (270,361). The expected completion time of $O_{0,1,3}$ is 280, if it is kept on the broken machine M4. Therefore, the operation $O_{0,1,3}$ is kept on the broken machine. The modified schedule after this disturbance is presented in Figure 8.6.

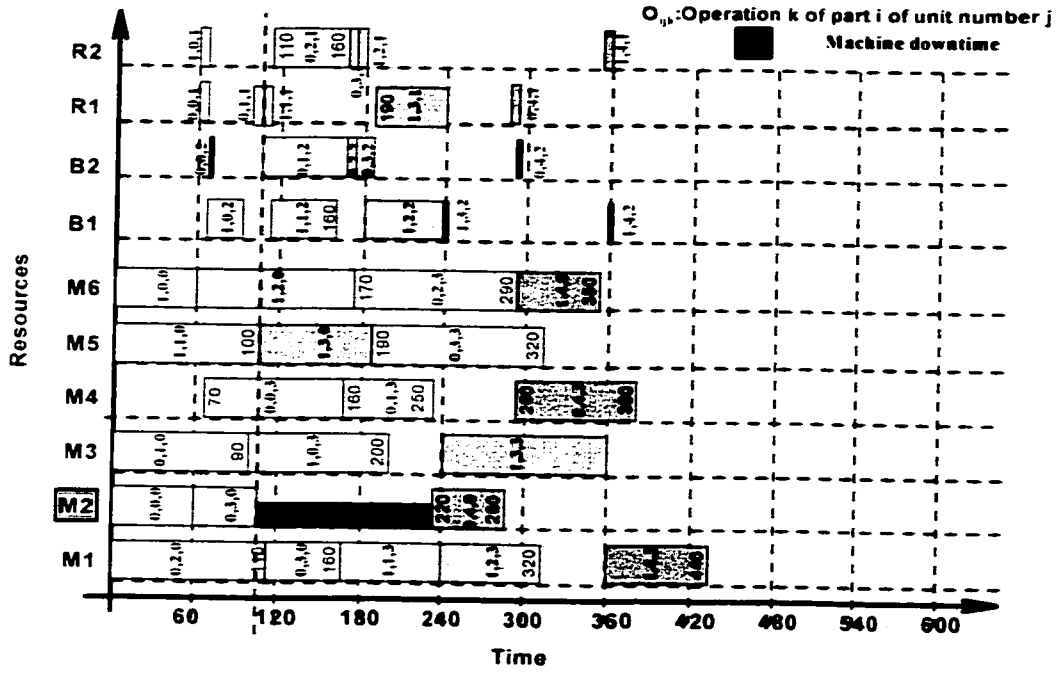


Figure 6.7: Disturbance #2: The modified schedule

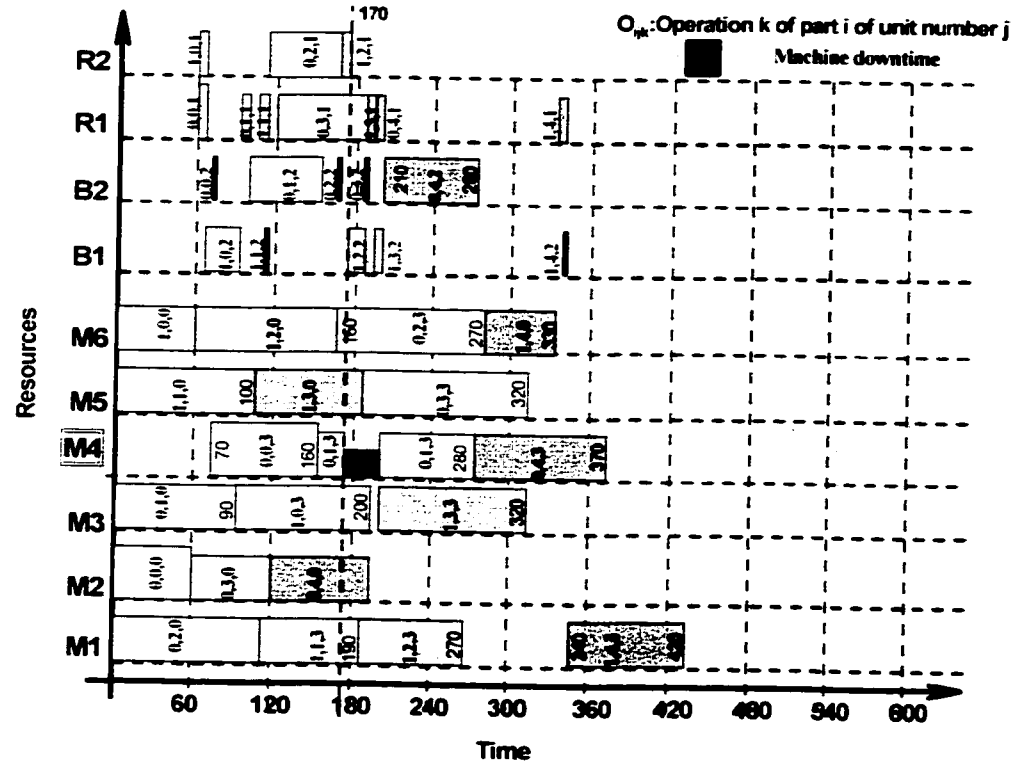


Figure 6.8: Disturbance #3: The modified schedule

6.3 RESULTS AND DISCUSSION

From the above results, it can be noted that there are some control factors that affect the system performance. Intuitively, it can be claimed that routing flexibility improves the manufacturing system performance in real-time especially with high frequency of machine breakdowns and high machine downtime. In the above example, routing an operation to an alternative machine was based on the expected completion time (ECT). Other criteria for routing the interrupted operations could be considered. In this research, routing (or keep) the operation to (or on) the lowest utilized machine (LUM) will be considered.

In case of routing an operation to an alternative machine, the operation will compete with other operations that were waiting for service on the same machine at the time of disturbance. Therefore, a dispatching rule must be used to find the order of serving the routed operation among those operations. Two dispatching rules are considered: Shortest processing time (SPT) rule and first come first served (FCFS) rule. The factors that will be considered to study their effect on the system performance (measured by the average flow time, makespan, and average machines utilization) are presented in Table 6.4.

The first factor, Routing Flexibility Utilization (RFU), represents the utilization of the manufacturing systems routing flexibility in real-time. The rescheduling algorithm will be used to test two strategies (two levels of RFU factor). First, when a breakdown occurs, the interrupted operation (and all the scheduled operations on the broken machine during its downtime) will wait until the machine is brought back to service and continues processing on the same machine. Second, when a breakdown occurs, routing the interrupted (or directly affected) operation to an alternative machine will be investigated.

The second factor, Probability of Machine Breakdowns (PMB), represents the frequency of machine breakdowns. The number of machine breakdowns (N) is estimated by multiplying the original schedule makespan by the probability of breakdowns. These breakdowns are generated randomly using a uniform distribution of interval (0, makespan). Then for every breakdown, a machine is selected randomly, from among all machines, with an equal opportunity (probability). The duration of machine downtime after a breakdown is represented by the Machine Downtime (*MDT*) factor. It is assumed that the machine will start its repair right after the breakdown.

There are two sets of experiments that will be conducted. The first set includes the HIGH level of the routing flexibility utilization (RFU) factor and all levels of the other factor. The second set includes the LOW level of the routing flexibility utilization (RFU) factor and both levels of the machine downtime (*MDT*) factor. The total number of experiments is 40 (32 for the first set and 8 for the second set).

The system presented in the illustrative example is modified by considering producing three part types: Part1, Part2, and Part3 and increasing the number of operations per part type. The production sequences of the three parts and the setup and processing times are shown in Table 6.5 and 6.6, respectively. Considering 20 units for each part type (a total of 60 units), the heuristic algorithm presented in chapter 4 is used to optimize the average flow time and obtain a deadlock-free schedule.

The results of the first and second set of experiments are presented in Figures 6.9-12 and Figure 6.13, respectively. The maximum CPU time in all experiments was 0.33 sec. It can be noted from Figures 6.9 and 6.13 that the deterioration in system performance is not high (average of 3.8%) at low machine downtime. On the other hand,

this effect becomes more significant (average of 12.3%) at high machine downtime as the probability of breakdowns increases. At low machine downtime and low frequency of machine breakdowns, the effect of machine downtime could be absorbed by the machine idle time within the originally generated schedule. On the other hand, at high machine downtime and high frequency of machine breakdowns, this effect is not totally absorbed by the machine idle time which in turn increases the makespan and delays the completion time of some jobs. Therefore, this effect reaches its highest value (average of 16.8%) at high machine downtime and high frequency of machine breakdowns.

Table 6.4: The considered factors to study their effect on the system performance.

1. Routing Flexibility Utilization (RFU):
   1.1. LOW: Do not consider routing flexibility
   1.2. HIGH: Consider routing flexibility
2. Probability of Machine Breakdowns (PMB):
   2.1. PMB =0.1%
   2.2. PMB =0.2%
   2.3. PMB =0.3%
   2.4. PMB =0.4%
3. Machine Downtime (*MDT*):
   3.1. Low: *edt* = Uniform(0.25 * apt, 0.75 * apt)
   3.2. High: *edt* = Uniform(1.0 * apt, 2.0 * apt)
4. Routing Criterion (RC):
   4.1. Earliest expected completion time (ECT)
   4.2. Lowest utilized machine (LUM)
5. Dispatching Rule (DR):
   5.1. FCFS
   5.2. SPT

Table 6.5: Production sequences of the three part types.

Part type	Operation						
	1	2	3	4	5	6	7
Part1	M1 M2 M3	R1	B2	M4 M6	R1	B1	M3
Part2	M4 M5	R2	B1	M1 M2	R1	B2	M4 M5
Part3	M6	R2	B1	M1 M3	R2	B2	M6

Table 6.6: Setup and processing times of the three part types.

Part	Operation						
	1	2	3	4	5	6	7
Part1	(10,100) (10,80) (10,50)	(0,10)	B2	(10,80) (10,100)	(0,10)	B1	(10,100)
Part2	(10,100) (10,70)	(0,10)	B1	(10,60) (10,100)	(0,10)	(0,10)	(10,60) (10,100)
Part3	(10,100)	(0,10)	B1	(10,50) (10,50)	(0,10)	(0,10)	(10,80)

The effect of the routing criteria factor is shown in Figure 6.10. Routing the interrupted (or directly affected) operation to the machine that finishes it at the earliest time (ECT) always gives the best performance. The difference in system improvement of ECT over LUM reaches its maximum value (2.13%-4.45%) at the highest machine downtime . The lowest utilized machine (LUM) criterion objective is to balance the load on the different machines. This might maximize the minimum machine utilization but it is not necessary to improve other criteria such as, average flow time, makespan and average machine utilization.

