AN ANALYSIS OF FMS SCHEDULING PROBLEM: A BEAM SEARCH BASED ALGORITHM AND COMPARISON OF SCHEDULING SCHEMES

A THESIS

SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL

ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCES

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Süleyman Karabük

September, 1994

AN ANALYSIS OF FMS SCHEDULING PROBLEM: A BEAM SEARCH BASED ALGORITHM AND COMPARISON OF SCHEDULING SCHEMES

A THESIS

SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCES

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Suleyman Karabuk

September, 1994

TS
1575
.K37
1994

B001382

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

[illegible]

Assist. Prof. Ihsan Sabuncuoglu (Principle Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

[illegible]

Assoc. Prof. Omer Benli

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

[illegible]

Asisst. Prof. M. Selim Akturk

Approved for the Institute of Engineering and Sciences:

[illegible]

Prof. Mehmet Baray
Director of Institute of Engineering and Sciences

ABSTRACT

AN ANALYSIS OF FMS SCHEDULING PROBLEM: A BEAM SEARCH BASED ALGORITHM AND COMPARISON OF SCHEDULING SCHEMES

Suleyman Karabuk
M.S. in Industrial Engineering
Supervisor: Assist. Prof. Ihsan Sabuncuoglu
September, 1994

FMS scheduling procedures in the literature can be classified into on-line and off-line schemes according to the number of scheduling decisions made at a point in time. On-line scheduling attempts to schedule operations one at a time when it is needed and off-line scheduling refers to scheduling operations of available jobs for the entire scheduling period. In the literature there is no unified argument for or against either of these scheduling schemes. This research has two main objectives: development of a new scheduling scheme called quasi on-line that makes a trade-off between on-line and off-line schemes and comparison of the proposed scheme with others under various experimental conditions. A new algorithm is proposed on which the quasi on-line scheme is based. The proposed algorithm is a heuristic and utilizes a beam search technique. It considers finite buffer capacity, routing and sequence flexibilities and generates machine and AGV schedules for a given scheduling period. A simulation model is also developed to implement and test scheduling schemes.

Keywords: Flexible Manufacturing Systems, scheduling, simulation.

ÖZET

ESNEK ÜRETİM SİSTEMLERİNDE ÇİZELGELEME PROBLEMİNİN BİR ANALİZİ: IŞIN ARAMA TABANLI BİR ALGORİTMA VE ÇİZELGELEME METODLARININ KARŞILAŞTIRILMASI

Süleyman Karabük
Endüstri Mühendisliği Bölümü Yüksek Lisans
Tez Yöneticisi: Yrd. Doç. İhsan Sabuncuoğlu
Eylül, 1994

Literatürdeki Esnek Üretim Sistemleri (EÜS) çizelgeleme yaklaşımları her bir çizelgeleme noktasında verilen karar sayısına göre anında yönlendirme ve önceden çizelgeleme olmak üzere iki kategoriye ayrılabilir. Önceden çizelgeleme yaklasımı çizelgeleme kararlarının gerektiği zaman ve tek tek yapılmasini gerektirir. Öte yandan, önceden çizelgeleme bütün çizelgeleme kararlarının bir kerede alınmasını öngörür. Literatürde hangi yaklaşımın üstün olduğu konusunda bir fikir birliği yoktur. Bu araştırmanın iki ana amacı vardır. Birincisi, anında yönlendirme ve önceden çizelgeleme yaklaşımlarının arasında olan ve her ikisinin olumlu taraflarını birleştiren yeni bir yaklaşım önermektir. İkincisi ise, önerilen yaklaşım ile diğerlerini değisik işletim çevrelerinde karşılaştırmasını yapmaktır. Bunun için, makinaların kısıtlı kuyruk kapasitesini, rota ve sıralama esnekliklerini gözönüne alarak makina ve otomatik güdümlü malzeme taşıtlarını çizelgeleyen bir algoritma geliştirilmiştir. Ayrıca değişik çizelgeleme yaklaşımlarının denenmesi için bir benzetim modeli de geliştirilmiştir.

Anahtar sözcükler: Esnek Üretim Sistemleri, çizelgeleme, benzetim.

ACKNOWLEDGMENTS

I would like to thank my advisor Professor Sabuncuoglu for his supervision, guidance, understanding and encouragement throughout my graduate education in Bilkent University. I am also indebted to Professor Benli and Professor Akturk for their valuable comments on this thesis.

I greatly appreciate Bilkent University administration for providing advanced computing facilities and a pleasant research environment.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

A flexible manufacturing system(FMS) can be defined as a group of processing stations connected by means of an automated material handling and storage system, and controlled by an integrated computer system (Groover [11].) As the definition implies, the main components of an FMS are processing stations (mostly NC machines with different types of machining tools), material handling system and a computer control system which coordinates the activities of processing stations and material handling system. These systems are highly automated and capable of producing a variety of part types simultaneously. The flexibility of an FMS is mainly due to the capability of processing stations which can perform several different types of operations, and its material handling system which provides fast and flexible part transfer within the system. The aim of FMSs is to fill the gap between high production volume transfer lines with low product variety and low production volume NC machines with high product variety. These systems provide a number of benefits in terms of higher machine utilizations, reduced work-in-progress, lower manufacturing lead times, greater flexibility in production scheduling and higher labor productivity (Groover [11].) However, the benefits of FMSs are not easy to realize. The management of these systems require the solution of several optimization problems faced at different stages of an FMS life cycle. These problems can be hierarchically decomposed into design, planning, scheduling and control problems (Stecke [38].)

Design problems are concerned with the physical shaping of the system subject to budget constraints and system goals. Specifically, determination of part types to be produced, process plan of part types, flexibilities and their levels, capacity of material handling system and buffers, number of pallets and fixtures, are among the design problems. FMS planning problems include decisions that have to be made before the system begins to produce parts. These decisions are also closely related to scheduling decisions. Determination of part types for immediate and simultaneous production, determination of production ratios, allocation of machines to groups and tools to machines constitute the planning problems.

The scheduling problem can be defined as the detailed minute by minute scheduling of machines, material handling system components, and other support equipment. Given the shop conditions and a set of parts with known or estimated processing requirements, it is concerned with scheduling actual job release times, determining the start and completion times of each operation on a wide variety of resources. Whereas, the control problem is concerned with monitoring the execution of the schedule and providing corrective actions in response to various changes in a manufacturing environment.

FMS scheduling problems and the proposed solution approaches can best be described by the help of a classification framework proposed by Hutchison [14]. In this study, the author decomposes the FMS scheduling research into two dimensions (Table 1.1). These are system factors that make up the scheduling problem and scheduling scheme factors that define the characteristics of a specific scheduling procedure.

According to the system factors, the number of part types simultaneously produced by the system determine the operational mode of an FMS. This factor ranges from dedicated FMSs with large set of part types and moderate demand to random FMSs with small set of part types and low demand. The flow pattern can be either jumbled as in a jobshop or each job may have a fixed processing sequence as in a flow shop. Demand pattern determines the scheduling environment. In a periodic demand environment order status changes periodically (e.g. every week), whereas new orders arrive dynamically over time in continuous demand environment. Another common clasification with respect to demand pattern is static and dynamic environments. In a static environment, the scheduling problem is defined with respect

System Factors	
Number of part types	• Dedicated • Intermediate • Random
Predominant flow pattern	• Jobshop • Flow shop
Demand pattern	• Periodic (static) • Continuous (dynamic)
Scheduling scheme factors	
Scheduling problems addressed	• Input sequencing • Detailed scheduling
Number of decisions made at a point in time	• On-line • Off-line
Characteristics considered	• Machine breakdowns • Material handling capacity • Tool magazine capacity • Pallet/fixture capacity • In-system storage capacity • Routing flexibility

Table 1.1 Classification of FMS scheduling (Hutchison [14]).

to a finite set of completely specified requirements; no additional requirements will be added to this set. Whereas in a dynamic environment the scheduling problem is defined not only for the known requirements but also with respect to the expectations for additional requirements and specifications generated over the planning horizon. The first issue in the scheduling scheme factors is the scheduling problem addressed. The scheduling problem is concerned with both the actual release times of jobs to the system (input sequencing) and determination of start and completion of jobs on system resources (detailed scheduling). Most of the existing studies give more emphasis on detailed scheduling activities.

The scheduling scheme factor is further classified in terms of the number of decisions made at scheduling points. A scheduling point is a point in time t when scheduling decision(s) is made. With an on-line scheduling scheme, scheduling decisions (i.e. start and completion times of jobs) are taken one at a time whenever needed. On the other hand, an off-line scheduling scheme generates a complete schedule of all jobs for the entire planning horizon. System characteristics that are considered in scheduling schemes are also listed in Table 1.1. This factor, in a way, determines the schedule generation algorithm used in a scheduling scheme. On-line scheduling schemes usually employ machine and Automated Guided Vehicle (AGV) scheduling scheduling rules in the decision making process, whereas exact or complicated heuristic schedule generation algorithms are used in off-line scheduling schemes.

There are two important aspects of on-line and off-line schedule generation schemes. One is the amount of information used at a scheduling point, other is the degree of responsiveness to changes in the environment. With an on-line scheme a scheduling decision is made in response to changes in the system state (e.g. machine finishes processing of a part or AGV completes delivery of a unit load etc.) Hence, individual decisions are delayed until the last moment. This requires making decisions one at a time frequently throughout the entire planning horizon. Since the scheduling process uses the most up to date information about the status of the system, on-line schemes have high degree of responsiveness to unexpected changes in the environment. However, only information about one small area of the system is utilized in the scheduling process and that leads to myopic decisions.

On the other hand, with an off-line scheme a complete schedule is generated for the entire planning horizon at one decision point. The entire condition of the shop is considered in the decision making process and that increases the quality of the scheduling decisions made. However, during execution of the off-line generated schedules various types of unexpected events can easily invalidate the fixed schedule. Therefore degree of responsiveness of an off-line scheme is low unless appropriate control strategies are adopted.

These two scheduling schemes represent two extreme points in terms of the extend of information usage in the scheduling process and the degree of responsiveness to changes in the environment. Yet, there is no unified argument for or against on-line or off-line scheduling schemes.

In this thesis we will study the scheduling problem in a random type FMS with jobshop type flow pattern under both periodic and continuous demand. The emphasis will be on detailed scheduling rather than input sequencing. Specifically, we will investigate two important issues that have not been addressed thoroughly in the literature. These are as follows:

1. In most of the studies that are concerned with comparison of on-line and off-line scheduling schemes, a deterministic and static manufacturing environment is used as a test bed. Only a few studies address the problem in a stochastic and dynamic environment. Moreover, these studies usually focus on development of an off-line algorithm and measuring its performance against scheduling rules in an environment where the levels of system characteristics are fixed (e.g. AGV load level, flexibility levels etc.) Hence, in order to accomplish a fair comparison between on-line and off-line schemes, various system characteristics must be taken into account in varying manufacturing environments.

2. As mentioned earlier, there are advantages and disadvantages of on-line and off-line scheduling schemes. In this thesis, a new scheduling scheme will also be proposed to make a trade-off between these two schemes.

In general, industrial scheduling problems are difficult to solve. The FMS scheduling problem is even more difficult due to the considerations of multiple resources and alternative processing steps and different material handling routes. The dynamic nature of the FMSs further complicates the problem. For these reasons, most of the proposed off-line scheduling algorithms are based on a number of simplifying assumptions in order to keep the computational burden at a reasonable level. As a result, some of the relevant and important features of FMSs are usually ignored in the existing work. For example, only machines are considered as the primary resource and other factors such as material handling system, flexibilities, finite buffer capacities are ignored.

In this thesis, a new scheduling algorithm is proposed. It is a heuristic based on the filtered beam search technique. It considers most of the scheduling factors listed in Table 1.1. Hence, it provides a tool to examine the effects of scheduling factors on the system performance. With this tool a fair comparison can be made between on-line and off-line schemes. Additionally, the algorithm can generate schedules for varying scheduling periods.

The algorithm uses a parameter called the *time window* which determines the scheduling horizon considered at one scheduling point. With an on-line scheme at each decision point only a specific point in time is considered, thus scheduling horizon is 0 at each decision point. On the other hand, with an off-line scheme the entire planning horizon is considered as the scheduling horizon at a scheduling point. Therefore, as the value of the time window parameter decreases (increases) the scheduling scheme which employs the proposed algorithm becomes closer to an on-line (off-line) scheduling scheme.

We call this new scheduling scheme as *quasi on-line*. Because it works like an on-line scheme except that several scheduling decisions (depending on the value of the time window parameter) are taken at a decision point. The role of this time window parameter is to adjust the degree of responsiveness of the scheme and the extend of information usage at scheduling points.

In this study, a simulation model is also developed to execute the schedules generated by different scheduling schemes in stochastic and dynamic manufacturing environments. The simulation model is linked with various scheduling algorithms to form a simulation based scheduling system. This system is composed of a simulation model, a controller and a scheduling module. The scheduling module contains several on-line scheduling algorithms as well as the scheduling algorithm developed in this research. With this system different scheduling schemes including on-line, off-line and the quasi on-line can be compared in different simulated environments, using different performance criteria.

The rest of the thesis is organized as follows: Chapter 2 contains a literature survey that provides supporting evidence for the observations made in this chapter. However, this chapter is not necessary to follow the rest of the manuscript, hence it can be skipped without loss of generality. In Chapter 3 the scheduling algorithm is described in detail. In Chapter 4, the simulation based scheduling system is described in detail and implementation issues are discussed. Chapter 5 presents experimental results obtained by running the simulation based scheduling system in various manufacturing environments. Finally, concluding remarks are made and future research directions are outlined in Chapter 6.

CHAPTER 2

LITERATURE REVIEW

The FMS scheduling problem has received ample attention from researchers of different disciplines. This is due to the fact that scheduling decisions effect the performance of an FMS significantly (Nof et al. [24]). The multidisciplinary nature of the topic prohibits an exhaustive search on the literature. Nevertheless, there are already some survey papers that cover most of the work done in this field. In this chapter the focus will be on the type of the scheduling scheme (i.e. on-line and off-line). The literature about some other topics will be covered when they are mentioned throughout the manuscript.

The rest of the chapter is organized as follows. Section 2.1. reviews recent survey papers about FMS scheduling and outlines general issues that are pointed out in these studies. In section 2.2., some recent studies about on-line and off-line scheduling schemes are examined in detail.

2.1. General Issues

Raman [29] shows the first attempt for a complete review of the existing literature on machine scheduling as it relates to flexible manufacturing systems. The main focus of his study is to document research on the development of a scheduling system for the Automated Manufacturing Research Facility (AMRF) at National Institute of Standards and Technology (NIST) of U.S.A.

Hutchison [14] proposes a classification framework (shown in Table 1.1) and reviews the FMS scheduling procedures in the context of this framework. Finally he makes the following conclusions:

- There is a trend toward building intermediate and random jobshop systems with periodic demand
- Off-line scheduling schemes seem most appropriate for the average system in the future
- More research is needed to examine the effects of breakdowns and delays on on-line and off-line scheduling schemes.

Most recently, Rachamadugu and Stecke [28] provide another classification and review of FMS scheduling procedures. The authors also discuss the differences between FMS scheduling and jobshop scheduling. In this study, the following features are found to be unique to FMSs from scheduling point of view.

- Alternative routing
- Buffer limitations
- Transportation time
- Transportation capacity
- Deterministic processing times
- Reduction of set-up between consecutive operations
- Pallet and fixture limitations.

They also point out that FMSs are more sensitive to machine breakdowns than jobshops due to the tighter synchronization, integration, and dependencies among the automated components. They argue that all these factors have to be considered when developing appropriate· FMS scheduling procedures. They make the following conclusions according to their observations.

- There is a lack of concern for due date related criteria and research should be directed towards developing scheduling procedures with the primary objective of meeting due dates while system utilization and minimizing in-process inventories as secondary objectives
- Limited buffer space aspect of an FMS is usually neglected in designing scheduling procedures. Consideration of finite buffers is important because of

the integration required of the system and the consequences of the potential blocking and locking

- Sequence flexibility which is inherent in the part type rather than the processing system is also an important feature and if properly exploited may have beneficial effects on various measures of system performance
- Although the limited transportation resources and transportation times that are comparable to the processing times can influence the overall system performance, these aspects have not been sufficiently considered in the literature.

2.2. On-line vs. off-line scheduling

2.2.1. On-line scheduling

On-line scheduling schemes usually employ dispatch rules in the decision making process. These rules are applied to select the next part (given a set of parts) which will take service when a resource becomes idle (e.g. machine, AGV, pallet etc.)

Since machine scheduling rules are widely used to solve jobshop scheduling problems, there is a wide base of literature available on these rules. Some survey papers about machine scheduling rules in a jobshop environment include, Panwalkar and Iskander [26], Blackstone et al. [2], and Kiran and Smith [17]. Montazari and Wassenhove [21] analyze the performance of machine scheduling rules in an FMS.

Scheduling rules for material handling transporters are first studied by Egbelu and Tanchoco [9]. They basically distinguish two types of AGV scheduling rules: workcenter initiated rules and vehicle initiated rules. The first type of rules are applied when a workcenter completes processing of a job and there are more than one idle AGVs that can satisfy the request. In such a situation, a move request is issued for the completed part and an idle vehicle is dispatched for the completed part according to a workcenter initiated scheduling rule. Whereas the second type of rules are applied when an AGV completes a delivery operation and there are more than one workcenters that issued a move request. In this case a vehicle initiated rule is used.

Sabuncuoglu and Hommertzheim [33] investigate the relative performance of machine and AGV scheduling rules against mean flow-time criterion. They test the scheduling rules utilizing a simulation model under varying machine and AGV load levels, different queue capacities and varying AGV speeds. They show that shortest processing time (SPT) (shortest distance (STD) and least queue size (LQS)) performs best among machine (AGV) scheduling rules with any AGV(machine) scheduling rule combination. They also point out that as the machine and/or AGV load increases, the differences in the performance of the scheduling rules become more significant. The same authors in a similar study [34] analyze machine and AGV scheduling rules for the tardiness criterion. They obtain different experimental conditions by changing distribution type and parameters for processing times, varying machine and AGV load levels, different queue capacities and AGV speeds and varying levels of due-date allowances. Their findings suggest that although none of the machine scheduling rules is the best under all conditions, modified operation due date (MOD) outperforms other rules under most of the experimental conditions. In this study, LQS is again found to be the best performing AGV scheduling rule under all of the conditions.

In another study, Sabuncuoglu and Hommertzheim [32] propose an on-line algorithm for scheduling machines and AGV in an FMS. Their algorithm uses more information than traditional machine and AGV scheduling rules. The information such as the current system load and the status of jobs in the system are utilized in a hierarchical structure so that different decision criteria are applied sequentially to identify the most appropriate scheduling decision. They compare the performance of the algorithm with that of several other machine and AGV dispatch rules by using mean flow time and tardiness criteria and show that the algorithm produces significant performance improvement over existing scheduling rules for all of the conditions tested.

Some studies are concerned with developing dispatch rules that can properly exploit routing flexibility. Yao and Pei [42] develop a quantitative measure to assess routing flexibility which incorporates all the job and machine characteristics that contribute to routing flexibility. They establish two dispatch rules which make use of this measure: part selection and machine selection rules. They compare these rules with the SPT rule in a simulation study and show that the proposed rules perform better than the SPT rule. However, material handling aspect of the problem is not considered in this study.

In a similar study, Chandra and Talavage [3] present a decision rule for dispatching parts which have alternative processing possibilities. At any decision point the rule employs information about shop congestion level, criticality of a part, preference of a part for a machine and current shop objective. They conduct a simulation study to test the performance of their rule against other dispatch rules and show that the proposed rule provides better results.

Mukhopadhyay et al. [23] describe a heuristic scheduling algorithm that take into account many system features. Essentially, this heuristic selects the next part to be processed by considering tool allocation, pallets scheduling, machine scheduling and material handling equipment scheduling. They formulate the problem as a hierarchical process and solve it by eigenvector analysis of priority ordering.

One approach to overcome the myopic nature of dispatch rules is to develop more complicated procedures that can utilize system wide information. Another approach is to develop on-line scheduling schemes that can select and apply different dispatch rules when the system operating characteristics change. Wu and Wysk [40] present such a scheduling approach. In their system, at the beginning of every fixed time period a set of dispatch rules are simulated for a short period of time and the best performing one is selected to be used for the next period. The authors also examine the effects of the period length. Experimental results show that a significant improvement can be obtained by this approach when compared to using a single dispatch rule for the entire scheduling horizon. Ishii and Talavage [16] further study the same approach and propose a transient based algorithm which selects a dispatch rules for variable time periods. This study indicates that a variable period length provides better results then a fixed length. In a more recent study, Shaw et al. [36] apply artificial intelligence techniques to capture the changes in the system operating characteristics. They use the following system attributes: number of machines in the system, total buffer size, variability in machine workload, overall system utilization, flow allowance factor which measures due date tightness and routing flexibility. Their proposed method performs well when system characteristics do not change frequently.

2.2.2. Off-line scheduling

Research on development of scheduling algorithms that generate off-line schedules has not been as intensive as it is with on-line scheduling algorithms. This is mostly due to the fact that the heavy computational requirements of off-line schemes prevent their usage in a real time scheduling environment. Due to the difficulty of the FMS scheduling problem, a wide variety of modeling and solution techniques are used in the literature, ranging from optimization algorithms, artificial intelligence methods to heuristics methods and simulation techniques. The following studies provide a representative collection of scheduling schemes that employ off-line scheduling algorithms in the FMS scheduling literature.

Chang et al. [4] propose a two phase heuristic off-line algorithm for FMS scheduling in a dynamic environment. According to their scheduling scheme, at each job arrival the algorithm reschedules all the available jobs. Because of this, they call their scheduling scheme as quasi real time. The algorithm consists of two phases: in the first phase a reduced enumeration algorithm is used to generate several feasible schedules for each job and in the second phase an integer programming model is solved to select schedules for each job so as to optimize a pre specified criteria. The optimization procedure uses a branch and filter algorithm that takes advantage of the special problem structure. They compare the proposed algorithm against six machine scheduling rules. They conduct a simulation study in a deterministic environment with an example FMS in which machines, a transfer line and pallets are explicitly modeled. Their scheduling algorithm generates schedules for the machines only. The performance measure is mean flow time. Computational tests indicate that the quasi real time scheme performs better than the dispatch rules. Specifically, it provides 8% lower mean flow time than the best performing rule. Another result of this study is that, the least work remaining (LWRK) rule performed better than others. It is quite surprising that the SPT rule did not perform well in dynamic FMS environment.

Yamamoto and Nof [41] investigates rescheduling policy in a static environment where frequent machine breakdowns occur. The off-line scheduling algorithm used in their study is adopted from a jobshop schedule generation algorithm, which is based on active schedule generation. Thus, as in the previous study it generates only machine schedules. According to the rescheduling policy, at each machine breakdown all the operations which are not yet processed are used to generate a complete schedule. They compare this policy with an on-line scheme which utilizes the first come first served (FCFS) rule. The performance measure is minimization of makespan. Also, most total work (MTWK) rule is used to release jobs to shop floor when a pallet becomes available. Two example systems which consists of machining centers and a conveyor loop are used in their study. They show that, schedules generated by the off-line algorithm provides better solutions than with that of the scheduling rules even when rescheduling policy is not active. Specifically, the rescheduling policy provides about 7% improvement over dispatch rules and 2.5% over fixed sequencing policy.

Sriskandarajah et al. [37] develop scheduling algorithms for a class of flexible manufacturing systems consisting of machining centers with no local buffer area and served by a conveyor loop. The scheduling problem in their system reduces to finding a job order in the conveyor. Two scheduling algorithms, one of which produces optimal solutions for a specific system configuration are proposed by the authors.

They compare their heuristic with random schedules in a static and deterministic environment and report promising results in favor of the heuristic.

Chang et al. [5] present another off-line scheduling algorithm that is based on a bottleneck based beam search. The proposed algorithm generates machine schedules and considers only routing flexibility during schedule generation. Other scheduling factors are ignored. The algorithm first constructs a search tree then applies beam search technique to find a good solution. The crucial part of the algorithm is the generation of the search tree. Each node of the search tree represents a partial schedule. The next layer of nodes which correspond to the operations that are immediately schedulable are determined as follows. The operations which are not included in the partial schedule are used to produce a complete schedule using the SPT rule. Then the critical path of the derived schedule is identified. This is the collection of operations that form the longest path over which the precedence constraints are active. Finally, the operations which can replace and finish earlier than the first operation on the critical path are added to the next layer of nodes. This procedure is very similar to PERT/CPM analysis because the processing times of operations on the critical path are reduced in order to reduce the makespan. In the proposed algorithm this is accomplished by making use of routing flexibility. The authors also propose a flexibility index to measure the routing flexibility quantitatively. In this study they also measure the performance of the algorithm in a static and deterministic environment by comparing it with several dispatch rules. Their experiments indicate that the effects of routing flexibility on both the off-line scheduling algorithm and scheduling rules are significant. Also, the algorithm outperforms the dispatch rules and exploits the routing flexibility better.

Raman et al. [30] describe an axact algorithm and examine its performance in a dynamic and deterministic environment. In their approach, at each job arrival a static problem is generated and solved by the off-line algorithm. Then the resulting schedule is implemented on a rolling horizon basis. The algorithm generates schedules for machines and material handling transporter simultaneously. They formulate the problem as an integer programming model in which demand for transportation is treated as a simple move request between two machining operations. Hence, transporters are assumed to turn back to load/unload station. Also, the buffer space at the machines is assumed to be uncapacitated. Moreover, it is assumed that any machining operation does not begin until the transporter returns to the load/unload station. They conduct simulation studies to evaluate the performance of the off-line scheme under balanced and unbalanced workload of machines with the objective of minimizing mean tardiness. However, the experimental results suggest that the off-line scheme produce similar results with that of dispatch rules. The authors attribute this to the low utilization level (e.g. 20%) achieved in the experiments which is deliberately set to keep computation time at a reasonable level.

The off-line scheduling algorithms developed by De [7] and De and Lee [8] are two examples for AI based studies. Both algorithms generate schedules for machines considering routing flexibility and transportation time. In the first study, the author represents the solution space with the state operator framework that is based on first order predicate calculus and a conflict resolution strategy is also used. The computational requirements of the algorithm is rather high. Consequently, only a simple example is solved to demonstrate the algorithm. In the second study a frame based knowledge representation scheme is used to represent the solution space. The filtered beam search technique is also applied to search for a good solution. A comparative study has not been done to see the performance of the algorithm.

Hutchison et al. [15] develop two exact scheduling algorithms for a random type FMS with jobshop type flow pattern and operates within a static and deterministic environment. The first algorithm is based on a mixed-integer zero-one programming model and finds optimal solution for the routing and scheduling problem simultaneously. The second algorithm decomposes the routing and scheduling problems into two subproblems. First, a routing problem is solved and then using this solution as input the scheduling problem is solved in sequel. This approach simplifies the original problem and reduces the computational efforts considerably. Both algorithms utilize a branch and bound technique. In this study they examine:

- the effects of using a procedure that decomposes the scheduling problem
- the effects of routing flexibility on scheduling schemes
- the appropriateness of on-line versus off-line scheduling for a random, jobshop FMS in a static environment.

They conduct experiments using an example FMS with seven machines. All other subsystems (e.g. material handling etc.) are assumed to have ample capacity. They compare two off-line schemes which employ the optimal-seeking algorithms and an on-line scheme which uses the SPT rule with a look-ahead control policy, under different levels of routing flexibility. The computational results reveal that both of the off-line schemes perform much better than the on-line scheme. In addition, the off-line schemes take advantage of increased routing flexibility more so than the on-line scheme does. They also observe that the decomposed off-line scheme performs very close to the optimal off-line scheme.