The DR factor did not significantly affect the system performance. The SPT rule always gives the best makespan and average machine utilization compared to the FCFS

rule. as shown in Figure 6.11. The interaction between the MDT factor and the RC factor is illustrated in Figure 6.12. It is interesting to note that the ECT always gives the best performance especially at high machine downtime. Figure 6.14 exhibits the comparison of results when routing flexibility is considered (HIGH) and not considered (LOW). Indeed. utilizing the routing flexibility in real-time always improves the system performance.

As mentioned above. the maximum CPU time in all experiments was 0.33 sec. In order to test the limitation of the rescheduling algorithm. 60 units for each part type (a total of 180 units), were considered. The heuristic algorithm presented in chapter 4 is used to optimize the average flow time and obtain a deadlock-free schedule. The generated schedule contained 1260 tasks (180 units multiplied by 7 operations per unit). When the rescheduling algorithm dealt with only one breakdown the CPU time was 0.17 seconds. In addition. the algorithm modified the schedule with 16 breakdowns in the same time in 0.82 seconds. It is obvious. from the results. that the rescheduling algorithm is capable of dealing with larger problem sizes but these problems can not be solved by the developed scheduling algorithm presented in chapter 4. In other words. the limitation is not due to he developed rescheduling algorithm but due to the capability of the scheduling algorithm.

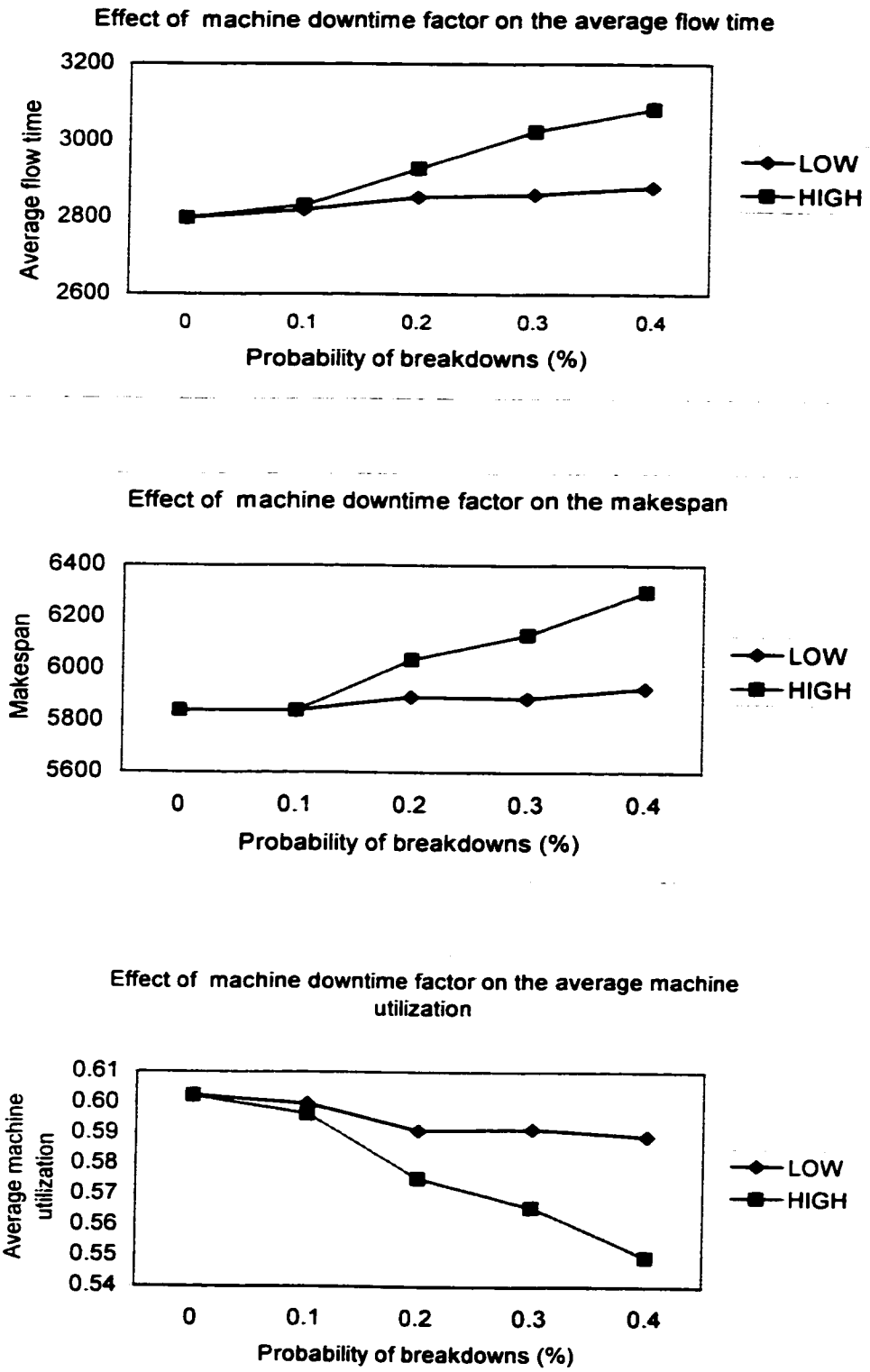
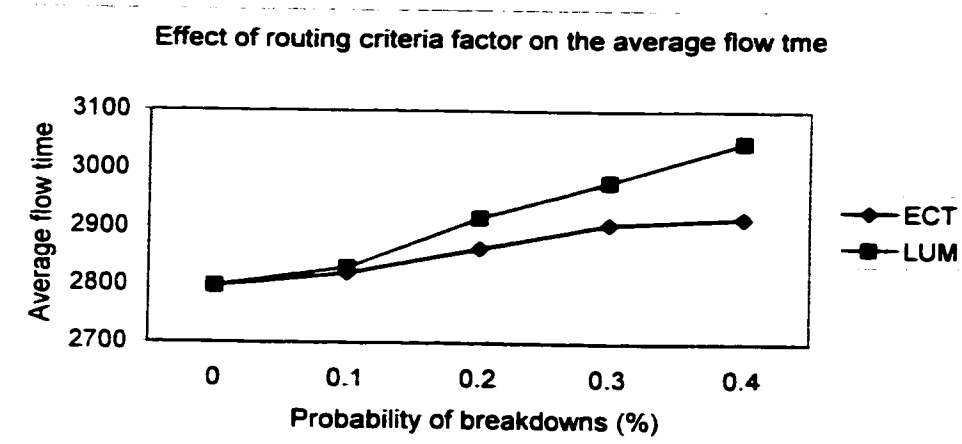
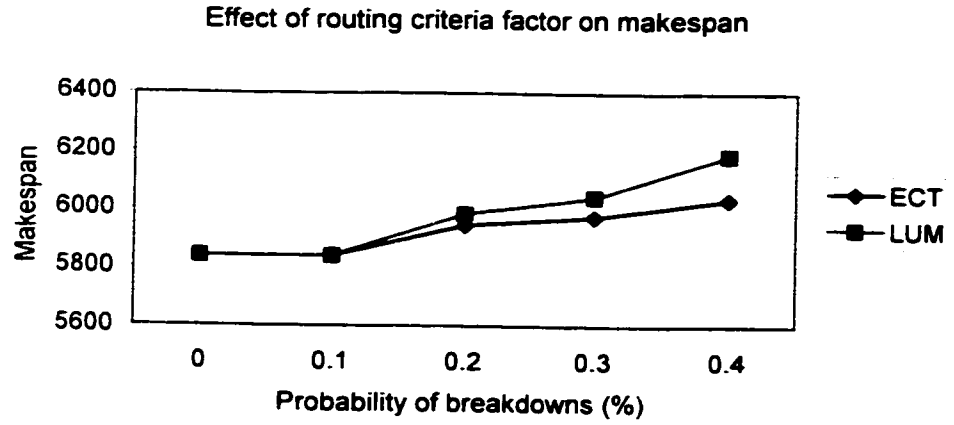


Figure 6.9: First set: Effect of machine downtime factor





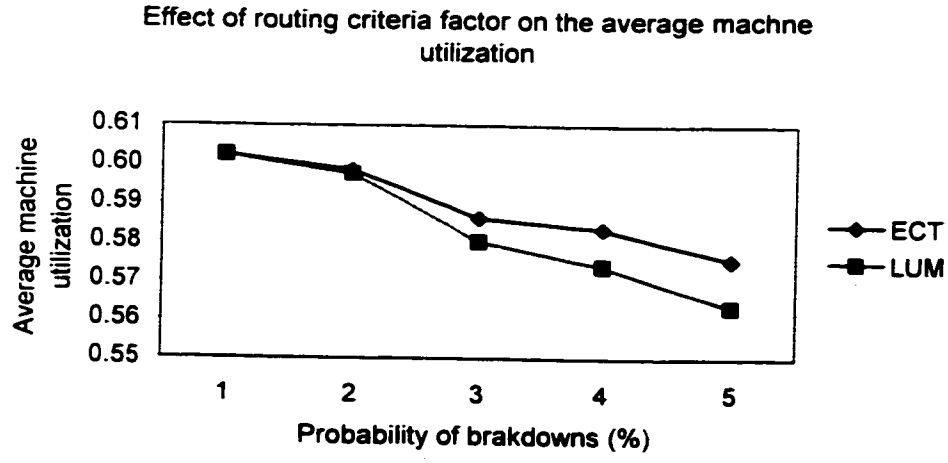


Figure 6.10: First set: Effect of routing criteria

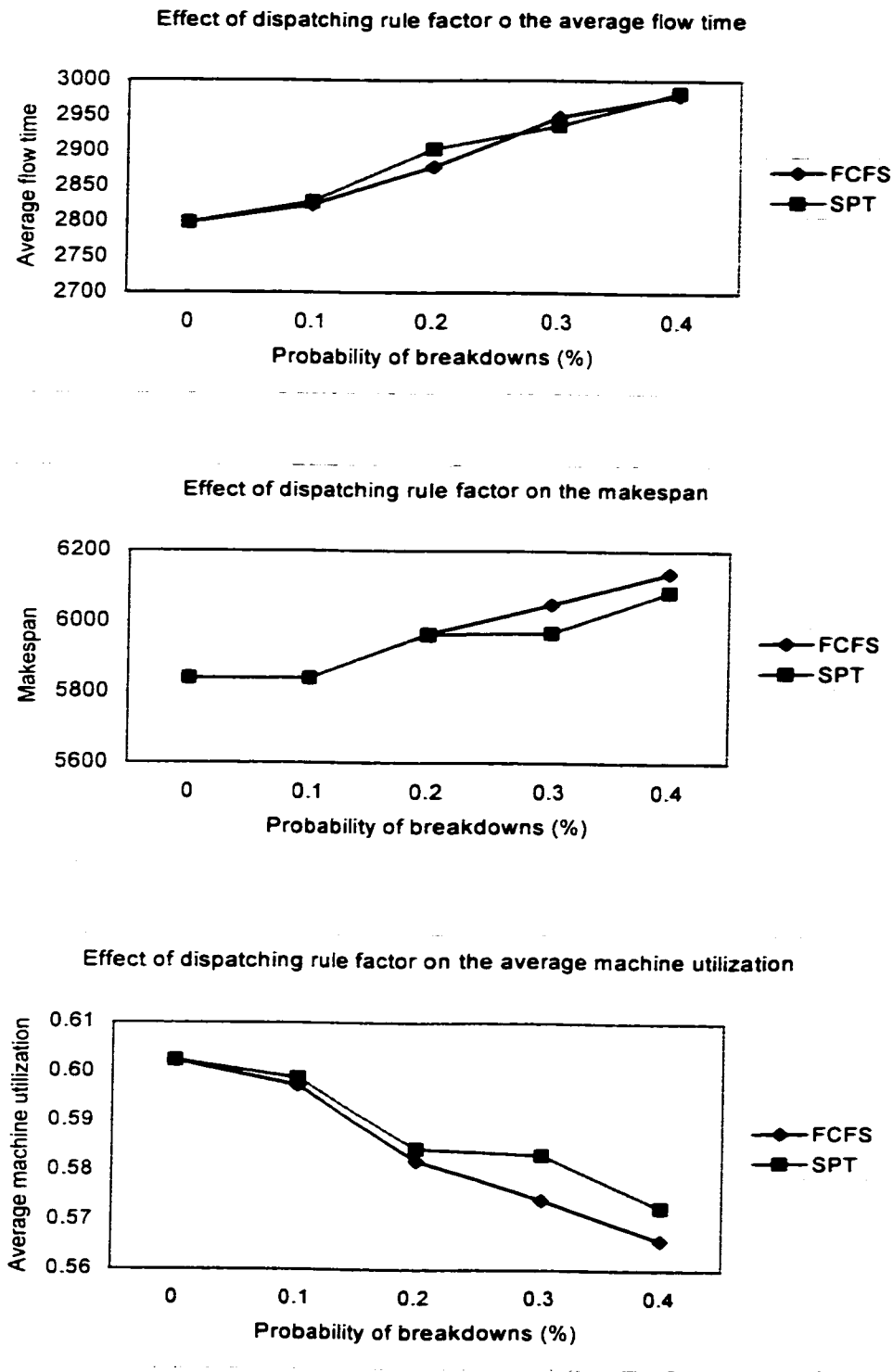
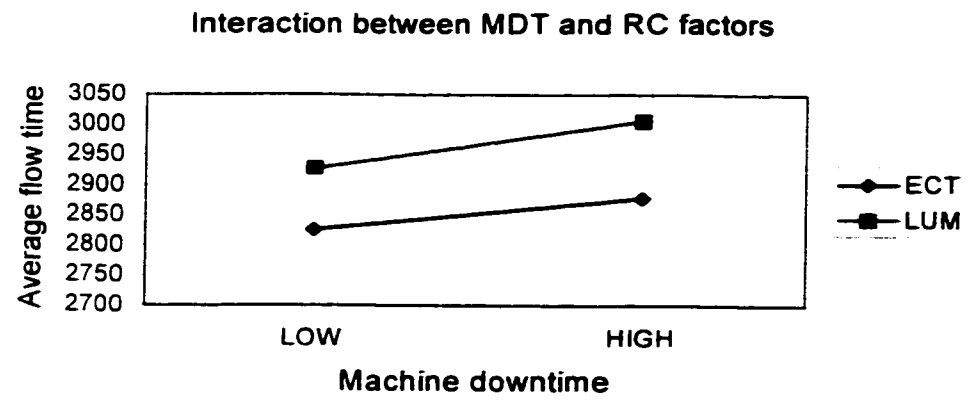
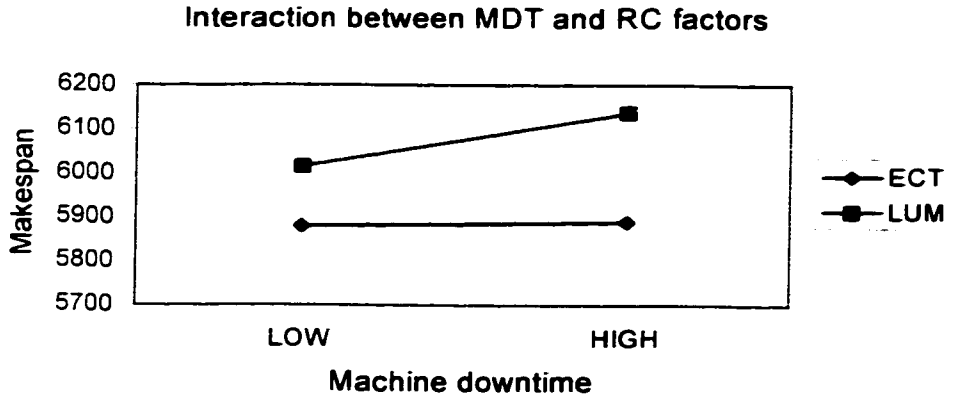


Figure 6.11: First set: Effect of dispatching rule factor





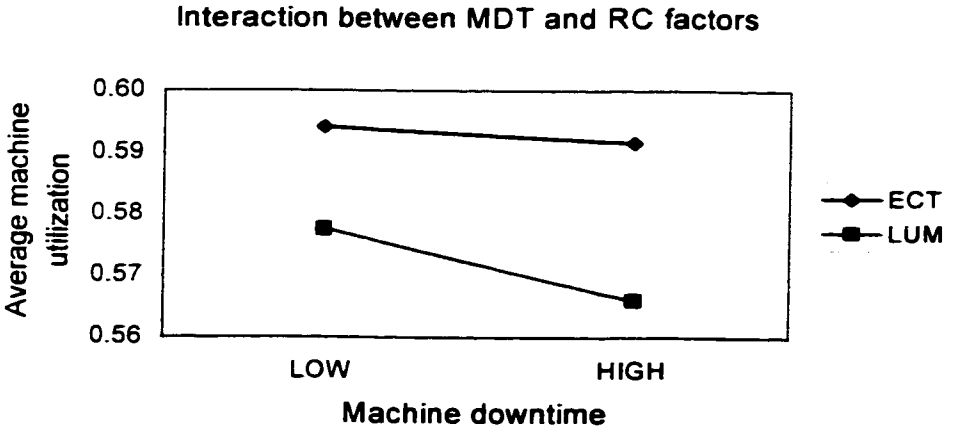


Figure 6.12: First set: Interaction between MDT and RC factors

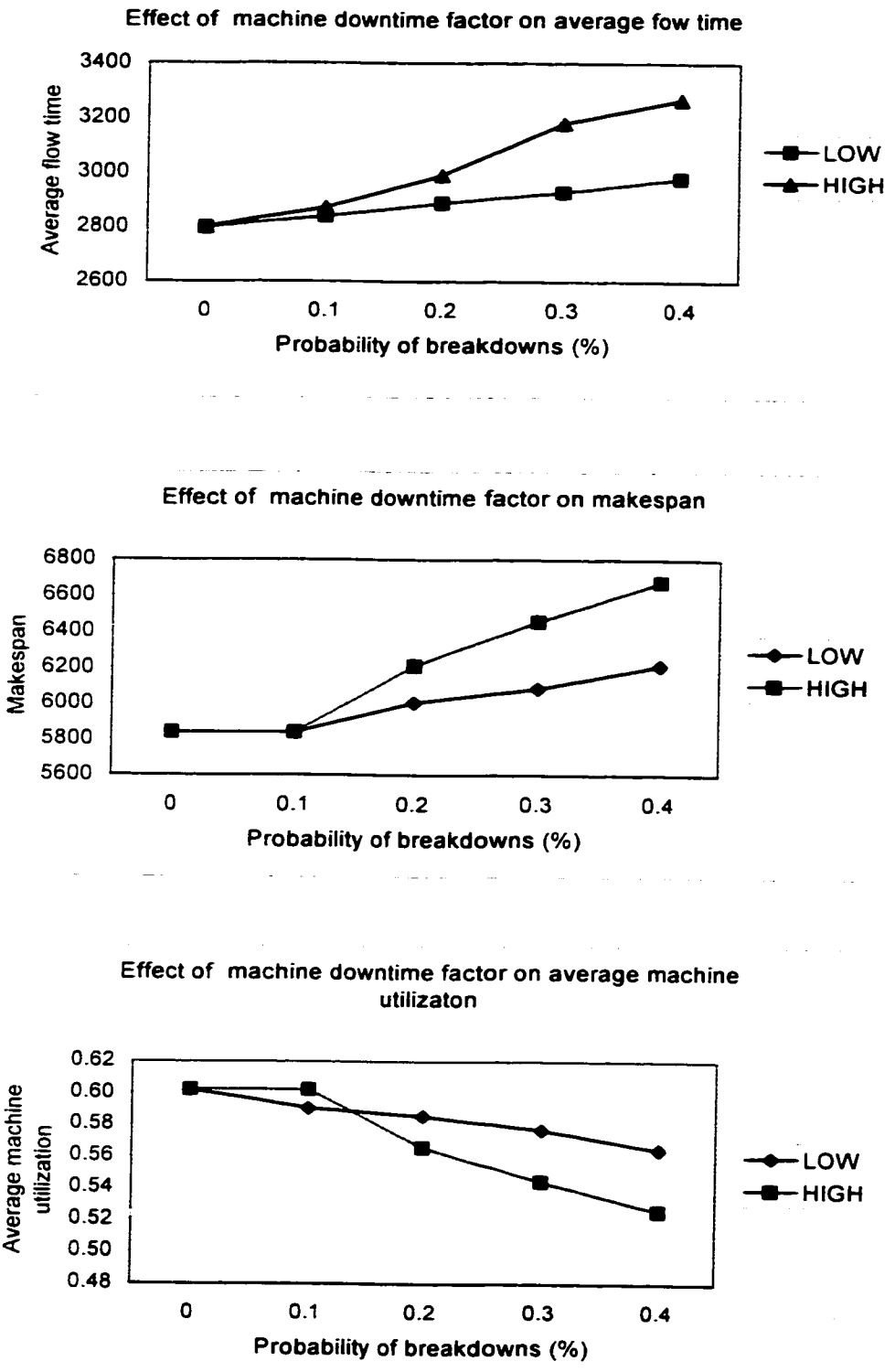
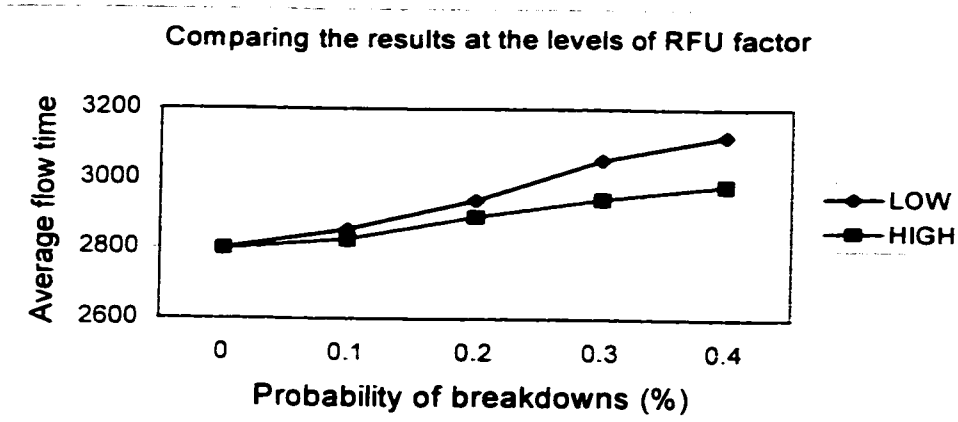
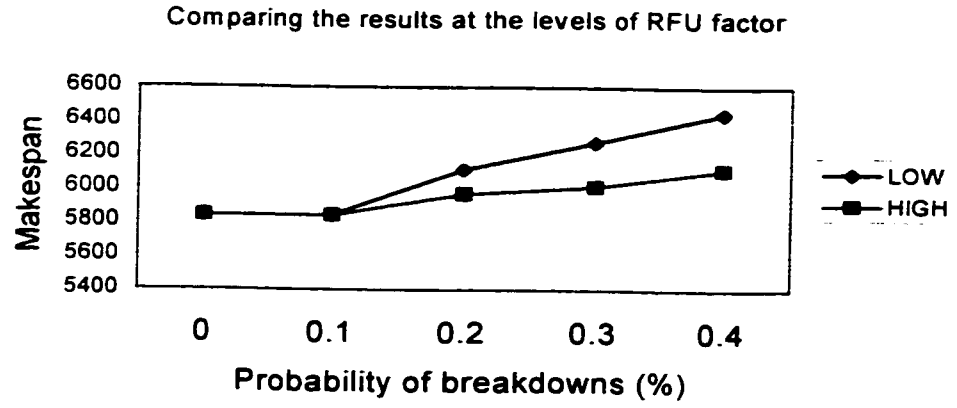