In another study, Aanen et al. [1] examine the scheduling problem of an FMS with a particular configuration. The FMS basically consists of two machines which can process a wide range of different jobs and each job consists of one or more processing operations on one or both machines. The problem is solved for sequence dependent setup times and constant transfer times that occur on both machines and between the machines. With these complexities the problem is handled within the context of general jobshop scheduling problem. The authors develop a branch and bound algorithm that take advantage of the special structure of the problem. They also use a method which limits the number of nodes investigated in order to reduce computational requirements. They test the algorithm using different bounding procedures and compared it with dispatch rules. They show that the algorithm with the best performing bounding procedure yields an improvement of 6.8% over the dispatch rules for makespan criterion. Although the run time of the algorithm is high, the authors conclude that this may be reduced by using faster machines and optimizing the computer code of the algorithm.

Ulusoy and Bilge [39] address the problem of scheduling machines and automated guided vehicles simultaneously in an FMS. They decompose the problem into two subproblems: machine scheduling and AGV scheduling. They develop an iterative procedure which essentially solves the machine scheduling problem first and then finds a feasible vehicle schedule that fits it. At each iteration, a new machine schedule is generated and investigated for its feasibility to the vehicle scheduling subproblem. The operation completion times obtained from the machine schedule are used to construct time windows for each material handling trip and the second subproblem is handled as a sliding time window problem. They also develop a single pass heuristic procedure so as to provide a basis for comparison. They conduct a set of experiments in a static and deterministic environment and examine the impact of processing times/travel times ratio on the performance of the procedure. The results suggest that the iterative procedure performs particularly well at high processing time/travel time ratio.

The following observations can be made from this brief literature review:

- Generally, off-line algorithms produce better solutions than on-line algorithms under static and deterministic environment. Their relative performance under more realistic environments (i.e. dynamic and stochastic) are not known and hence, open to further research.
- The computational requirements of off-line algorithms are usually much higher than on-line scheduling algorithms. In order to use off-line algorithms in real time scheduling more simplifying assumptions are made to reduce their computational burden. In the absence of considerations of many relevant features of the system, it is hard to examine the effects of scheduling factors on on-line and off-line scheduling schemes.

CHAPTER 3

THE SCHEDULE GENERATION ALGORITHM

In this chapter the proposed scheduling algorithm is described in detail and its performance is analyzed. The scheduling factors considered by the algorithm are machines, AGVs, buffer capacities, and flexibilities (routing and sequence). A deadlock avoidance and resolution mechanism is also embedded in the proposed algorithm. The algorithm can also generate partial schedules in varying time windows. However, in this chapter the algorithm will be used in off-line mode (i.e. complete schedules will be generated at one time for entire scheduling horizon) and its performance will be compared with that of machine and AGV scheduling rules. An analysis of the effects of scheduling factors on the system performance is also provided.

The rest of the chapter is organized as follows: in section 3.1. the algorithm is described in detail and its properties are discussed. This is followed by a discussion on system considerations and experimental conditions in section 3.2. Finally, section 3.3. gives computational results.

3.1. Description of the algorithm

There are a number of solution approaches for the FMS scheduling problem in the literature. These can be simply classified into: mathematical modeling techniques with application of exact solution methods, and heuristic procedures. The former approach involves formulating the problem as a mathematical model and solving it using an exact algorithm (e.g. Hutchinson et al. [15], Sriskandarajah et al. [37]). Successful applications of this approach is limited with size of the problem and types of simplifying assumptions to be made. Unfortunately, inherent intractability of FMS scheduling problems make heuristic procedures attractive alternatives. These heuristic methods range from simple rules to more sophisticated algorithms. The scheduling algorithm proposed in this paper is a heuristic. According to the heuristic classification framework proposed by Zanakis et. al. [43], it is a construction type heuristic. That is, a solution is generated by adding individual components (e.g., nodes, variables, arcs) one at a time to a partial solution until a feasible solution is obtained. In the proposed algorithm, a decision tree is first constructed and then heuristic methods explore this search tree for the best solution. Hence, the algorithm is implemented in two consecutive stages: 1) decision tree representation to define a solution space and 2) application of a search methodology to find a good solution. These two steps of the algorithm are discussed in detail in the following sections.

3.1.1. Representation Scheme

The solution space is represented as a decision tree where each node corresponds to a scheduling decision to be made and each unique path from the root node to any particular node defines a partial solution associated with that node. Leaf nodes at the end of the tree specify complete solutions. In the proposed method, the search tree is constructed in such a way that various system resources, their capacities and flexibilities are taken into account at each layer. This means that availability of machines, AGVs, buffer spaces and flexibilities of the jobs are considered when next layer of nodes are sprouted from a parent node at each decision point. The detailed structure of this sprouting procedure is given below with illustrative examples. The following notation is used in the algorithm:

Notation and definitions:

i	subscript of jobs
j	subscript of operations
m	subscript of machines
g	subscript of AGVs
G	set of AGVs
$d_{i,j,m,g}$	time at which AGV g delivers job i to machine m for jth operation (delivery time)
$p_{i,m,g}$	time at which AGV g loads job i from machine m (pick up time)
$s'_{i,j,m}$	earliest possible start time of jth operation of job i on machine m, assuming instantaneous delivery
$s_{i,j,m,g}$	earliest possible start time of operation j of job i on machine m if the job is transported by AGV g
$f_{i,j,m,g}$	earliest possible finish time of operation j of job i on machine m if the job is transported by AGV g
K	a partial schedule
U(K)	a set of immediately schedulable operations for a given partial schedule K, $U(K)=\{n \mid n=(i,j,m,s')\}$ where each element n is defined by machine m to start processing jth operation of job i at time s'
W(K)	a set of scheduling decisions for a given K, $W(K)=\{n \mid n=(i,j,m,g,p,d,s,s',f)\}$, where each element n corresponds to scheduling of AGV g to pick up job i at time p, deliver it to machine m at time d, and scheduling of machine m to start processing jth operation of job i at time s and finish it at time f

Sprouting Algorithm

Step 1. Given a partial schedule K, determine the elements of U(K) by considering routing and sequence flexibilities, and buffer space availabilities.

Step 2. Construct elements of W(K).

Step 2.1. For each combination of $n \in U(K)$ and $g \in G$, compute (p,d,s,f) values of W(K).

Step 2.2. Compute $d^* = \min\{d_{i,j,m,g}\}$ over the elements of W(K). Delete the elements with $p_{i,m,g} > d^*$.

Step 2.3. Group the elements of W(K) according to the same (i,j,m) values. For each group, keep the element which satisfies $\min\{d_{i,j,m,g} - s'_{i,j,m}, 0\}$ and delete others. Break ties in favor of the one with the least $p_{i,m,g}$ values.

Step 3. Group the elements of W(K) according to the same i value. For each group, keep the one with the smallest $f_{i,j,m,g}$ and delete others. Break ties arbitrarily.

Step 4. Compute earliest finish time, $f^* = \min\{f_{i,j,m,g}\}$, over the elements of W(K). Delete the elements with $s_{i,j,m,g} > f^*$.

The first step of the sprouting algorithm integrates routing and sequencing decisions. The route of a job is formed progressively by taking into account the current state of the solution (i.e. partial schedule). In Step 2 of the algorithm, an AGV is schedulable immediately after delivering of its previously assigned load and never waits idle for a transportation request. A job which is assigned to an AGV can be in the output queue or still be in process. If it is in the process, the AGV goes to the respective machine and waits until the job is placed in output queue. Hence, the waiting time component associated with material handling system is reduced considerably. Step 2.2 ensures that all AGV schedules are active. This means that an AGV cannot meet transportation requirements of other jobs without violating the feasibility of the AGV schedule. In the proposed algorithm, machines and AGVs are simultaneously scheduled. This provides an opportunity for a job to shift a part of its waiting time in the input queue of the destination machine to its material handling time. Hence, the machine waiting time component of the job is also reduced. Moreover, choosing the AGV with the $\min\{d_{i,j,m,g}-s'_{i,j,m},0\}$ value avoids long job waiting times of the machine.

Step 3 of the algorithm provides a mechanism for effective use of flexibilities. An active schedule generation scheme proposed by Chang and Sullivan [6] is used in Step 4 so that none of the operations of the jobs can be started earlier without violating the feasibility of the schedule (i.e. remaining elements of W(K) form active machine schedules).

3.1.2. Search Methodology

After representing the solution space, an appropriate search procedure is used to find good solutions within this space. In the proposed scheduling algorithm, filtered beam search is used as the search methodology. Beam search is a fast and approximate branch and bound (B&B) method which operates on a search tree. It uses heuristics to estimate a certain number of best paths, permanently pruning the rest. Since large parts of the search tree is pruned off aggressively to obtain solutions quickly, its running time is polynomial in the size of the problem.

Beam search was first used in artificial intelligence by Lowerre [19] for a speech recognition problem. It was also applied to several scheduling problems. For example, Fox [10] used this technique as a part of ISIS system for solving real-life jobshop scheduling problems. Ow and Morton [25] investigated the performance of beam search with other heuristic procedures. They also proposed a variation of this technique called filtered beam search. In another study, Chang et al. [5] used beam search as a part of their FMS scheduling algorithm which is called bottleneck based beam search. More recently, De and Lee [8] used filtered beam search as a part of AI based scheduling procedure. An overwiev of beam search applications for the scheduling problems can be found in Morton and Pentico [22].

In filtered beam search, only a certain number of nodes (filterwidth) are sprouted, others are filtered out using a local evaluation function. These remaining nodes are then evaluated by a global evaluation function and the ones found most promising are added to the partial solution. This procedure is repeated on a certain number of parallel paths (beamwidth). Hence, the number of solutions saved at any level of the tree is equal to size of the beamwidth. Figure 3.1 illustrates an example of beam search. In the figure, shaded circles are the nodes on the solution path and dashed circles represent the nodes that are filtered out, whereas solid circles represent nodes left after filtering. Since the performance of beam search depends on the quality of these functions (i.e. local and global evaluation functions) and the parameters (i.e. beamwidth and filterwidth), they need to be specified carefully in order to adopt it to a particular

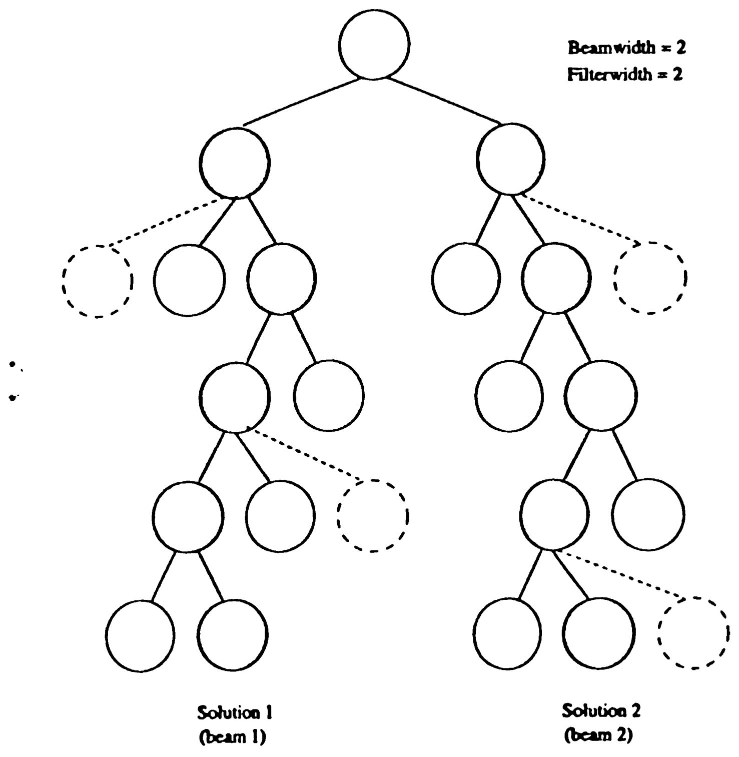


Figure 3.1. An example of filtered beam search.

solution space representation. The values of filterwidth and beamwidth are usually determined empirically. In most of the cases an iterative procedure is used by increasing parameter values until the point beyond which neither the filterwidth nor the beamwidth adds to the value of the solution, but to computation time. In our study, we used the filterwidth of 5 and the beamwidth of 3 as suggested by our pilot experiments.

The global evaluation function is a probe search beginning from the argument node and returns an upper bound value for the solutions that can be generated if that node is added to the solution. Whereas, local evaluation function uses only information inherent in a node, hence, is local to that node. In the proposed algorithm, the global evaluation function produces a tentative schedule by successively sprouting next layer of nodes and selecting one by local evaluation function to add to the tentative schedule. How far the global evaluation function goes in the search tree can be kept as a parameter. The further it extends the partial schedule, the better is the consequences of adding the argument node to the permanent schedule of the beam being investigated. Therefore, the probe length of the global evaluation function can control both the amount of information used by the algorithm and the computation time. This is adjusted by a parameter we call a time window. The global evaluation function produces a tentative schedule until no more nodes can be sprouted because of the constraint imposed by the parameter.

This also brings another design issue within the context of the proposed off-line algorithm, that is how to compare given partial schedules for a given performance criteria. In general, all of the performance measures that can be used for off-line schedules are based on complete schedules. Therefore, new measures are needed to evaluate the performance of partial schedules. In the makespan case, partial schedules are evaluated according to the average utilization level. In the flowtime case, the average waiting time per scheduled operations is employed. For the mean tardiness case, operation due dates are used to compute mean tardiness performance of the partial schedule. These measures are equivalent to their counterparts (i.e. performance measures) which are defined for complete schedules in terms of ranking a given set of schedules.