Figure 6.13: Second set: Effect of machine downtime





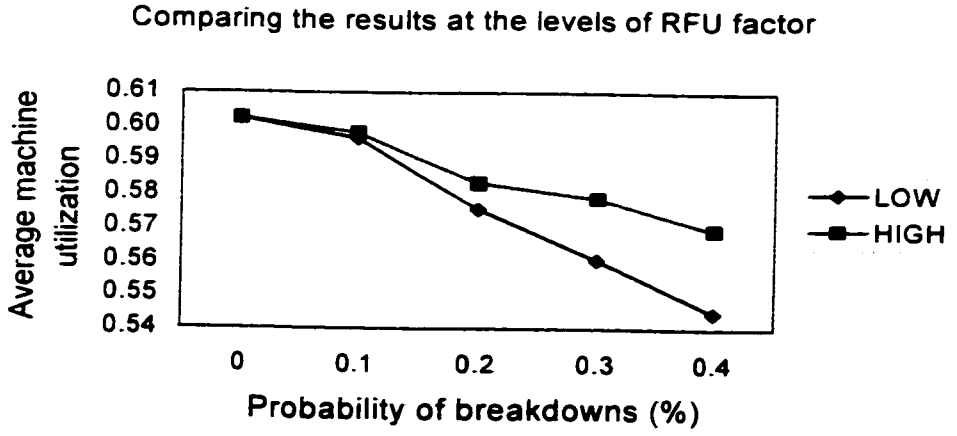


Figure 6.14: Comparing the results at the levels of RFU factor

6.4 CONCLUSION

A rescheduling algorithm is developed to deal with machine breakdowns in real-time. The developed rescheduling algorithm guarantees a deadlock-free new schedule. The existence of alternative routes, availability of material handling facilities, and the limitations of buffer capacities are taken into consideration. The response time is an essential factor when performing rescheduling in real-time. Therefore, rather than rescheduling all jobs, the developed algorithm modifies the portion of the original schedule affected due to the disturbance.

Five factors were considered (see Table 6.4) to study their effect on the system performance that was measured by the average flow time, makespan, and average machine utilization. The analysis of the results indicated that utilizing the system routing flexibility in real-time improves system performance. Moreover, the decision to route the interrupted operation to an alternative machine is based on the minimum expected completion time and results in a better performance compared to that obtained using the lowest utilized machine criterion. The dispatching rules have not affected the system performance significantly.

It is recommended to consider other sources of disturbances such as order cancellation, new order release, and change in order priorities as part of future research. In addition, other measures should be considered, especially those related to the due date. The experiments were performed on a medium size flexible manufacturing system. It is intended to consider larger flexible manufacturing systems and study the effect of other dispatching rules on the system performance.

CHAPTER 7

CONCLUSIONS AND RECOMMENDATIONS

7.1 CONTRIBUTIONS

The main objective of this research was to resolve the deadlock problem, off and on-line, in flexible manufacturing systems using Petri nets. In addition, the performance of systems exhibiting deadlocks was addressed. The generation of deadlock-free schedules using heuristic search was selected as research approach. In the passage of the research, two challenges were faced. First challenge was the scheduling optimization complexity. Second challenge was the guarantee of obtaining deadlock-free schedules. The first challenge was approached by developing truncation techniques and heuristic functions. The second challenge was overcome using the powerful features of Petri nets in analyzing the deadlock problem. The following has been achieved in the research work:

1. Using Petri nets, an efficient search algorithm has been developed. The algorithm is based on the best-first and backtracking technique. The algorithm consists of two stages: Initialization and Backtracking. The Initialization step finds an initial solution and the Backtracking step finds an optimal (or near-optimal) solution. Obtaining a deadlock-free schedule is guaranteed. The algorithm was originally developed in order to optimize the makespan. Another version of the algorithm was developed by modifying the original one in order to optimize the average flow time. The results presented in this research constitute a contribution towards solving the deadlock-free

scheduling problem in FMSs and the development of efficient deadlock-free schedules for large FMSs.

2. By computing the reachability graph (RG) of the Petri net model an optimal schedule can be generated. However, enumerating the whole RG is an exponentially complex problem. Hence, truncation techniques and heuristic function were developed to generate only a portion of the RG and improve the efficiency of the search algorithm. Working towards this direction the following has been achieved:

   2.1. Two truncation techniques are developed and added to the search algorithm. The first truncation technique is based on the structure theory of the Petri nets (minimal siphon property of the Petri net model). This technique deals with the deadlock problem and its objective is to avoid the deadlock situation efficiently during the search process. The second truncation is based on the Petri net structure and the concepts of delay and non-delay scheduling. This technique provides an opportunity for trade-off between the solution quality and the search effort. The technique is called user control factor (*UCF*).

   2.2. Three heuristic functions are developed to reduce the complexity of the scheduling problem when optimizing the average flow time. Each heuristic function is equipped with a parameter (*MF*, see section 4.3.3) to provide a trade-off between the solution quality and the search effort. The developed heuristic functions are the Remaining Processing Time (*RPT*), Average Operation Waiting Time (*AOWT*), and Dispatching Rules (*SDR*).

3. Performance evaluation of manufacturing systems attracted the interest of researchers due to its importance. The traditional techniques, such as simulation and queuing

theory, are not suitable for analyzing the performance of systems that exhibit deadlocks. Therefore, Petri nets were introduced as a tool for analyzing the performance of these systems under different levels of routing flexibility (RF). The effect of other factors such as system loading (SL), machine failure rate (MFR), and processing time variability (PTV) were taken into consideration. The factorial design of the experiments was introduced and the ANOVA method was used for the analysis.

4. Because of the dynamic nature of the recent manufacturing systems, real-time scheduling (such as rescheduling) is a crucial activity of their control. In the literature, the problem of deadlock is ignored when dealing with rescheduling in manufacturing systems. In order to deal with the deadlock problem in real-time the following has been done:

   4.1. A rescheduling algorithm that guarantees a new deadlock-free schedule was developed. The algorithm can deal with machine breakdowns in real-time with taking into consideration the availability of alternative route, availability of material handling facilities, and the limitations of buffer capacities. The developed algorithm modifies the affected portion of the original schedule due to the disturbance rather than rescheduling the whole jobs.

   4.2. Five factors were considered in order to study their effect on the manufacturing system performance that was measured by the average flow time, makespan, and average machine utilization. These factors are (see section 6.3): routing flexibility utilization (RFU), probability of machine breakdowns (PMB), machine downtime (*MDT*), routing criterion (RC), and dispatching rule (DR).

7.2 RESEARCH CONCLUSIONS

The developed scheduling algorithms, rescheduling algorithm, and automatic generation of Petri net models algorithms were implemented in C language in order to validate them and perform the experiments of the three parts of the thesis. In order to deal with large and many case studies it was necessary to automate the process of generating the Petri net models. Two algorithms were developed in order to reach this objective. The developed algorithms receive the process plans of the different part types as an input and generate the corresponding Petri net models. The main conclusions that are related to the three parts of the thesis are presented in the following section.

7.2.1 DEADLOCK-FREE SCHEDULING

- In order to validate the scheduling algorithm, many examples were used and two of them were presented in the thesis [Chen and Jeng (1995), Xiong and Zhou (1997), and Ramaswamy and Joshi (1996)]. The same schedules that were presented in the literature were obtained in much less solution time.
- The effect of using the siphon concept on the solution CPU time is studied by applying the algorithm with and without it on randomly generated cases. The siphon concept enhanced the ability to develop deadlock-free schedules of systems with high number of deadlocks (i.e., uncontrolled siphons). No significant usefulness was found in using the siphon concept for systems with a small number of uncontrolled siphons.
- Using the highest value of *UCF* (i.e. 1) reduces the CPU time dramatically, particularly for higher number of part types (e.g. the CPU time was reduced from 2178.2 seconds to 285.1 seconds at 100-150 part types). In addition, the high value of the *UCF*

does not affect the solution quality. Therefore, unforced idleness in those generated scheduling problems (i.e., delay schedules) does not lead to better average flow time.

- The results obtained using the three heuristic functions exhibit the same trend. The average flow time decreases and the solution time increases as the *MF* (an arbitrary parameter, see section 4.3.3) value increases. In other words, a low value of *MF* results in better average flow time but higher solution time.

- The developed method helped in handling larger systems [compared to Jeng and Chen (1998)]. In addition, the use of the *MF* parameter in the heuristic functions provided a trade-off between the solution quality and solution time. The resulting increase (average of 14%) in the obtained average flow time is less significant than the reduction in the solution time (average of 71%). The Average Operation Waiting Time (*AOWT*) function outperformed the other two functions with respect to the obtained average flow time and solution time.

7.2.2 PERFORMANCE EVALUATION OF SYSTEMS EXHIBITING DEADLOCKS

- This part of the research was based on the hypothesis: routing flexibility will increase the probability of deadlock occurrence which in tern will negatively affect the system performance. The results agreed with the first part of the hypothesis. On the other hand, it was found that increasing routing flexibility improves the system performance (average flow time) in systems exhibiting deadlocks. Furthermore, the highest gain of system performance improvements, due to routing flexibility, was obtained when routing flexibility changes from RF2 (low) to RF3 (medium).

- The benefits of routing flexibility of a manufacturing system can not be utilized without having an efficient real-time scheduler, an efficient tool management system, and flexible fixture devices. Therefore, the gain of routing flexibility and its corresponding cost should be considered when setting the routing flexibility level of a manufacturing system.

- It was found that routing flexibility improving the system performance whatever the level of machine reliability, system loading, or processing time variability. Even though the routing flexibility has its highest effect if the system is at the highest levels of system loading and processing time variability.

7.2.3 DEADLOCK-FREE RESCHEDULING

- In analyzing the system performance in real-time (using the rescheduling algorithm) under different routing criteria, it was found that the expected completion time (ECT) criterion dominates the lowest machine utilization (LMU) criterion. In other words, routing the interrupted operations (due to machine breakdown) gives the best system performance. The results of the test cases showed that the improvement in system performance due to using ECT over LMU reaches its maximum value (average of 2.1-4.5 %) at the high level of machine downtime.

- The results showed that the deterioration in the system performance is not significant (average of 1.4-6.3%) at the low levels of machine downtime and probability of breakdowns. While the combination of high machine downtime and high probability of breakdowns resulted in a significant (average of 7.8-16.8%) system performance deterioration.

- The system performance was not significantly affected by the dispatching rule (DR) factor. It was found that the SPT rule gives the best makespan and average machine utilization compared to the FCFS rule. Indeed, utilizing the routing flexibility in real-time improves the system performance.

7.3 DISCUSION AND RECOMMENDATIONS

1. The main objective of this part was to resolve the deadlock problem with optimizing the schedule of flexible manufacturing systems (FMSs). The advantage of this approach of deadlock resolution is in its high utilization of the system flexibility. Moreover, it does not impose complexity on the system control. As mentioned in chapter 4, the effort of deadlock-free scheduling is composed of searching for optimality and avoiding deadlock states. If the system has many shared resources, with significant interaction between them, imposed by the produced parts process plans, a significant time is usually spent in avoiding deadlock states. In this research, a truncation technique has been developed in order to reduce this search effort. The truncation technique is using another algorithm [Abdallah, ElMaraghy, and ElMekkawy (1997)] which has an exponential complexity. Therefore, more research should be done in developing other truncation techniques and heuristic that deal with this problem particularly for solving large systems.
2. Although the developed algorithm combined with the truncation techniques and heuristic functions helped in handling larger systems compared to the previously reported results in the literature, these approaches still can not handle real-life problem sizes. Simulation with dispatching rules could be used in generating off-line

schedules. Unfortunately, the traditional simulation modeling can not handle the deadlock problem. Therefore, simulation must be based on tools (such as Petri nets, Graph theoretic approaches, or mathematical models) that take this problem into consideration when modeling the manufacturing systems. For example, The Initialization stage of the developed algorithm, in this research, can be used as a simulator for the PN model. Therefore, for very large systems, some dispatching rules can be tested to obtain good deadlock-free schedules

3. In this research two measures of performance were considered: makespan and average flow time. Despite the importance of both measures, management usually considers meeting the delivery date the highest priority. Therefore, it is recommended to modify the algorithm and heuristic function to deal with due date measures. Moreover, considering multi-objective optimization is a practical perspective. For example, meeting the due dates can be considered as a constraint in the model formulation while minimizing both of the outsourcing and overtime in the same time. Other combination of these practical measures can be considered in a future research.
4. One of the prime goals of this research was to develop truncation techniques and heuristic function that trade-off between solution quality and search effort by equipping them with an adjustable parameter. This is considered an ambitious approach that can help in either solving scheduling problem in a dynamic environment or relaxing the optimality condition in order to solve complex problems. If the problem is not large or the solution time is not a crucial concern then it is recommended to use a parameter value (in the truncation technique or heuristic function) that can obtain the highest possible solution quality.

5. As stated in section 7.1, the routing flexibility improved the system performance of systems exhibiting deadlocks. Therefore, it is recommended to take this into consideration when controlling these types of systems. In addition, it is recommended to have routing flexibility whatever the level of machine reliability, system loading, or processing time variability. Because the use of routing flexibility showed an advantage when it was used at the different levels of these factors.
6. Using one route for each part type simplifies the control of the manufacturing system but deteriorates the system performance. On the other hand, increasing the parts routing flexibility complicates the manufacturing system control and acquires additional expenses such as buying flexible machines (that can manufacture wide range of part types), efficient tool management system, and flexible fixture devices. Therefore, It is highly recommended to develop models that maximizing the system performance and minimize the associated costs at the same time. These models could be generic in a way that can be applied to different types of manufacturing systems.
7. The different types of manufacturing flexibility are not independent. In other words, a manufacturing system may not have routing flexibility without having machine flexibility, labor flexibility, material handling flexibility and operation flexibility. Therefore, the relationship between these types of flexibility needs to be firmly established.
8. In addition to optimizing the performance of the manufacturing systems, generating off-line schedules is essential in order to control them. These schedules are usually susceptible to uncertainties such as machine breakdowns, new order release, order cancellation, and over or under estimation of processing times. In this research, the

machine breakdowns were considered. Therefore, it is recommended to develop other rescheduling algorithms that can deal with other sources of uncertainties especially for systems exhibiting deadlocks.

REFERENCES

Abdallah I.B., ElMaraghy H.A., and ElMekkawy T.Y., "A Logic Programming Approach for Finding Minimal Siphons in S^3PR Nets Applied to Manufacturing Systems", IEEE International Conference on Systems, Man, and Cybernetics. Orlando, Florida, vol. 2, pp. 1710-1715, Oct. 12-15, 1997.

Abdallah I.B., and ElMaraghy H.A., "Deadlock Prevention and Avoidance in FMS: A Petri Net Based Approach", Int. J. Adv. Manuf. Technol. , 14, pp. 704-715, 1998-a.

Abdallah I.B., and ElMaraghy H.A., "A Deadlock-Free Scheduling Method for a Class of FMS", 29th CIRP Int. Seminar on Manufacturing, Osaka University, Osaka, Japan, pp. 257-263, 1998-b.

Abdallah I.B., ElMaraghy H.A., and ElMekkawy T.Y., "Heuristic Search for Deadlock-free Schedule in FMS Using Petri Nets", Proc. of the IEEE Int. Conf. on Robotics and Automation, Leuven, Belgium, May 16-21, 1998-a.

Abdallah I.B., ElMekkawy T.Y., and ElMaraghy H.A., "On Deadlock-free Scheduling in FMS", IEEE Int. Conf. on Systems, Man, and Cybernetics, San diego, California, USA, October 11-14, 1998-b.

Abumaizar R.J., and Svestka J.A., "Rescheduling Job Shops Under Disruptions", Int. J. Prod. Res., 35(7), pp. 2065-2082, 1997.

Aytug H., Bhattacharyya S., Koehler G.J., and Snowdon J.L., "A Review of Machine Learning in Scheduling", IEEE Trans. on Engineering Management, 41(2), pp. 165-171, 1994.

Banazak Z.A., and Krogh B.H., "Deadlock Avoidance in Flexible Manufacturing Systems with Concurrently Competing Process Flows", IEEE Trans. on Robotics and Automation, 6(6), pp. 724-734, 1990.

Banerjee A., and Flynn B.B., "A Simulation Study of Some Maintenance Policies in a Group Technology Shop", Int. J. Prod. Res., 25 (11), pp. 1595-1609, 1987.

Barkaoui K., and Alloua B., "Performance of alternative strategies for dealing with deadlocks in FMS", IEEE Symposium on Emerging Technologies & Factory Automation, Los Angeles, CA, , USA, pp. 281-286, 1997.

Barkaoui K., and Abdallah I.B., " A Deadlock Prevention Method for a Class of FMS", IEEE Int. Conf. on System Man and Cybernetics, Vancouver, British Columbia, Canada, pp. 4119-4124, October 1995.

Basnet C., and Mize J.H., "Scheduling and Control of Flexible Manufacturing Systems: a Critical Review", Int. J. Computer Integrated Manuf., 7(6), pp. 340-355, 1994.

Bateman N., Stockton D.J., and Lawrence P., "Measuring the mix response flexibility of manufacturing systems, " International Journal of Production Research, 37(4), pp. 871-880, 1999.

Beach R., Muhlemann A.P., Price D.H.R., Paterson A., Sharp J.A., "Review of manufacturing flexibility, " European Journal of Operational Research, 12(1), pp. 41-57, 2000.

Benjaafar S., "Models for Performance Evaluation of Flexibility in Manufacturing Systems", Int. J. Prod. Res., 32(6), pp. 1383-1402, 1994.

Benjaafar S., and Ramakrishnan R., "Modeling, Measurements and Evaluation of Sequencing Flexibility in Manufacturing Systems, "Int. J. Prod. Res., 34(5), pp. 1195-1220, 1996.

Bernardo J.J., and Mohamed Z., "The Measurement and Use of Operational Flexibility in the Loading of Flexible Manufacturing Systems, " European Journal of Operational Research, 60, pp. 144-155, 1992.

Berry W.L., and Cooper M.C., "Manufacturing flexibility: Methods for measuring the impact of product variety on performance in process industries, " Journal of Operations Management, 17(2), pp. 163-178, 1999.

Brill P.H., and Mandelbaum M., "On Measures of Flexibility in Manufacturing Systems", Int. J. Prod. Res., 27, pp. 747-756, 1989.