In our implementation of beam search, the local evaluation function is used for both filtering and selecting nodes in the global evaluation function. Since the size of the solution tree is huge in our case due to considerations of multiple resources, various flexibility types, a local evaluation function to be used in the proposed algorithm must be computationally very cheap. In the scheduling literature, the most popular approach for making quick and local decisions is to use scheduling (or dispatching) rules. As reported by Sabuncuoglu and Hommertzheim ([33], [34]), there are several machine and AGV rules used for FMS scheduling. In this study, some of these rules were used as the local evaluation functions. During rule selection process, a few subtle points have been noted that are worthwhile discussing here. The first point is that the selected rule should control job releases into the system. In our experiments, it was observed that the rules like SPT and MOD result in early job releases which eventually cause congestion in the system. The second point is that the relative urgency of jobs should not change frequently during the scheduling process. Otherwise, a job which is scheduled first on a current machine can wait for a long time on another machine and loose the advantage gained in the previous operation. For that reason, some job based rules are used in the evaluation functions rather than operation based rules (Table 3.1). Note that all these discussion are valid with reference to the proposed scheduling algorithm and experimental conditions used in this study.

Performance measure	Rule
Makespan	MTWK (most total work)
Mean flowtime	LWKR (least work remaining)
Mean tardiness	MDD (modified due date)

Table 3.1. Scheduling rules used as local evaluation functions

3.1.3. System Blocking

In the proposed algorithm, the schedule is constructed progressively by sprouting a layer of nodes and adding the most promising node to the partial schedule. However, the algorithm may not sprout next layer of nodes if the system is blocked in the partial schedule constructed so far. For example, when parts cannot go to the next machine on their route due to unavailable buffer space, job movement in the system can be blocked. These events in succession cause a deadlock in the system and job flow cannot be retained unless the jobs that cause deadlock are moved.

In the proposed algorithm, blocking problems are solved in two stages. First, a preventive action is taken at the global evaluation function level. This is accomplished as follows. If a next layer of nodes cannot be sprouted in the process of constructing a tentative schedule because of a deadlock situation, the global evaluation function does not return the performance measure value of the partial schedule, but returns the number of nodes added to the tentative schedule so far. When selecting a node from globally evaluated nodes, two subgroups are formed. One group contains nodes through which schedules can be obtained without any deadlock and the second group contains nodes that led to a deadlock situation in their global evaluation. If the first group is not empty, the node with the highest evaluation value is selected (Figure 3.2). Otherwise, the one which has probed farthest is selected (Figure 3.3). In this way, the potential paths which may lead to deadlock is avoided as much as possible. However, deadlocks can still occur, especially when global evaluation function does not probe far enough to detect possible deadlocks. In this case, only step 1 of the sprouting algorithm is changed as follows:

Exception Step: Consider the jobs which reside in output queue of the machine with no buffer space left. In the processing requirements of these jobs, insert a dummy operation on the L/U station with zero processing time.

The exception step ensures that every job which has a potential of resolving the deadlock becomes a candidate to be transferred to the L/U station and releases the

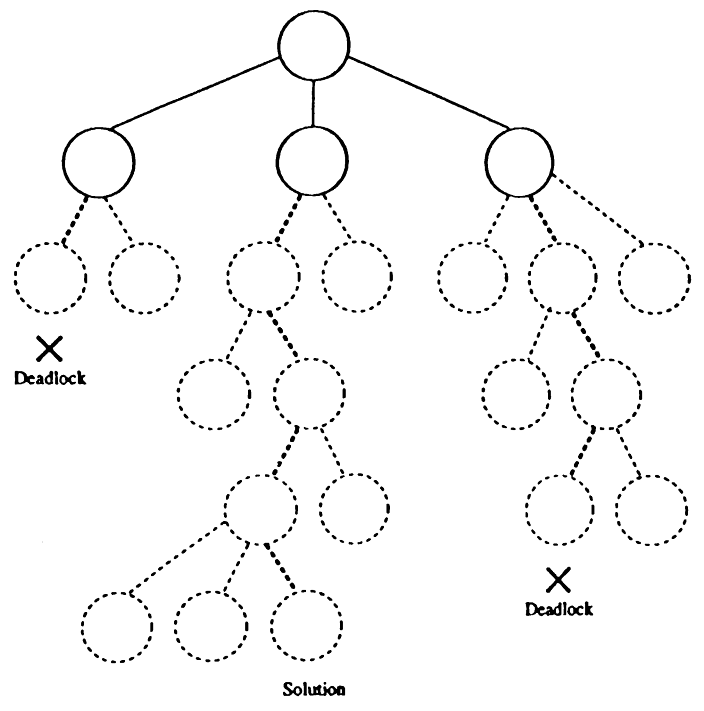


Figure 3.2. Step 1 of deadlock avoidance mechanism.

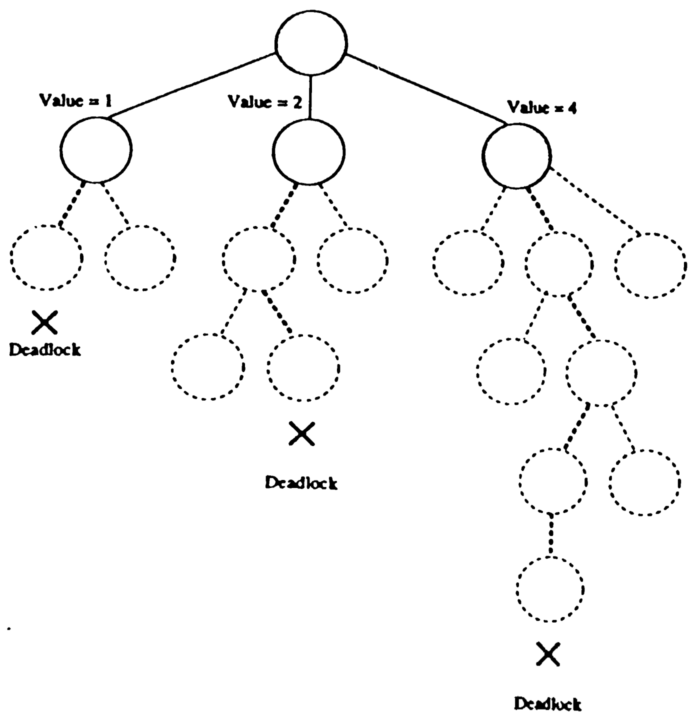


Figure 3.3. Step 2 of deadlock avoidance mechanism.

buffer space occupied. Thereafter the sprout algorithm resumes its regular steps. When exception step is used, the filtering mechanism filters out the nodes with the highest local evaluation function value. Because in this way jobs with higher priority value are kept in the system and jobs with lower priority value become candidate jobs, whose routes are to be interrupted. Then these nodes are passed to global evaluation function and the nodes which cannot resolve the deadlock are detected during global evaluation. In this way, deadlocks are resolved with the expense of increasing flow time of jobs which are moved to the L/U station.

3.2. System considerations and experimental conditions

As shown in Figure 3.4, the hypothetical FMS under study consists of six machines each with a finite buffer capacity, and one load/unload (L/U) station. Parts

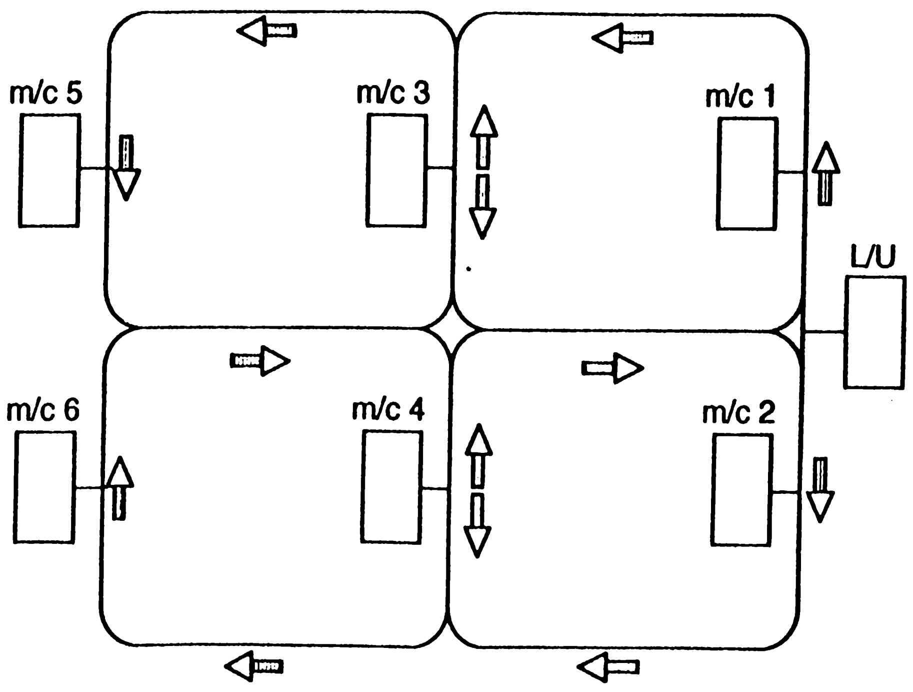


Figure 3.4. A schematic view of the hypothetical FMS under study

are transferred by three AGVs in the system. The distance between two ends of each segment in the layout is 5 distance units. Parts enter and leave the system through the L/U station. This station is also used as a central buffer area when blockings occur in the system. All of the jobs are assumed to be ready at the L/U station at time zero. Randomly generated 25-job problems are used in the experiments. Each job has either 5 or 6 operations with equal probability and each operation is assigned to a different machine. Hence, machine loads are kept nearly equal. Operation times are drawn from a 2-Erlang distribution. The performance of the proposed algorithm is measured under various operating conditions with the following experimental factors: 1) machine load, 2) AGV load, 3) local buffer capacity of machines; 4) routing flexibility, 5) sequence flexibility, 6) due date tightness, 7) scheduling criteria. For each of the above factors, except the scheduling criteria, two levels (low and high) are considered in the experiments (Table 3.2).

Factor	Low	High
Machine load level (ML)	15	30
AGV load level (AL)	2	1
Routing flexibility (RF)	1	2
Sequence flexibility (SF)	0.25	0.75
Queue capacity (Q)	2	5
Tardiness factor (TF)	0.35-0.40	0.85-0.90

Table 3.2. Experimental factors and their levels

As suggested by Sabuncuoglu and Hommertzheim [32] machine loads are set by varying the mean of the operation time distribution and AGV load levels are adjusted by changing AGV speeds. Similarly, the mean of the processing time distribution is set to 15 and 30 for low and high machine load levels, respectively.

The queue capacity of the machines is set to 2 and 5, corresponding to tight and loose values. Routing flexibility measure is taken from Chang et al. [5] who defined it in terms of the average number of machines that an operation can be processed. The value is set to 1 and 2 for low and high levels of this factor, respectively. We assume that the first assigned machine is the ideal machine with the least processing time. The processing time on the alternative machine is computed by adding a random number to the processing time of the operation on the ideal machine. This random number comes from a uniform distribution with a mean of half the processing time of the operation on the ideal machine.

Sequence flexibility measure is adopted from Rachamadugu and Scrieber [27]. According to their approach, operations of a job are viewed as nodes on an acyclic graph. The density of precedence arcs on this graph determines the degree of sequence flexibility. Its equation is as follows:

SFM=1.0-(2*all precedence arcs)/(n*(n-1))

where n is the number of operations. The SFM value ranges between 0.0 and 1.0. The closer is SFM to 1.0, the higher the sequence flexibility a job possesses. In our experiments, SFM is set to 0.25 and 0.75 for low and high sequence flexibilies, respectively.

Tardiness factor (TF) is defined as follows. Given a set of jobs and their transportation requirements with flexibilities fixed at their low levels and queue capacity set to its loose level, tardiness factor (TF) is equal to the fraction of tardy jobs that comes from the solution of the problem by using dispatch rules. This factor helps to examine the effects of individual factors on the system performance and to compare different solution methods (i.e. the proposed algorithm and scheduling rules). The tardiness factor is currently set to 0.85-0.90 for tight due dates and 0.35-0.40 for loose due dates.

In practice, the due dates are dictated by the customer and called exegenous due date assignment. At other times, the due dates are totally under the control of a company which sets them based on expected completion time of parts. This type of due date setting is called endogenous due date assignment. Both exogenous and endogenous due date assignment methods are implemented in this study. Exogenous due dates are generated from a uniform distribution with a particular mean and variance. The mean of the uniform distribution is varied to obtain the desired value of due date tardiness factor (TF). Endogenous due dates are assigned by the total work content (TWK) rule, because this rule has been found to be robust in the previous studies. According to this rule, due date of a job is determined by multiplying total work content of the job by a constant multiplier so that the desired TF value is achieved.

Performance of the algorithm is tested for makespan, mean flowtime, and mean tardiness criteria. Both the proposed algorithm and the simulation model used to implement the scheduling rules are coded in C programming language. Computations are performed on a Sun 4 workstation. Five different problem sets are randomly generated for each factor combination. The performance of the proposed algorithm is compared with the scheduling rules listed in Table 3.3. The MODFIFO rule is a modified version of the FIFO rule such that priority is given to jobs in the output queue of the machine at which the material handling device waits at that moment.

In the simulation model used to implement the rules, an alternative machine in the process route is selected using the least total work content criterion. If there is more than one AGV available to transfer the part, the one closest to the machine which is demanding service is selected. In addition, the deadlock preventive scheme proposed by Sabuncuoglu and Hommretzheim [32] is used. This type of use of scheduling rules in a simulation model (i.e. event based scheduling) is a typical

Criteria	Machine scheduling rules	AGV scheduling rules
Makespan	MWRK:most work remaining LPT:largest processing time MTWK:most total work content	STD:shortest distance MODFIFO:modified FIFO LQS:smallest queue space left
Mean flowtime	LWRK:least work remaining SPT:shortest processing time	STD:shortest distance MODFIFO:modified FIFO LQS:smallest queue space left
Mean tardiness	MDD:modified due date MOD:modified operation due date	STD:shortest distance MODFIFO:modified FIFO LQS:smallest queue space left

Table 3.3. The list of machine and AGV scheduling rules

example for implementation of on-line scheduling schemes. Therefore, this chapter will also provide a comparison between the off-line and on-line scheduling schemes in a deterministic and static environment.