Browne J., Dubous D., Rathmill K., Sethi S.P., and Stecke K.E., "Classification of Flexible Manufacturing Systems",The FMS Magazine, 2, pp.114-117, 1984.

Burton J.S., Banerjee A., and Sylla G., "A Simulation Study of Sequencing and Maintenance Decisions in Dynamic Job Shop. "Computers Ind. Engng., 17 (1), pp. 447-452, 1989.

Buzacott J.A., "Modeling Manufacturing Systems", Robotics and Computer Integrated Manuf., 2, pp. 25-32, 1985.

Buzacott J.A., "The Fundamental Principles of Flexibility in Manufacturing Systems", Proc. of the 1st Int. Conf. on Flexible Manuf. Systems, Brighto, UK, October 22-22, 1982.

Buzacott J.A., Yao D., "Flexible Manufacturing Systems: a Review of Analytical Models", Management Science, 32, pp. 890-905, 1986.

Camurri P., Franchi P., Gandolfo F., and Zaccaria R., "Petri Net Based Process Scheduling: a Model of the Control System of Flexible Manufacturing Systems", J. of Intelligent and Robotic Systems, 8, pp. 99-123, 1993.

Caprihan R., and Wadhwa S., "Impact of Routing Flexibility on the Performance of an FMS-A Simulation Study", International Journal Flexible Manufacturing Systems, 9, pp. 273-298, 1997.

Carlier J., and Chretiene P., "Timed Petri Nets Schedules", in Rozenberg, editor, Advances in Petri Nets, LNCS 340, pp. 62-84, 1988.

Carter M.F., "Designing Flexibility into Automated Manufacturing System", Proc. of the 2nd ORSA/TIMS Conf. on Flexible Manuf. Systems, Ann Arbor, MI, K.E. Steke, and R. Suri (Eds.), Elsevier, Amesterdam, The Netherlands, pp. 107-118, 1986.

Chandra P., and Tombak M.M., "Models for the Evaluation of Routing and Machine Flexibility", European Journal of Operational Research, 60, pp. 156-165, 1992.

Chang Y.L., Sullivan R.S., and Bagchi U.S., "Experimental Investigation of Real-time Scheduling in Flexible Manufactuirng Systems, "Proceedings of the First ORSA/TIMS Conference on Flexible Manufactuirng Systems, K. E. Stecke and R. Suri, Editors. University of Michigan, Ann Arbor, MI, pp. 307312, August 1984.

Chatterjee R., Cohen M. A., and Maxwell W.L., "A Planning Framework for Flexible Manufacturing Systems", WP # 87-07-04, University of Pennsylvania, Philadelphia, PA, 1987.

Chen J., and Chung C.H., "Effects of Loading and Routing Decisions on Performance of Flexible Manufacturing Systems", Int. J. Prod. Res., 29(11), pp. 2209-2225, 1991.

Chen J., and Chung C.H., "An Examination of Flexibility Measurements and Performance of Flexible Manufacturing Systems", Int. J. Prod. Res., 34(2), pp. 379-394, 1996.

Chen S.C., and Jeng M.P., "Heuristic Approach Based on the State Equations of Petri Nets for FMS Scheduling", Proc. of the IEEE Int. Conf. On Industrial Automation and Control: Emerging Technologies, Piscataway, NJ., USA, pp. 275-281, 1995.

Coffman E.G., Elphick M.J., and Shoshani A., "System Deadlock", Computer Surveys, 3 (2), pp. 67-78, 1971.

Cott J., and Macchietto S., "Minimizing the Effects of Batch Process Variability Using On-line Schedule Modification", Comput. Chem. Engng., 13, pp. 105-113, 1989.

D'Souza D.E., and Williams F.P., "Toward a taxonomy of manufacturing flexibility dimensions, " Journal of Operations Management, 18(5), pp. 577-593, 2000.

Dutta A., "Reacting to Scheduling Exceptions in FMS Environment", IIE Transactions. 22(4), pp. 300-314, 1990.

ElMekkawy T.Y., Abdallah I.B., and ElMaraghy H.A., "A Heuristic Search Approach for Deadlock-free Scheduling in FMSs Using Petri Nets and AI Techniques", ASME, Computers in Engineering Conference, Atlanta, Georgia, September 13-16, pp. 1-8, 1998.

Ezpeleta J., Colom J.M., and Martinez J., "A Petri Net Based Deadlock Prevention Policy for Flexible Manufacturing Systems", IEEE Trans. on Robotics and Automation, 11(2), pp. 173-184, 1995.

Ezpeleta J., Colom J.M., " Automatic synthesis of colored Petri nets for the control of FMS", IEEE Transactions on Robotics and Automation, 13(3), p 327-337, 1997.

Gargeya V.B., and Deane R.H., "Scheduling in the dynamic job shop under auxiliary resource constraints: A simulation study, " International Journal of Production Research, 37(12), pp. 2817-2834, 1999.

Ferrarini L., Piroddi L., and Allegri S., "A Comparative Performance Analysis of Deadlock Avoidance Control Algorithms for FMS, " Journal of Intelligent Manufacturing, Vol. 10, pp. 569-585, 1999.

Garmhausen V.H., Clarke E.M., and Campose S., "Deadlock Prevention in Flexible Manufacturing Systems Using Symbolic Model Checking", IEEE Int. Conf. on Robotics and Automation, Minneapolis, pp. 527-532, April 1996.

Gerwin D., "An Agenda for Research on the Flexible Manufacturing Systems", Int. J. of Operations and Production Management, 7(1), pp. 38-49, 1987.

Gonzalez L.R., "On-line Knowledge-based Simulation for FMS: a State of the Art Survey", Proc. of the 1995 Winter Simulation Conf., pp. 1057-1061.

Gupta Y.P., Gupta M.C., and Bector C.R., "A Review of Scheduling Rules in Flexible Manufacturing Systems", Int. J. Computer Integrated Manuf., 2(6), pp. 356-377,1989.

Gupta Y.P., Evans G.W., and Gupta M.C., "A Review of Multi-criterion Approaches to FMS Scheduling Problems", Int. J. Prod. Economics, 22, pp. 13-31, 1991.

Gupta Y.P., and Sameer G., "Flexibility trade-offs in a random flexible manufacturing system. A simulation study", International Journal of Production Research, 30(3), pp. 527-557,1992.

Gupta Y.P., "On Measurement and Valuation of Manufacturing Flexibility", Int. J. Prod. Res., 31(12), pp. 2947-2958, 1993.

Harmonosky M., and Robohn S.F., "Real-time Scheduling in Computer Integrated Manufacturing: a Review if Recent Research", Int. J. Computer Integrated Manuf., 4(6), pp. 331-340, 1991.

Harmonosky M., "Simulation-based Real-time Scheduling: Review of Recent Development", Proc. of the 1995 Winter Simulation Conf., pp. 220-225.

Isloor S.S., and Marsland T.A., "The Deadlock Problem: an Overview", Computer, 13(9), pp. 58-78, 1980.

Jain K., and ElMaraghy H. A., "Production Scheduling/Rescheduling in Flexible Manufacturing", Int. J. Prod. Res., 35(1), pp. 281-309, 1997.

Jeng M.D., Lin C.S., Chung S., and Yi S., "Petri net dynamics-based scheduling of flexible manufacturing systems with assembly, " Journal of Intelligent Manufacturing, 10(6), pp. 541-555, 1999.

Kim M.H., and Kim Y., "Simulation-based Real-time Scheduling in a Flexible Manufacturing System", J. of Manuf. Systems, 13, pp. 85-93, 1994.

Kumaran T.K., Chang W., Cho H., and R.A. Wysk, "A Strutured Approach to Deadlock Detection, Avoidance, and Resolution in Flexible Manufacturing Systems", Int. J. Prod. Res., 32(10), pp. 2361-2379, 1994.

Kundu S., and Akyildiz I.F., "Deadlock Free Buffer Allocation in Closed Queuing Networks", Queuing Systems, 4, pp. 47-56, 1989.

Lawley M., Reveloitis S., and Ferreira P., "Design Guidelines for Deadlock-Handling Strategies in Flexible Manufacturing Systems", Int. J. of Flexible Manufacturing Systems, Vol. 9, pp. 5-30, 1997.

Lee D.Y., and Dicesare F., "Scheduling Flexible Manufacturing Systems Using Petri Nets and Heuristic Search", IEEE Trans. on Robotics and Automation, 10(2), pp. 123-132, 1994.

Leung Y.T., and Sheen G.H., "Resolving Deadlocks in Flexible Manufacturing Cells", J. of Manuf. Systems, 12(4), pp. 291-304, 1993.

Lie C. H., Hwang C. L., and Tillman F.A., "Availability of Maintained Systems: A State of the art Survey",AIIE Transactions, Vol. 9, pp. 247-259, 1987.

Li R., Shyu Y., and Adiga S., "A Heurstic Rescheduling Algorithm for Computer-based Production Scheduling Systems", Int. J. Prod. Res., 31(8), pp. 1815-1826, 1993.

Lin G.Y., and Solberg J.J., "Effectiveness of Flexible Routing Control", International Journal of Flexible Manufacturing Systems, 3, pp. 189-211, 1991.

Lin C.R., "Alternative shop floor control of a flexible manufacturing system - a temporal rescheduling approach, " International Journal of Industrial Engineering: Theory Applications and Practice, 7(2), USA pp. 168-178, 2000.

Mahmoodi F., Mosier C.T., and Morgan J.R., "Effects of scheduling rules and routing flexibility on the performance of a random flexible manufacturing system, " International Journal of Flexible Manufacturing Systems, 11(3), pp. 271-289, 1999.

Mandelbraum M., 'Flexibility in Decision Making: An Exploration and Unification,' Ph.D. Thesis Dept. of Industrial Engineering, University of Toronto, Canada, 1978.

Masory O., "An Integrated Approach to Simulation, Control, and Real-time Diagnostic of Manufacturing Systems Based on Adaptive Augmented Timed Petri Nets", Trans. of NAMRI/SME, 1994.

Melcher A.J., and Booth D., "Advanced Manufacturing Systems: Characteristics and Strategic Implications, " ORSA/TIMS, St. Louis, MO, October 25-28, 1987.

Merchawi N.S. and ElMaraghy H.A, "Analytical approach for using simulation in real-time decision making in FMSs", Journal of Manufacturing Systems, 17(6), pp. 418-435, 1998.

Minoura T., and Ding C., "A Deadlock Prevention Method for a Sequence Controller for Manufacturing Control", Int. J. of Robotics and Automation, 6(3), pp. 149-155, 1991.

Montgomery D.C., "Design and Analysis of Experiements".John Wiely, 1976.

Nagarur N., "Some Performance Measures of Flexible Manufactuirng Systems", Int. J. Prod. Res., 30, pp. 799-809, 1992.