3.3. Computational Results

3.3.1. Comparison of the algorithm with the scheduling rules

In this section, performance of the proposed algorithm is compared with the scheduling rules (on-line scheduling scheme). As discussed in the previous section, a number of randomly generated problems are used in the experiments. In addition to the experimental factors discussed in the previous section, schedule generation scheme is added as another factor in the full factorial experimental design. The computations are performed by using makespan, mean flowtime and mean tardiness criteria. The results revealed that the schedule generation factor is statistically significant in favor of the proposed off-line algorithm. The analysis also indicated that two way interactions between the scheduling scheme and other factors are significant (Table 3.4). These are explained in detail for each performance measure in the following sections.

	Mean flowtime	Makespan	Mean tardiness (endogenous)	Mean tardiness (exogenous)
Factors	AL,ML,RF,Q	AL,RF,Q	AL,ML,SF,TF,Q	AL,RF,TF,Q

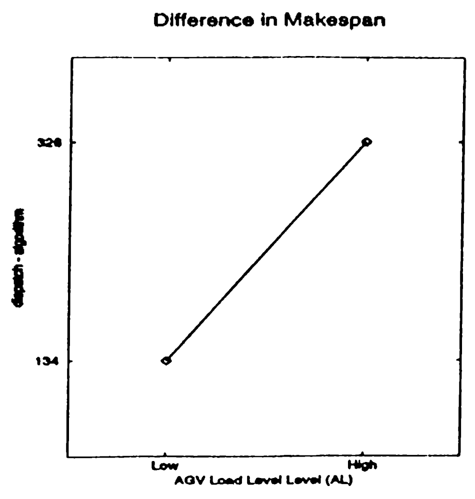
Table 3.4. Factors that have significant two way interactions with scheduling methods.

Makespan

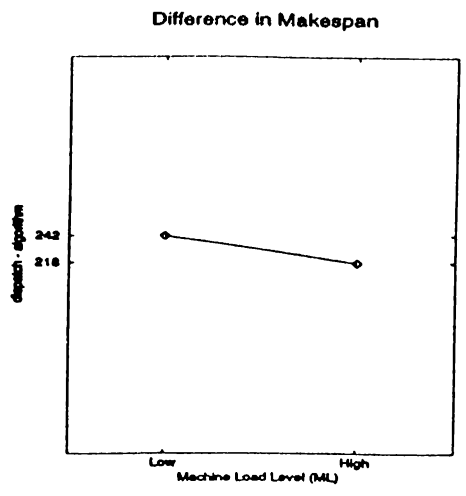
According to the results of pilot experiments, the two combinations, MWKR/MODFIFO and MWKR/LQS are selected for scheduling rules. Specifically, the first rule pair is the good choice when buffer capacity is loose, whereas the second rule pair is better in tight queue capacity cases. Hence, these two rule combinations are used as the on-line methods in the experiments.

On the average (over all of the factors) the algorithm provided a 20% improvement over the scheduling rules. Figure 3.5 illustrates makespan differences between the proposed algorithm and the scheduling rules for every experimental factor. The points on a factor's graph is obtained by averaging the differences in makespan, fixing the factor's level to a specific value and taking average over all levels of all other factors. In general, the graphs show that the proposed algorithm outperforms the scheduling rules at every level of each factor. The results of the ANOVA test also confirmed that differences in the relative performances of the algorithm and rules are more significant when AL and RF is high, and Q is tight. Notice that the slopes of these graphs also measure the effect of a factor to differences in the performances of the algorithm and scheduling rules. As can be seen in Figure 3.5, the proposed algorithm performs particularly well when AGV load is high and queue capacity is tight. For example, the percentage improvement of the algorithm is as high as 34% when AGV load is high and queue capacity low.

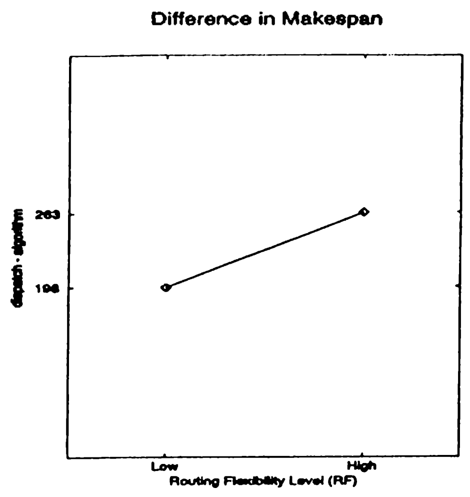
The computation time of the algorithm depends on the number of schedulable operations and the level of flexibilities in the system. On the average 25-job problems are solved within 350 to 450 CPU seconds when flexibilities are high. This reduces to 20 to 40 CPU seconds when the flexibilities are at low levels.



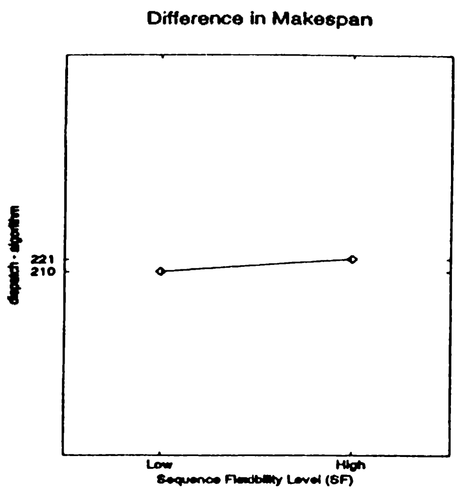
(a)



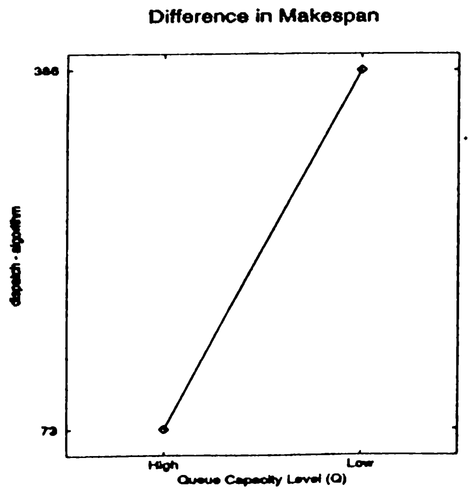
(b)



(c)



(d)



(e)

Figure 3.5. Makespan difference between scheduling schemes

Mean flowtime

Similar observations can be made for the mean flowtime criterion. In this case, LWRK/MODFIFO and LWRK/LQS are used as the machine and AGV scheduling rule combinations for loose and tight queue capacity cases, respectively. As can be seen in Figure 3.6, the proposed algorithm provides a substantial flowtime improvement (about 30%) over these scheduling rules. Again, differences in the performances of the algorithm and rules become more significant as resource constraints get tighter and routing flexibility is increased. However, this difference is not statistically significant at any level of SF. Thus, both schemes utilize the SF at the same rate. As in the makespan case, AGV load level and queue capacity tightness are the two most dominating factors for their performance differences. Specifically, the improvement is more than 39% when the AGV load is high and the queue capacity is tight. Whereas, it is 21% in the reverse case.

Mean tardiness

In the tardiness case, MDD/MODFIFO and MDD/LQS rule combinations are selected to be compared with the proposed algorithm. Figure 3.7 shows differences in the mean tardiness between the algorithm and the scheduling rules for each of the due date assignment methods. The algorithm outperforms the scheduling rules at each level of every factor. With exogenous due date assignment method, except for machine load case, the algorithm performs better than the rules at high or tight level of the factors. With endogenous due date assignment, the algorithm seems to perform relatively worse, at high routing flexibility level, although this difference is statistically insignificant. In any case, the algorithm achieves very small mean tardiness values at high levels of the flexibilities. Moreover, the fraction of tardy jobs is reduced significantly by the algorithm at high levels of flexibilities. One important point that is not seen on these graphs is that, when due dates are assigned endogenously, setting machine load to its high level, does not increase the mean tardiness obtained by the algorithm. But the performance of the scheduling rules are affected by this type of change. This is a very good property of the algorithm. When the relative performance