Newman W.E., "Models to Evaluate the Benefits of FMS Pallet Flexibility", In Proc. of the 2nd ORSA/TIMS Conf. on Flexible Manuf. Systems, Ann Arbor, MI, Stecke K.E., and Suri R. (Eds.), Elsevier, Amesterdam, The Netherlands, pp. 209-220, 1986.

Onaga K., Silva M., and Watanabe T., "On Periodic Schedules for Deterministically Timed Petri Net Systems", Proc. of the 4th Int. Workshop on Petri Nets and Performance models, Melbourne, Australia, Dec. 2-5, pp. 210-215, 1991.

Pearl J., "Heuristics: Intelligent Search Strategies for Computer Problem Solving", Reading, MA, Addison-Wesley, 1984.

Pierce N.G., and Yurtsever T., "Dynamic dispatch and graphical monitoring system, " IEEE International Symposium on Semiconductor Manufacturing Conference, Proceedings, Santa Clara, CA, , USA, pp. 65-68, Oct 11-Oct 13 1999.

Pinedo M., "Scheduling: Theory, Algorithms, and Systems", Prentice-Hall, New Jersey, USA, 1995.

Piramuthu S., Shaw M., and Fulkerson B., "Information-based dynamic manufacturing system scheduling, " International Journal of Flexible Manufacturing Systems, 12(2), pp. 219-234, 2000.

Qi J.G., Burns G.R., and Harrison D.K., "Application of parallel multipopulation genetic algorithms to dynamic job-shop scheduling, " International Journal of Advanced Manufacturing Technology, 16(8), pp. 609-615, 2000.

Ramaswamy S.E., and Joshi S.B., "Deadlock-free Schedules for Automated Manufacturing Workstations", IEEE Trans. on Robotic and Automation, 12(3), pp. 391-400, 1996.

Reveliotis S.A., "An Analytical Investigation of the Deadlock Avoidance Versus Detection and Recovery Problem in Buffer-Space Allocation of Flexibly Automated Production Systems, " IEEE Trans. on Systems, Man and Cybernetics, 30(5), pp. 799-811, 2000.

Reynolds R., "Knowledge-based Production Scheduler Reacts in Real-time", Chilton's IECS, 61(7), pp. 45-46, 1988.

Rovito V.P., and Dvorsky R.E., "Planning Models for Designing FMS, "Technical Report, Cincinnati Milacrom, Cincinnati, Ohio, 1984.

Sagi S. R., Chen F. F., and Wu D., "An Intelligent Real-time Shop Floor Control System", Proc. of the sixth Int. Conf. on AI and Expert Systems Applications, Houston, TX, pp. 245-250, December 1994.

Sarker B. R., Krishnamurthy S., and Kuthethur S.G., "A Survey and Critical Review of Flexibility Measures in Manufacturing Systems, " Production Planning and Control, 5(6), pp. 512-523, 1994.

Schmidt F.B., and Ranta J., "FMS World Data Bank", International Institute for Applied Systems Analysis (IISA), May 1989.

Schneeweiss C., and Schneider H., "Measuring and designing flexibility as a generalized service degree, " European Journal of Operational Research, 12(1), pp. 98-106, 1999.

Sethi K., and Sethi S. P., 'Flexibility in Manufacturing: A Survey,' Int. J. of Flexible Manuf. Systems, 2(4), pp. 289-328, 1990.

Shanker K., and Tzen Y.J., "A Loading and Disptaching Problem in a Random Flexible Manufacturing System. International Journal of Production Research. 23(2), pp. 579-595. 1985.

Shen L., Chen Q., and Luh J.Y.S., "Truncation of Petri Net Models for Simplifying Computation of Optimum Scheduling Problems. " Computers in Industry. 20. pp. 25-43, 1992.

Shih H., and Sekiguchi T., "A Timed Petri Net and Beam Search Based On-line FMS Scheduling System with Routing Flexibility", Proc. of the 1991 IEEE Int. Conf. on Robotics and Automation. Sacramento. CA, pp. 2548-2553, April 1991.

Shukla S., and Chen F.F., "The State of the Art in Intelligent Real-time FMS Control: a Comprehensive Survey", J. Intelligent Manuf., 7, pp. 441-455. 1996.

Slack N., 'The Flexibility of Manufacturing Systems.' Int. J. of Operations and Production Management, 7(4), pp. 35-45. 1987.

Smith D.J., "Reliability and Maintainability in Perspective", Macmillan Publishers LTD., 1985.

Smith J.S., Wysk R. A., D.T. Sturrock, S.E. Ramaswamy, G.D. Smith, and S. B. Joshi, "Discrete Event Simulation for Shop Floor Control", Proc. of the 1994 Winter Simulation Conf., pp. 962-969.

Son Y.K., and Park C.S., "Economic Measure of Productivity, Quality and Flexibility in Advanced Manufacturing Systems", Journal of Manufactuirng Systems. 6. pp. 193-206. 1987.

Stecke K. E., "Production Planning Problems for Flexible Manufacturing Systems", Ph.D. Dissertation. Purdue University. West Lafayette. Indiana. 1981.

Stecke K.E., and Browne J., "Variations in Flexible Manufacturing Systems According to the Relevant Types of Automated Materials Handling", Material Flow. Vol. 2, pp. 179-185. 1985.

Stecke K.E., and Rachamadugu R., "Classification and Review of FMS Scheduling Procedures", Prod. Planning and Control. 5(1), 1994. 2-20.

T. Sun, and L. Fu, "An automatic Petri-net generator for modeling a flexible manufacturing system", Proc. of the IEEE International Conference on Systems, Man and Cybernetics, Antonio, TX, USA, 1, p 670-675, Oct 2-5, 1994.

Suri R., "An Overview of Evualtive Models for Flexible Manufacturing Systems", Annals of Operations Research, 3, pp. 13-21, 1985.

Szelke S., and Kerr R.M., "Knowledge-based Reactive Scheduling", Prod. Planning and Control, 5(2), pp. 124-145, 1994.

Tawegoum R., Castelain E., Gentina J.C., "Real-time Piloting of Flexible Manufacturing Systems", European J. of Operational Research, 78, pp. 252-261, 1994.

Terpstra V.J., Bruijckere R.M., and Verbruggen H.B., "A Prototype of a Reactive Scheduler for Mixed batch/continuous Plants", ESCAPE 4: Fourth European Symposium on Computer Aided Process Engineering, Rugby U. K., 1994.

Tipi N.S., and Bennett S., "Dispatching rules for scheduling and feedback control in a virtual enterprise: a simulation approach, " International Journal of Flexible Automation and Integrated Manufacturing, 7(3), pp. 293-303,1999.

Toni D.E., and Tonchia S., 'Manufacturing Flexibility: a Literature Review,' Int. J. Prod. Res., 36(6), pp. 1587-1617, 1998.

Tsubone H., and Horikawa M., "A Comparison Between Machine Flexibility and Routing Flexibility, " International Journal of Flexible Manufacturing Systems, 11, pp. 83-101, 1999.

Viswandaham N., Narahari Y., and Johnson T.L., "Deadlock Prevention and Deadlock Avoidance in Flexible Manufacturing Systems Using Petri Nets", IEEE Trans. on Robotics and Automation, 6(6), pp. 713-723, 1990.

Watatani Y., and Fujji S., "A Study on Rescheduling Policy in Production Systems", Proc. of the Japan/USA Symposium on Flexible Automation, ASME, 2, pp. 1147-1150, 1992.

Wilhelm W.E., and Shin H. M., "Effectiveness of Alternate Operations in a Flexible Manufactuirng System", International Journal of Production Research, 23(1), pp. 65-79, 1985.

Wittrock R. J., "An Adaptive Schedulig Algorithm for Flexible Flow Lines, " Operations Research, 36 (3), pp. 445-453, 1988.

Wu D. S., Stopper H., and Chang P. C., "One Machine Rescheduling Heuristics with Efficiency and Stability as Criteria", Computers in Operations Research, 20 (1), pp. 1-14, 1993.

Wu N., "Necessary and sufficient conditions for deadlock-free operation in flexible manufacturing systems using a colored Petri net model." IEEE Transactions on Systems, Man and Cybernetics, 29(2), p 192-204, 1999.

Wysk R.A., Yang N.S., and Joshi S., "Detection of Deadlocks in Flexible Manufacturing Cells", IEEE Trans. on Robotics and Automation, 7(6), pp. 853-859, 1991.

Wysk, R.A., Yang N.S., and Joshi S., "Resolution of deadlocks in flexible manufacturing systems: Avoidance and recovery approaches", Journal of Manufacturing Systems. 13(2). pp. 128-136. 1994.

Xiong K., and Chen H., "Deadlock Avoidance Policy for Flexible Manufactuirng Systems". In M.C. Zhou, editor, Petri nets in Flexible and Agile Automation. pp. 239-263. Kluwer Academic Publishers, 1995.

Xiong H.H., and Zhou M.C., "Deadlock-free Scheduling of an Automated Manufacturing Systems Based on Petri Nets", Proc. of the IEEE Int. Conf. on Robotics and Automation, pp. 945-950. Albuquerque. New Mexico. April 1997.

Xue Y., Kieckhafer R.M. and Choobineh F.F. "Automated construction of GSPN models for flexible manufacturing systems", Computers in Industry, 37(1), pp. 17-25. 1998.

Yamamoto M., and Nof S.Y., "Scheduling/rescheduling in the Manufacturing Operating System Environment", Int. J. Prod. Res., 23(4), pp. 705-7 22. 1985.

Zelenovich M., 'Flexibility: A Condition for Effective Production Systems.' Int. J. Prod. Res., 20(3), pp. 319-337, 1982.

Zhang Y., and Chen H., "Knowledge-based dynamic job-scheduling in low-volume/high-variety manufacturing, " Artificial Intelligence in Engineering, 13(3), pp. 241-249, 1999.

Zhou M.C., Chin H.S., and Xiong H.H., "Petri Net Scheduling of FMS Using Branch and Bound Methods", Proc. of the IEEE Industrial Electronics Conf., Los Alamitos, CA. 1, pp. 211-216, 1995.

Zuberek W.M., and Kubiak W., "Timed Petri nets in modeling and analysis of simple schedules for manufacturing cells", Proceedings of Computers and Mathematics with Applications, the 6th International Workshop of the Bellman Continuum, 37(11), Hachioji Tokyo Jpn, pp. 191-206, 1999.

Zukin M., and Dalcol P.R., "Manufacturing flexibility: assessing managerial perception and utilization, " International Journal of Flexible Manufacturing Systems, 12(1), pp. 5-23, 2000.