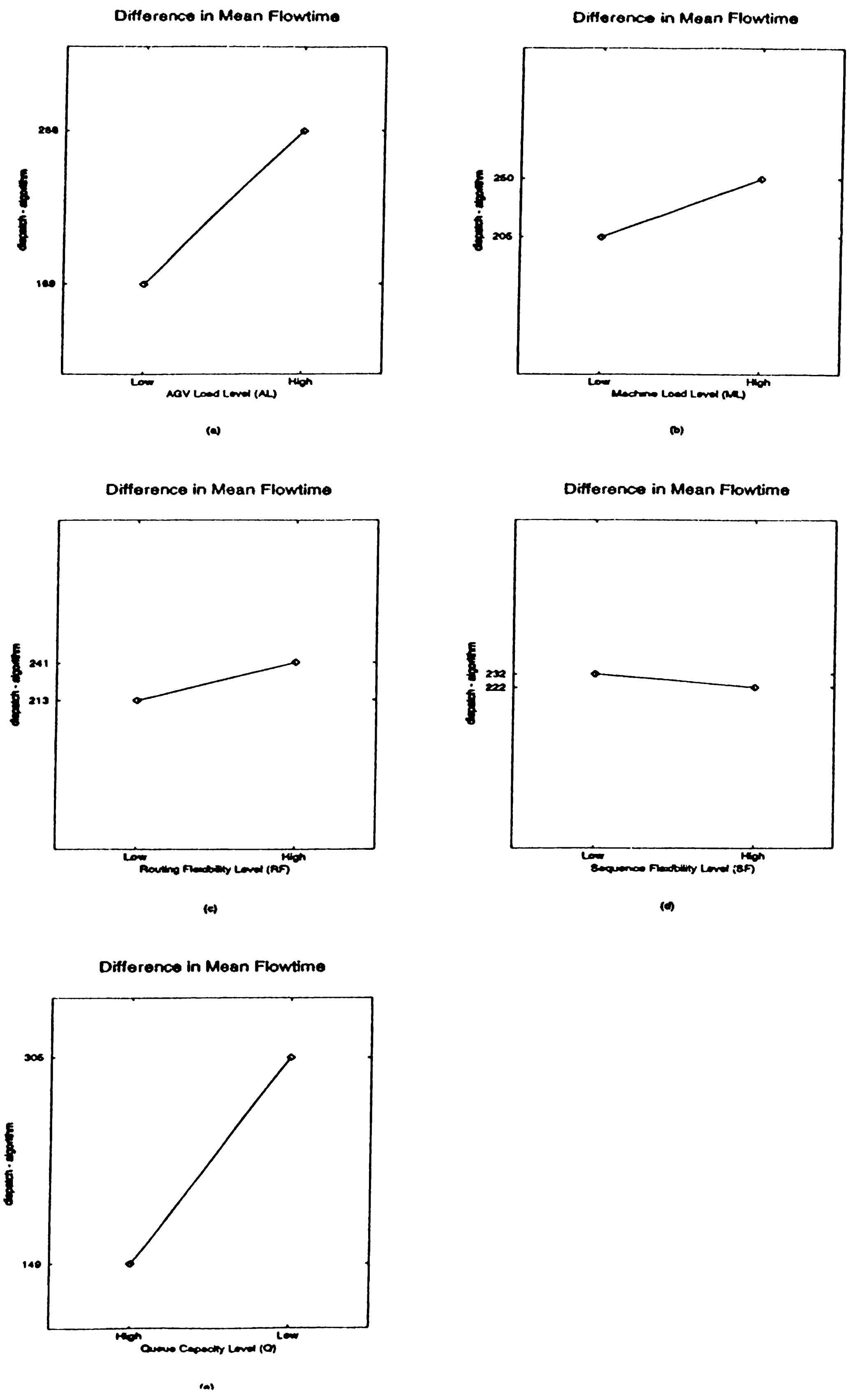


Figure 3.6. Mean flowtime difference between scheduling schemes

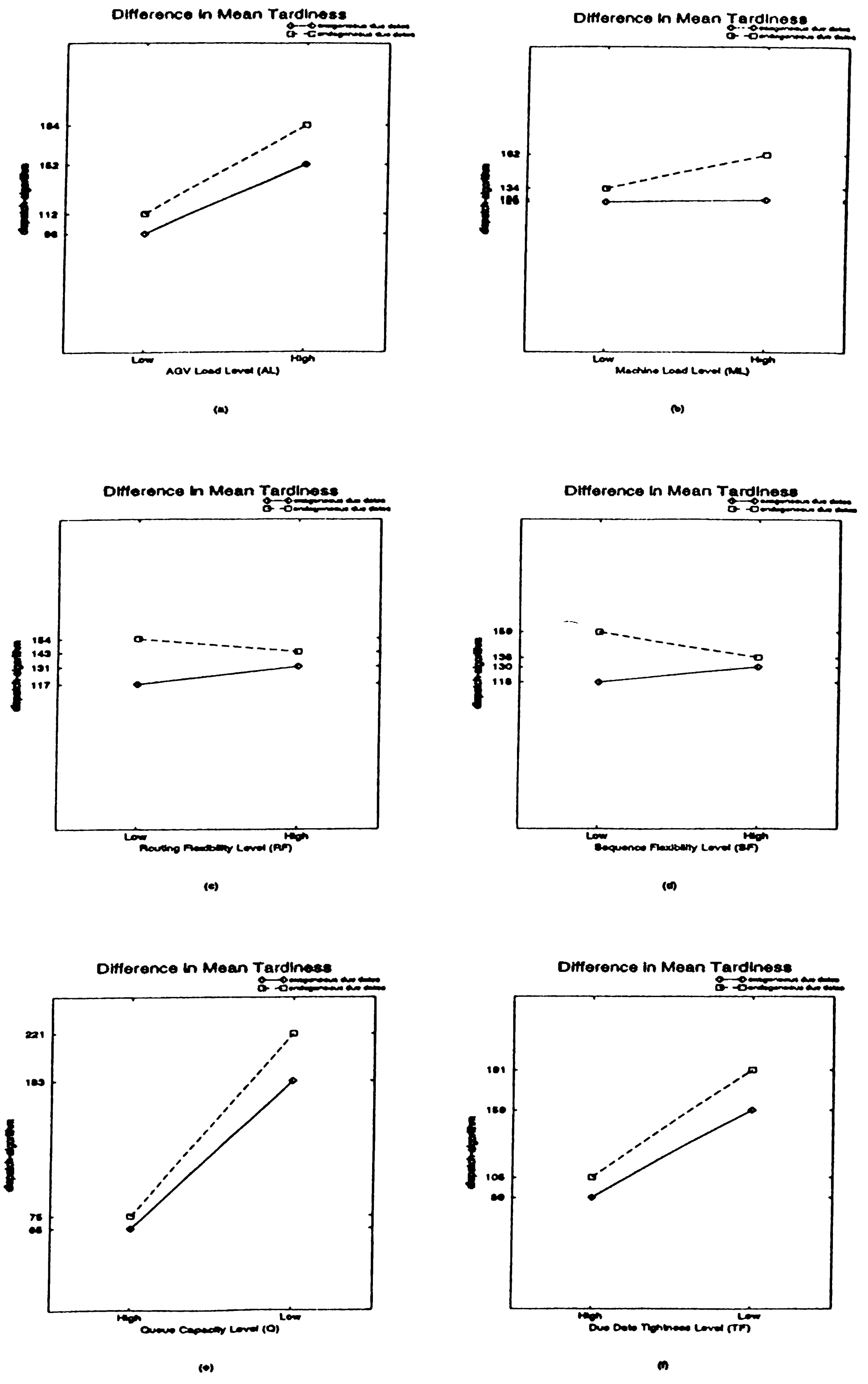


Figure 3.7. Mean tardiness difference between scheduling schemes

of the algorithm is compared according to the two due date assignment methods, it produces better results with endogenous due date assignment method. Again, from the slopes of graphs, queue capacity level, AGV load level, and due date tightness level seem to be the most important factors in terms of their effects to the relative performances of the algorithm and rules.

3.3.2. Performance evaluation of the algorithm and the effects of scheduling factors

In this section, performance of the scheduling algorithm is analyzed with respect to scheduling factors. These results provide valuable insights into how various scheduling factors in an FMS environment (i.e. machine load levels, flexibilities, etc.) interact with each other and affect the system performance. Another ANOVA test is performed for each scheduling criterion to test the significance of the main factors and high order interactions. Table 3.5 summarizes the test results in terms of the main factors and high order interactions that are found significant at 1% level.

In general, effects of the main factors are significant for each of the performance measures. Only exception is noticed for ML in the mean tardiness case with endogeneous due date assignment method. This is due to the fact that both due date assignment and machine load level determination methods are based on the total work content rule. Consequently, due dates are postponed to later dates (i.e. loose due dates) at higher values of machine loads.

The effects of routing and sequence flexibilities on the system performance are also significant. This means that performance of the system can be improved considerably by utilizing the available flexibility inherent in FMSs. As it is expected, high values of machine and AGV load levels, and the low value of level of buffer capacity have adverse effect on the system performance. It can also be observed from Figure 3.8 that effects of factors on the system performance are different for different performance criteria. For example, effects of ML and AL are greatest for the

	Mean flowtime	Makespan	Mean tardiness (endogeneous)	Mean tardiness (exogeneous)
Main factors	RF,SF,AL,ML, Q	RF,SF,AL, ML,Q	RF,SF,AL,TF,Q	RF,SF,AL,ML, TF,Q
Two-way interactions	RF,SF	RF,SF	RF,SF	RF,SF
	RF,AL	RF,AL	RF,AL	RF,AL
	RF,ML	RF,ML	RF,ML	RF,ML
	SF,AL	RF,Q	RF,TF	RF,TF
	AL,ML	SF,AL	RF,Q	RF,Q
		SF,Q	SF,TF	SF,TF
		AL,ML	AL,TF	SF,Q
		ML,Q	Q,TF	AL,ML
				AL,TF
				ML,TF
				TF,Q
Three-way interactions	None	None	RF,SF,TF	RF,AL,TF
			RF,AL,TF	RF,ML,TF
			RF,ML,TF	AL,ML,TF

Table 3.5. Summary of ANOVA results for each scheduling criteria

mean flowtime and the makespan criterion. Whereas, TF and RF are two most dominating factors for the mean tardiness criterion irrespective of the due date assignment methods.

Two and three way interactions are also analyzed. The results indicate that the impact of RF (SF) is greater with low values of SF (RF). The effect of RF is greater when AL is high and ML is low. Especially, the impact of RF is magnified at the low level of ML. This is because the processing time of an operation at an alternative machine depends on the processing time at the ideal machine. As ML is increased the processing time at an alternative machine increases at a higher rate. Nevertheless, RF still improves performance at the high value of ML. RF has also interactions with Q and TF and its effect becomes stronger when these factors are at their tight levels. We also observe interactions of SF with AL, Q and TF. In all of the cases, SF improves the performance at higher rates when these factors are more binding. TF has also interactions with all the other factors and the examination of these interactions reveal that the effects of these factors are stronger when TF is tight.

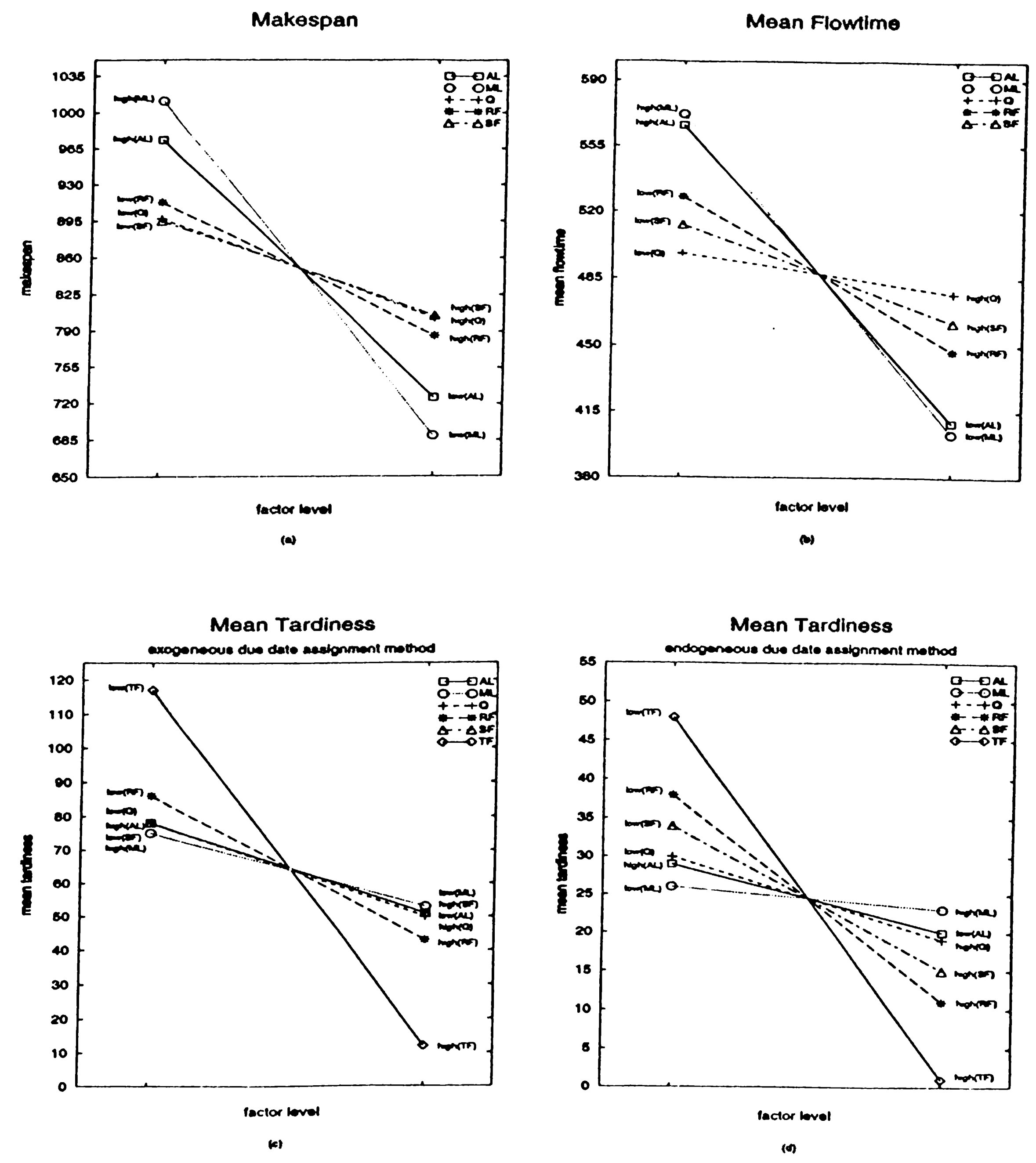


Figure 3.8. Effects of scheduling factors

As can be seen in Table 3.5, TF also makes some of the combined effects of the factors stronger in the three way interactions.

Another important interaction is observed between AL and ML. Their interaction indicates that the adverse effect of increasing the level of AL(ML) is small when ML(AL) is at its high level. Similarly, the effect of increasing AL(ML) level becomes stronger when ML (AL) is at its low level. This implies that, among the factors AL and ML, if one of them is at its high level, changing the level of the other does not affect the system performance significantly. Finally, the interaction between ML and Q in the makespan case indicates that the effect of Q becomes stronger when ML is high.

In summary, an off-line FMS scheduling algorithm is developed in this chapter. The proposed algorithm considers a wide variety of system resources in an FMS environment. The computational results indicate that the algorithm performs significantly better than the rules under various experimental conditions for each of the scheduling criteria. Especially, differences in the relative performance of the proposed off-line algorithm and the scheduling rules increase as the system resources become tighter. Although the computation time of the algorithm is higher than that of the scheduling rules, improvement in the performance of the system can justify its computational burden.

CHAPTER 4

A SIMULATION BASED SCHEDULING SYSTEM

4.1. Introduction

Today, simulation is accepted as one of the most valuable OR tools in practice. This can be attributed to several reasons such as the reduction in the cost of computers and development of new flexible simulation languages. The increased use of simulation is due to the growing need for solving complex problems in business and manufacturing. Especially, the ability of simulation models to capture necessary details of dynamic and complex systems makes simulation the most used OR tool. This characteristics of simulation is especially important for Flexible Manufacturing Systems (FMSs) because it is very difficult for analytical models to properly handle the detail and complexity of such systems. Hence, from current FMS practice, simulation is seen as one of the most frequently used OR tool.

From current practice, simulation applications can be classified into stand-alone applications and hybrid applications. In the former case, which accounts for the majority of simulation applications, a simulation model is used as a test-bed for evaluating different design alternatives or operational policies without disturbing the actual system. In a typical situation, long and multiple runs are taken from the simulation model and its results are analyzed by statistical methods. This type of simulation application can be called as an off-line use of simulation because there is no real time communication between the simulation model and the system elements. In general, the off-line use of simulation gives an overall picture about the system being simulated. In the second category, there are hybrid applications of simulation with other scientific tools such as expert systems (ES)/artificial intelligence (AI) and analytical techniques. These hybrid systems are usually developed for real time operation and control of the manufacturing systems. This approach also facilitates the on-line use of simulation as it is invoked more frequently in this mode. The simulation model discussed in this chapter has also several on-line capabilities.

The purpose of a hybrid model is to combine the powers of its constituting elements to solve much larger and complex problems with reduced computational efforts (Shanthikumar and Sargent [35]). In general, scheduling problems are in this nature. Except in relatively simple cases, determination of optimum schedules by analytical means is extremely difficult. The problem is further complicated by the dynamic and stochastic nature of manufacturing environment in which schedules must also be maintained (or updated) frequently over time. Traditional approaches (i.e. scheduling algorithms and math programming) may not be self-sufficient in dealing with these problems. Simulation methods, artificial intelligence techniques, or their combinations may also be needed for efficient operations of advanced manufacturing systems. In this chapter, one such a hybrid approach in which both simulation and analytical model is utilized, is described in detail.

The idea of integrating simulation model and scheduling algorithms has existed for a long time. There are already some studies in which several simulation plus ES based scheduling systems are proposed (e.g. Manivannan and Banks [20], Wu and Wysk [40]) and their implementation issues are discusses (Harmonosky & Robohn [12]).

4.2. The proposed scheduling system

4.2.1. Description of the system

It can be observed that the majority of simulation applications to scheduling problems are in the form of testing several on-line scheduling policies or rules. Simulation of off-line scheduling methods has not received considerable attention from the literature. This is partly due to difficulty in applying simulation to the off-line generated schedules in a dynamic and stochastic manufacturing environment. In this section, we describe a simulation model that implements both on-line and off-line scheduling methods. The proposed model also provides a framework to compare a wide range of reactive scheduling policies under different environmental conditions. The proposed system is coded in C programming language and implemented in UNIX environment using a SUN workstation. The length of the source code is more than 7000 lines and the size of the executable code is about 150 Kbytes.

As shown in Figure 4.1, the simulation-based scheduling system consists of three major components: scheduler (scheduling module), simulation model, and controller. Next, the basic functions of each module will be described.

Scheduler is responsible for making all scheduling decisions. Given the system status and other relevant data including the scheduling method (e.g. on-line, off-line, etc.), it generates a partial or complete schedule. It contains several machine and AGV scheduling rules and the scheduling algorithm described in chapter 3.

Simulation model uses two sets of input data: system related data and values of environmental parameters. System related data consists of a physical description of the manufacturing system (e.g. number of machines, number and speed of transporters, layout). Arrival rate of jobs, parameters of stochastic events (e.g. machine breakdown rate, processing time variation), part types, machine and part flexibilities constitute the environmental parameters. In the simulation model, machining subsystem, movement of material handling equipment (AGVs), and in-process storage capacity are represented in greater detail. The main task of the simulation model is to execute (or implement) the scheduling decisions which is made by the scheduler and downloaded by the controller. This decision can be in the form of a partial machine and AGV schedule or a single decision for a resource. When an on-line scheduling policy is implemented, a resource triggers the controller upon completing a task which, in turn, invokes the scheduler. The scheduler makes a decision by applying some scheduling rules and passes the final decision to the

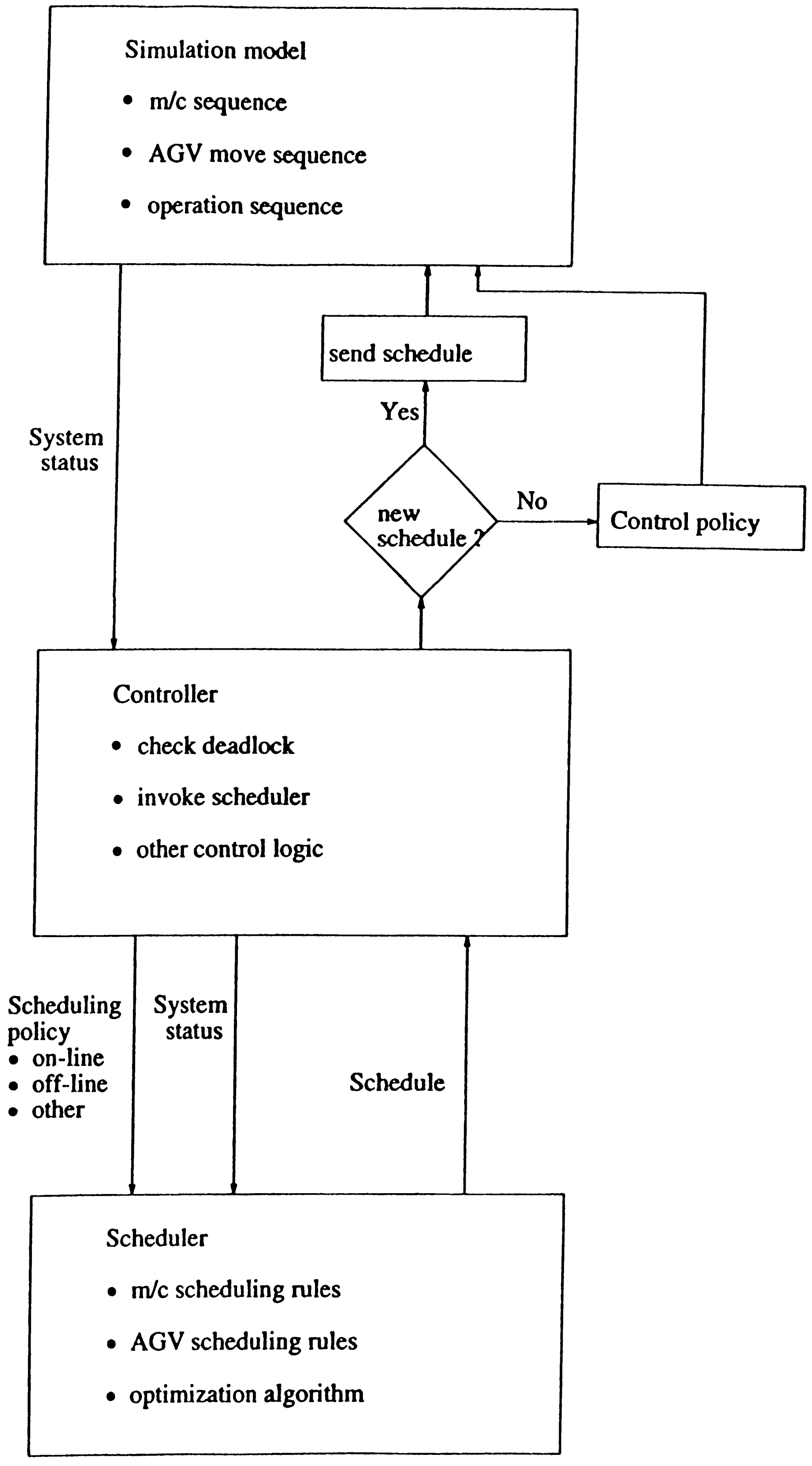


Figure 4.1. A simulation based scheduling system.

controller. Then the controller sends this schedule to the simulation model for execution. In the other case, a partial schedule is passed to the simulation model in the form of a machine processing sequence, AGV move sequence and operation processing sequence for the jobs.

The control module examines the state of the system at every discrete event that occurs in the simulation model and provides appropriate course of actions to be executed by the simulation model. The control module has the following tasks:

- Keep up with the machine and AGV sequence in off-line mode
- Avoid and resolve deadlock situations
- Implement scheduling policies

The objective in simulating an off-line schedule is to observe its results in a stochastic environment. However, it is not easy to follow the exact start and completion times imposed by the off-line schedule in a dynamic and stochastic environment. When this is not possible, machine processing sequences and AGV move sequences are tried to be followed as close as possible to the original schedule.

In most of the manufacturing systems, in-system storage capacity is limited. Hence, there is always a possibility for blocking (and locking) in the system due to finite capacities. This necessitates the use of effective control policies to avoid blocking of material movement in the system. In the literature, the problem has often been addressed as a part of the on-line scheduling (Egbelu and Tanchoco [9], Sabuncuoglu and Hommertzheim [32]). However, it has not been thoroughly studied for off-line scheduling purpose. The problem is more complicated due to a fixed sequence that material handling transporters have to follow in the off-line mode. This is illustrated with the following examples. In the first case of Figure 4.2, part #2 waits for part #1 to be picked up from the current machine and delivered to the destination station. However, part #1 also needs to wait for part #2 for its delivery according to the AGV schedule. This contradicts with the demand of part #1, leading to a deadlock situation. In the second case, there is a similar dependency between part #1 and part #3.

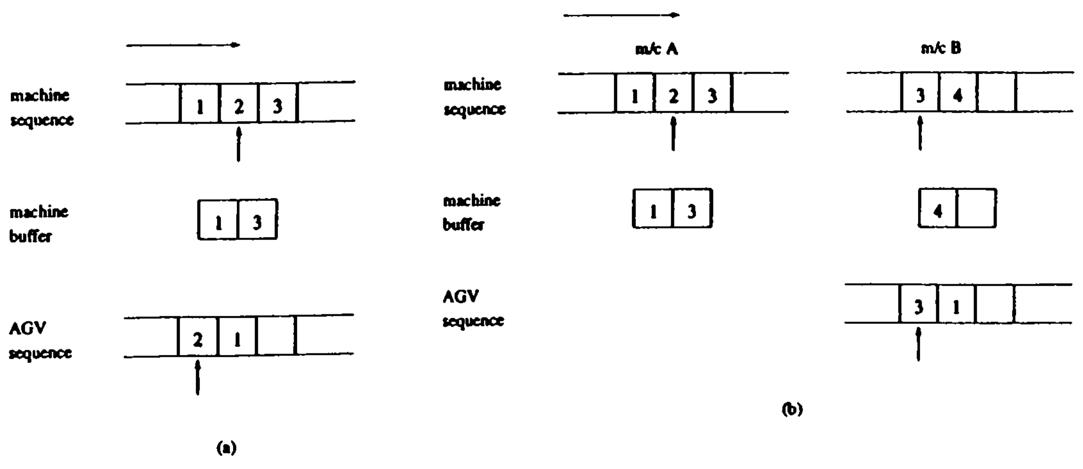


Figure 4.2. Two deadlock examples

As the third task, the controller is responsible for implementation of scheduling policies by considering the environmental conditions over time. In order to accomplish this, the controller must either be supplied with the appropriate control policy or must simulate alternative policies and choose one according to the simulation results. The first case is encountered in off-line use of simulation, whereas the second stands for on-line use. In the second case, simulation is also used to evaluate different policies at decision points. This method has the advantage of being more adaptive to the dynamically changing manufacturing environment. However, the trade-off between simulation run length and statistical validity of the simulation results becomes an important issue (Harmonosky [13]). An expert system can also be used to reduce the number of scheduling policies that will be evaluated by the simulation model (Wu and Wysk [40]). The controller in the proposed scheduling system can take a snapshot of the simulation model at any instant and then recover the saved state at a later time. In the proposed system, all these features are incorporated. At a decision point a snapshot of the simulation model can be saved and then alternative policies can be evaluated using the same simulation model.

4.2.2. Implementation issues

As already mentioned, the proposed simulation based system is implemented using a general purpose programming language (i.e. C language). The other option was to use a specialized simulation language. There are advantages and disadvantages with each. However, the advantages associated with using a general purpose language are far better for this case.

From modeling point of view, simulation languages provide a higher level of abstraction to build a model. Although this helps in constructing the model easily and quickly, it also brings restrictions. In most cases the control logic of the simulation model cannot be implemented with the routines supplied by the simulation package. This is the most crucial part of a simulation model because simulation is mostly used to evaluate different control policies. In such a case, an user written code is interfaced with the simulation language. Of course, this brings an overhead to the users. As the control logic gets more complex, the use of a simulation language becomes less attractive. To give a simple example, suppose that in a jobshop different queuing disciplines will be evaluated with the help of a simulation model. It is easy to build the jobshop model with a simulation language which represents the physical system. But in order to implement alternative queuing rules, appropriate code must be written with a general purpose language and special routines which interfaces the user's routine with the rest of the model must be used. Another issue is that the description of a model coded with a simulation language cannot be changed without recompiling. On the contrary, a model coded with a general purpose language can alter its description by taking input during run time.

From implementation point of view, general purpose languages produce faster and more compact executable codes than simulation languages. To give a specific example, the simulation language SIMAN produces at least 1300 Kbytes of executable code when it compiles a model which includes user written code. On the other hand, our proposed system contains over 7000 lines of computer code (including all scheduling algorithms) and the size of the executable, when compiled using the C compiler on SUN computer systems with the optimization flag of the compiler set, is only 150 Kbytes. In a hybrid application of simulation with an optimization model speed is very important to attain real time response. Another important implementation issue is debugging. Although most simulation languages have built in debuggers, these are only used to trace the code written with the language. There are no debugging tools available to trace the user written code. In our implementation we used a UNIX based debugger called ups. It is a shareware program and provides an integrated visual environment for debugging.

A significant advantage of simulation languages over general purpose languages is that they provide statistical analysis tools and animation facilities. These are important to analyze and monitor the system and they are hard to implement with a general purpose language. In our case, however, special purpose animation routines can be written in the X-Windows environment without much effort.

In summary, the structure of the simulation based scheduling system is described in this chapter. The proposed system enables the use of simulation in scheduling environment. It can be used both in on-line and off-line decision making modes. In this study, we use simulation in off-line mode to investigate the performances of various scheduling schemes under different simulated environments. The results of these experiments are discussed in the next chapter.

CHAPTER 5

COMPARISON OF SCHEDULING SCHEMES

This chapter discusses the evaluation of three scheduling schemes, namely on-line, off-line and quasi on-line schemes. The simulation based scheduling system described in the previous chapter is used to obtain computational results in various simulated environments. Section 5.1 describes the implementation of the quasi on-line scheme within the proposed scheduling system. Section 5.2 gives a comparison in deterministic and static environment. The simulation results of static and stochastic environment are discussed in section 5.3. Finally section 5.4 provides the results for a dynamic environment.

5.1. Implementation of the quasi on-line method

In this thesis a new scheduling scheme called quasi on-line is also developed. Basically, this scheme makes a trade-off between on-line and off-line schemes. In the proposed approach, a partial schedule is generated whenever a scheduling decision is needed. This schedule is then executed until the most imminent time at which a resource completes its assigned tasks. At that point a new partial schedule is generated and the same process is repeated. Hence, the time between each scheduling point is not fixed a priori, but the time window parameter which basically determines the extend of information usage is fixed. In the quasi on-line scheme, more than one task to resource assignment decisions are made as opposed to a single decision in the on-line scheduling case. The number of scheduling decisions to be made is controlled by the time window parameter. This parameter determines the extend of information usage and the degree of responsiveness.

For an illustration of the use of time window parameter, consider the example in Figure 5.1, in which a partial schedule is generated for the period $[t_{01},t_{02}]$. In this example, AGV #1 is the first resource that completes its assigned tasks at time t_{11}, provided that an unexpected event does not occur in the mean time. At this point in time, a new scheduling period is defined from t_{11} to t_{12} (i.e. $[t_{11},t_{12}]$). Here, all scheduling decisions that start before t_{11} are fixed and others are cleared. As seen in Figure 5.1, the second operation on m/c 3 is cleared and others are fixed. This procedure continues until all operations are scheduled. Notice that the proposed quasi on-line scheme is very similar to the on-line scheme except that several AGV and machine scheduling decisions are made when a resource becomes available.

5.2. Static and deterministic environment

In this section, the performance of the quasi on-line scheduling scheme is measured under different values of time window parameter. Initially, two parameter values are used in the experiments. In the former case, the parameter is set to an arbitrarily small value (e.g. 100) to limit the amount of information usage. Whereas, it is set to infinite to simulate an off-line scheme in the latter case. The methodology described in chapter 3 is also repeated here (i.e. a full factorial experimental design is done.) The results of the ANOVA table indicates that the off-line scheme performs significantly better than the quasi on-line scheme. This supports the earlier findings in chapter 3 that, the use of more information increases the quality of schedules significantly. The conditions under which off-line scheme performs relatively better (i.e. the extend of information usage is more effective) are also examined. The results (Table 5.1) reveal that, when resources are tight (AL, ML, TF high and Q low) or there is no alternative machines for operations (RF low), information usage becomes more critical. Hence, the off-line scheme performs particularly better than the quasi on-line scheme.