Zurawski R., and Zhou M.C., "Petri Nets and Industrial Applications: A Tutorial", IEEE Trans. on Industrial Electronics, 41(6), pp. 567-583, 1994.

APPENDIX 1

THE DEADLOCK-FREE SCHEDULING ALGORITHM

The following notations are used in the search algorithm.

Notations:

- m_o: initial marking of the PN model.
- m_f: final marking of the PN model.
- *m* : current marking of the PN model.
- *P_R_T*: a vector represents the ready times of tokens in every place at time t.
- *K*: the expected maximum number of transitions to be fired to reach mf from mo.
- *k*: current search step number, k=1,2,3,....K.
- *ubv*: value of the upper bound of average flow time.
- *cv*: current value of the average flow time.
- *U_B_S*: sequence of transitions which gives *ubv*.
- *C_S*: the current sequence of transitions.
- *Clock_time*: current clock time.
- *RN*: random number, $0.0 \leq RN \leq 1.0$.
- Data[k]:
  - *m[k]*:marking of the PN model at the kth step.
  - *P_R_T[k]*: place ready time at the kth step.
  - *TE_set[k]*: list of enabled transitions based on P_R_T at the kth step.
  - *SE_set[k]*: list of enabled transitions without considering time at the kth step.
  - *E_L[k]*: list of completion times of operations in progress at kth step.

- *Clock_time[k]*: the value of *Clock_time* at the kth step.

Initialization:

Input: *m*o, *m*f, and K;

Output: *ubv* and *U_B_S*;

1. $m = m_o$;
2. k = 1;
3. Estimate Data[k] ;
4. If($m = m_f$) go to (10) ;
5. If(*SE_set[k]* = ϕ) go to (9) ;
6. If(*TE_set[k]* = ϕ) go to (7) ;
   6.1. select t from *TE_set[k]* ;
   6.2. delete t from *TE_set[k]* and *SE_set[k]* ;
   6.3. if *(Test0* =False*)* go to (9) ;
   6.4. *U_B_S[k]* = t ;
   6.5. Fire t and update *m* , and *P_R_T* ;
   6.6. *k* = *k* +1 ;
   6.7. *Estimate Data[k]* ;
   6.8. *Go* to (4) ;
7. Clock_time = min. { *E_L[k]* } ;
8. Update Data[k]; go to (6) ;
9. Do backtracking, go to (5) ;
10. *Estimate ubv of the obtained schedule* ; stop.

Backtracking:

Input: *ubv* and *U_B_S;*

Output: Optimal (or near optimal) firing transitions sequence and the minimal average flow time ;

1. Copy *U_B_S* to C_S :
2. k = k -1 ;
3. If (k<0) stop ;
4. *m = m[k]* ;
5. If (*m* = m_f) go to (11) ;
6. If (SE_set[k] = ϕ) go to (10) ;
7. If (*TE_set[k]* = ϕ) go to (8) ;
   - 7.1. select t from *TE_set[k]* ;
   - 7.2. Delete t from *TE_set[k]* and *SE_set[k]* ;
   - 7.3. If (Test0 =False OR Test1=False OR Test2=False) go to (10) ;
   - 7.4. *C_S[k] = t* ;
   - 7.5. Fire t and update *m, and P_R_T* ;
   - 7.6. k = k +1 ;
   - 7.7. Estimate Data[k] ;
   - 7.8. Go to (5) ;
8. *Clock_time* = min { *E_L(k)* } ;
9. Update Data[k], go to (7) ;
10. Do backtracking, go to (6) ;
11. *Estimate cv for the current schedule* ;

12. If ($cv < ubv$) ;

12.1. $ubv = cv$;

12.2. Copy C_S to U_B_S ;

13. Else ;

13.1. Copy U_B_S to C_S;

14. Go to (2) ;

Backtracking-routine

1. Generate a RN ;

2. If ($RN \geq UCF$) ;

2.1. While ($SE\_set[k] = \phi$) ;

2.1.1. $k = k$ -1 ;

2.2. If ($TE\_set[k] = \phi$) ;

2.2.1. $Clock\_time$ = min $\{E\_L(k)\}$;

2.2.2. Update Data[k] ;

2.3. Else;

2.3.1. $Clock\_time = Clock\_time[k]$;

2.4. $m = m[k]$;

2.5. $P\_R\_T = P\_R\_T[k]$;

3. Else;

3.1. While ($TE\_set[k] = \phi$);

3.1.1. $k = k$-1;

3.2. *Clock_time* = *Clock_time*[k] ;

3.3. *m* = *m[k]* ;

3.4. *P_R_T* = *P_R_T[k]* ;

APPENDIX 2

EXPERIMENTAL DATA SETS

Data set 1:

Consider a flexible manufacturing system, which consists of 4 work centers with the following specifications

- Number of similar machines within each work center is represented by a uniform distribution of interval (1,2);
- Each work center has an input buffer;
- The capacity of each buffer is represented by a uniform distribution of interval (1,2);
- Number of part types produced is represented by a uniform distribution of interval (varies according to the generated test cases: 1-50, 50-100, 100-150, 150-200);
- The batch size of each part type equals to 1;
- Number of required operations for each part type is represented by a uniform distribution of interval (1,3);
- Number of sequential processes for each part type is represented by a uniform distribution of interval (1,3);
- Processing time of each operation is represented by a uniform distribution of interval (50.0,200.0);

Data set 2:

Consider a flexible manufacturing system. which consists of 4 work centers with the following specifications.

- The number of machines within each work center is represented by a Uniform distribution of interval (1-3).
- Each work center has a central buffer with Uniform capacity of (1-3).
- Number of operations of each part type is represented by a uniform distribution of interval (2-4);
- Number of sequences of each part type is represented by a uniform distribution of interval (1-3);
- Processing time of each operation is represented by a uniform distribution of interval (50.0.200.0);
- Batch size of every part type is one
- Four intervals of uniform distribution have been used to generate the number of part types of each generated test case. In each interval. five test cases have been generated. Table 1 shows the average values of some parameters of the generated test cases.

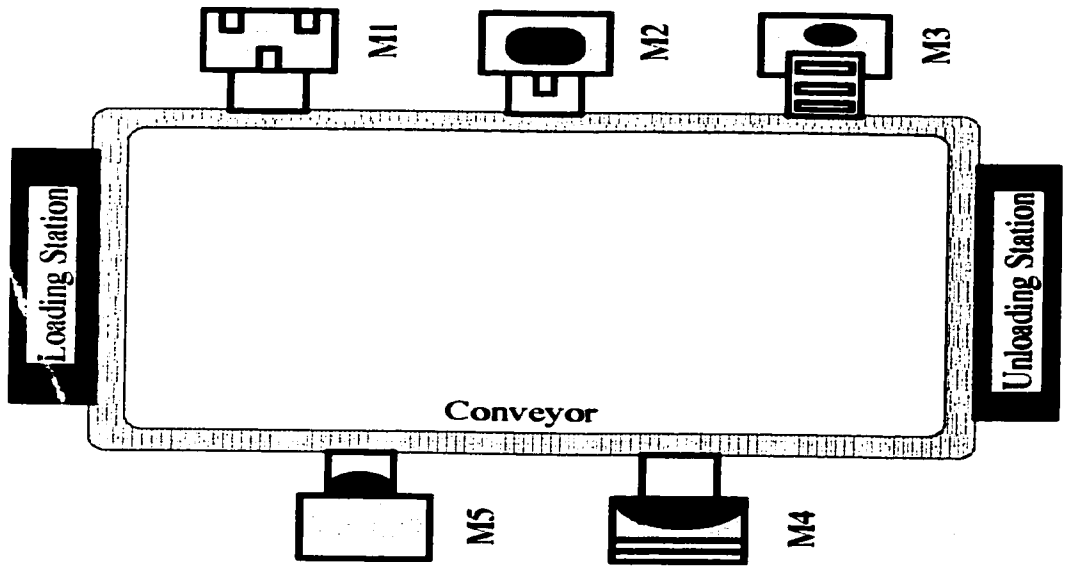
Table 1: Average parameters values of the randomly generated test cases.

Number of part types	Avg. number of places	Avg. number of transitions	Avg. number of operations
Uniform (1-50)	193	168	111
Uniform (50-100)	369	325	214
Uniform (100-150)	680	611	397
Uniform (150-200)	864	773	514

Data set 3:

The Figure below shows a schematic layout of a hypothetical flexible manufacturing system. The number of operations of each part type is represented by a uniform distribution of interval (2.6). The system has loading and unloading stations. There are no intermediate buffers in the system. The following assumptions were used in the experiments:

1. Every machine can process only one operation at a time;
2. The machines can load and unload themselves;
3. Setup times are sequence independent and included in the processing time;
4. Intermachine transportation times are negligible;
5. Pre-emption is not allowed;



Schematic layout of the hypothetical flexible manufacturing system

APPENDIX 3

DETAILED RESULTS OF ANOVA OF CHAPTER 5

******Analysis of Variance--design 1******					
Tests of Significance for AFT using UNIQUE sums of squares					
Source of Variation	SS	DF	MS	F	Sig of F
WITHIN+RESIDUAL	4454053.52	288	15465.46		
RF	8166358.46	3	2722120	176.01	0.000
MR	321540.57	2	160770.3	10.40	0.000
SL	15596846.95	2	7798424	504.25	0.000
PTV	8647481.07	1	8647481	559.15	0.000
RF BY MR	73563.64	6	12260.61	0.79	0.576
RF BY SL	1501108.99	6	250184.8	16.18	0.000
RF BY PTV	1458912.35	3	486304.1	31.44	0.000
MR BY SL	59046.57	4	14761.64	0.95	0.433
MR BY PTV	25323.78	2	12661.89	0.82	0.442
SL BY PTV	1464925.08	2	732462.5	47.36	0.000
RF BY MR BY SL	30131.17	12	2510.93	16.00	0.999
RF BY MR BY PTV	12668.75	6	2111.46	14.00	0.991
RF BY SL BY PTV	305838.69	6	50973.12	3.30	0.004
MR BY SL BY PTV	3603.80	4	900.95	0.06	0.994
RF BY MR BY SL BY PTV	5547.27	12	462.27	0.03	1.000
(Model)	37672897.13	71	530604.2	34.31	0.000
(Total)	42126950.65	359	117345.3		

VITA AUCTORIS

NAME: Tarek Younis ElMekkawy

PLACE OF BIRTH: Giza, Egypt

YEAR OF BIRTH: 1966

EDUCATION: Cairo University, Giza, Egypt

1985-1990 B.Sc.

Cairo University, Giza, Egypt

1990-1994 M.Sc.

University of Windsor, Windsor, Ontario, Canada

1996-2001 Ph.D.