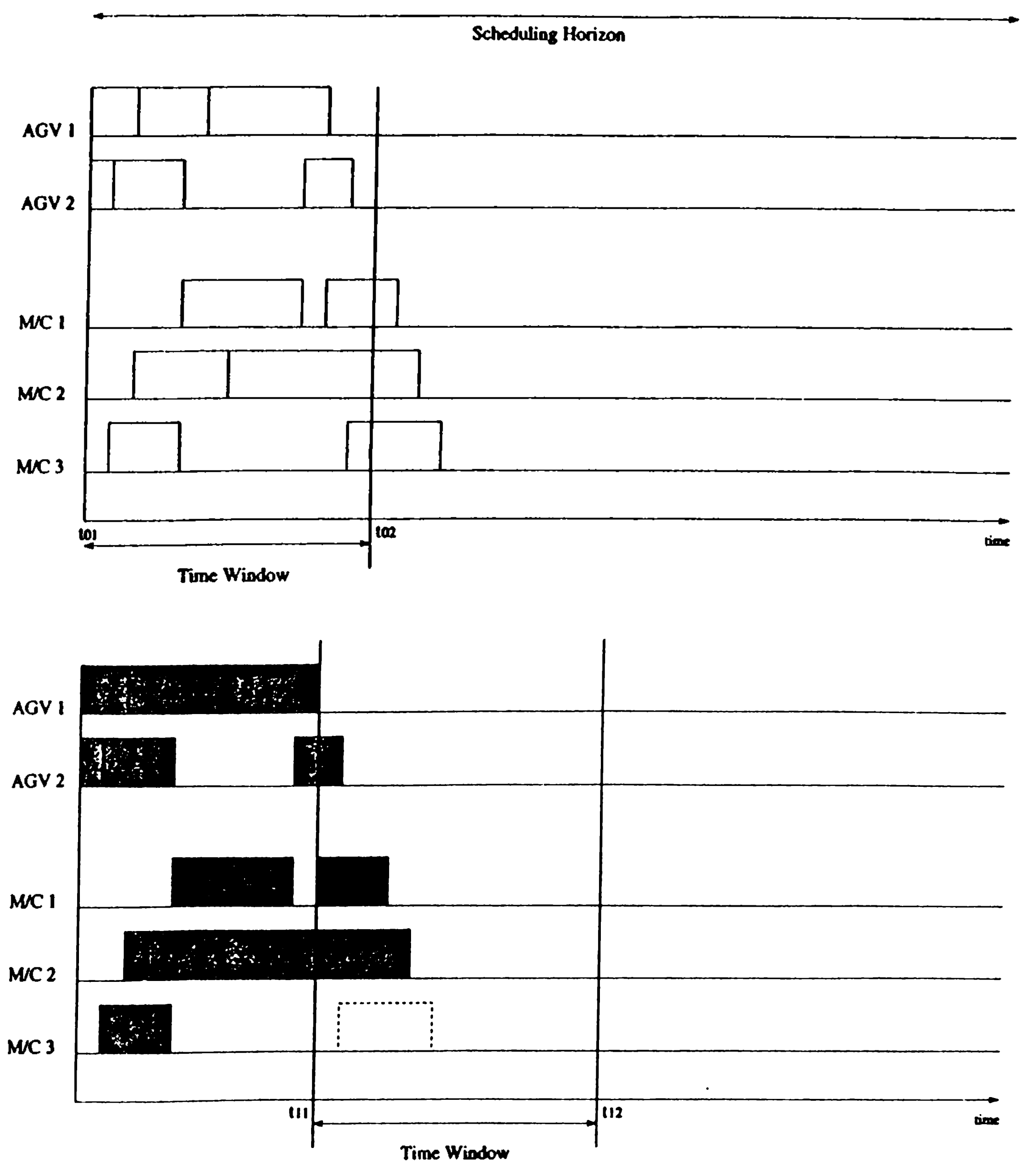


Figure 5.1 Implementation of the quasi on-line scheme

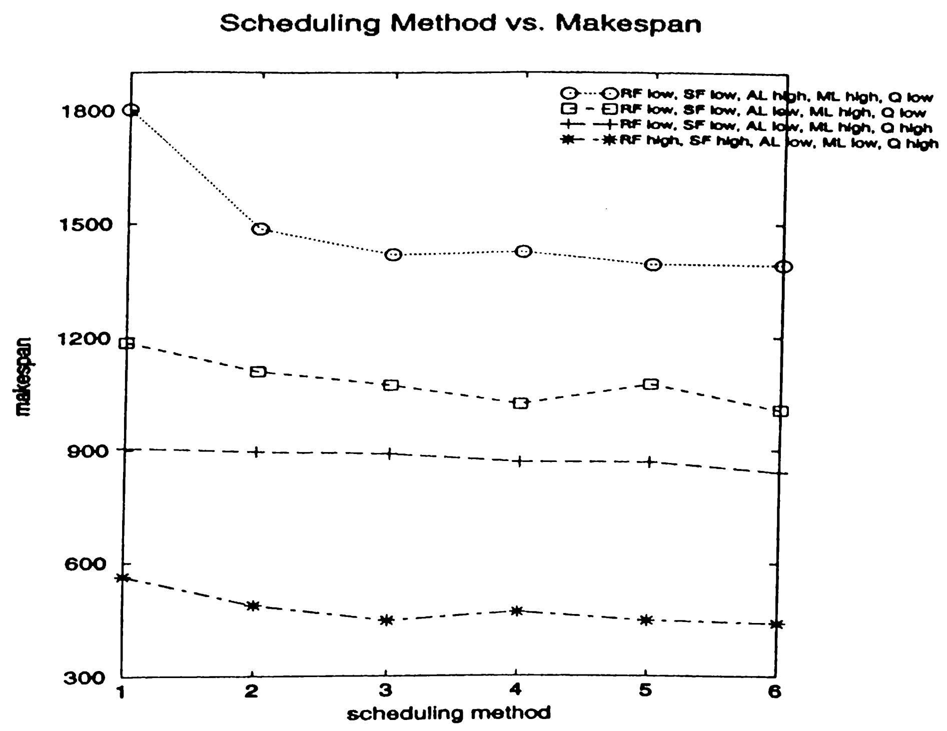
	Makespan	Mean Flowtime	Mean Tardiness	
			exogenous	endogenous
Factors	AL,RF,Q	AL,ML,RF,Q	AL,ML,RF,Q,TF	AL,ML,RF,Q,TF

Table 5.1. Experimental factors that have two-way interactions with the scheduling methods.

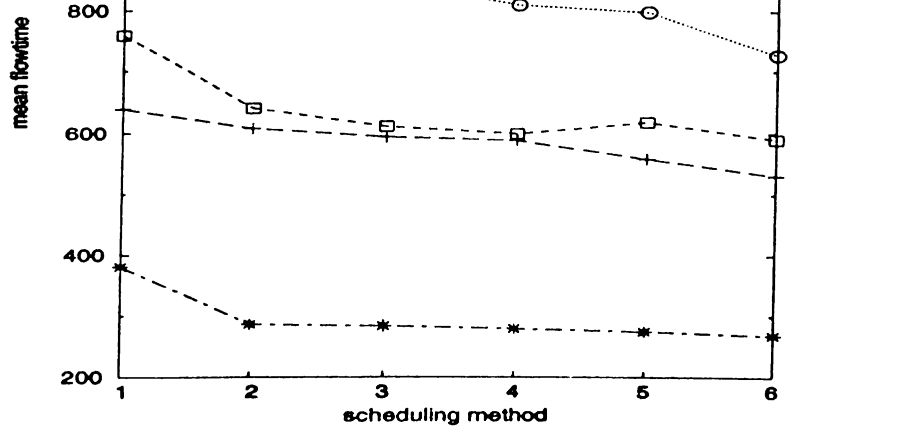
The use of a fixed time window for all factor combinations in the experiment may effect the interactions of the above mentioned factors. As the makespan of a specific problem varies, the number of decisions considered within a time window also vary. Hence, the extend of information usage may change. Two most important factors that effect the makespan of a problem are AL and ML. In order to understand the effects of AL and ML on the nature of the interactions mentioned above, the three way interactions are also examined. The results indicated that the above arguments hold at both levels of AL and ML.

Another set of experiments is performed to analyze the effects of the time window parameter on the quality of the schedules generated. The performances of on-line, off-line and the quasi on-line schemes are measured with four different combinations of experimental factors. These combinations have been determined as a result of the previous experiments. The levels of experimental factors are set so that we obtain the largest performance difference, smallest performance difference and two intermediate difference levels between the off-line scheme and the quasi on-line scheme. In these experiments, the value of the time window parameter is varied in such a way that it covers a certain fraction of the makespan value of a specific problem. These levels correspond to 1/8, 1/4, 1/3 and 1/2 of the makespan value of a problem. For example, if the typical makespan of a factor combination is 1600, then we fixed the time window values to 200, 400, 533, and 800 for the four levels.

In Figure 5.2, which illustrates the results of the above experiments, the scheduling methods numbered as 1 and 6 correspond to on-line and off-line schemes, respectively. The methods numbered 2 to 5, represent various versions of the quasi on-line scheme with different time window parameter values in increasing order. The results show that as the scheduling horizon increases from 0 (on-line method) to a full

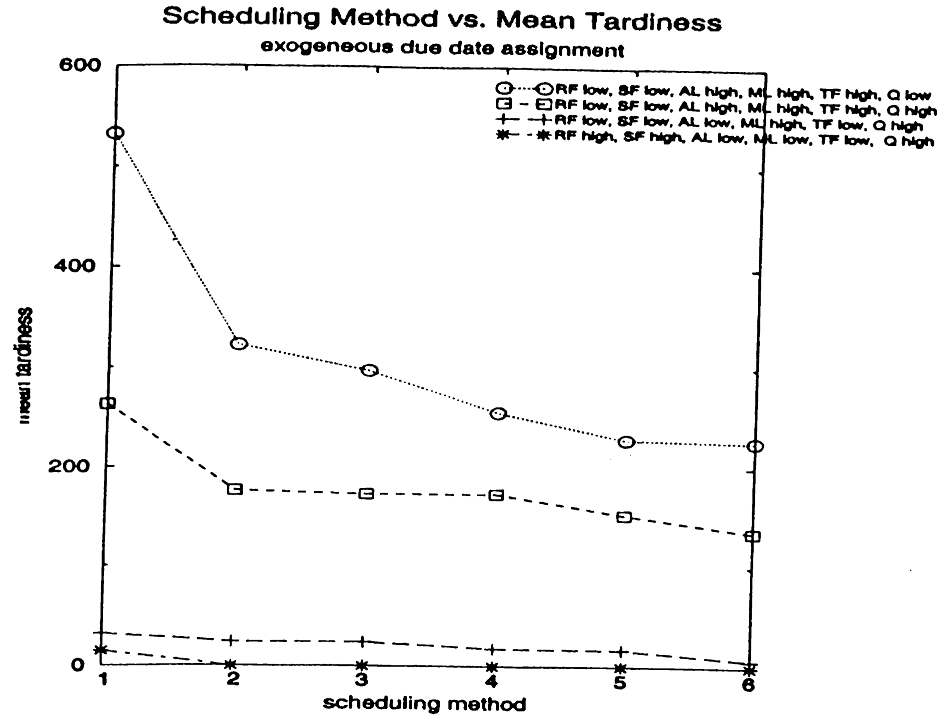


(a)

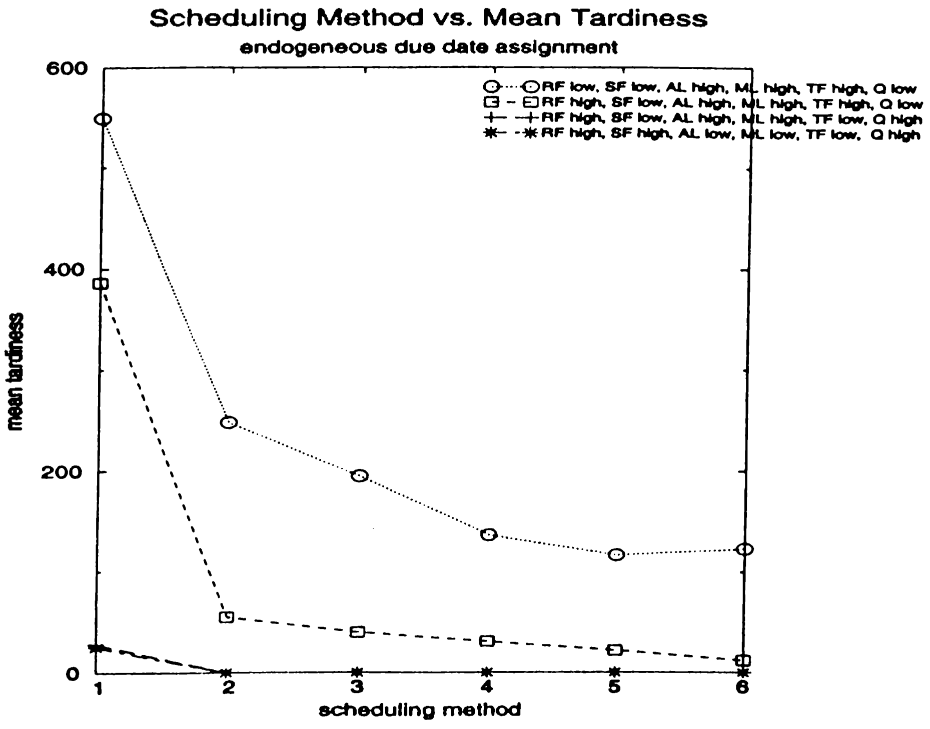


(b)

Figure 5.2 Comparison of scheduling schemes: deterministic and static environment



(c)



(d)

Figure 5.2 (Cont'd)

horizon (off-line method), the performance of the schedules tend to improve. The largest improvement is observed when switching from on-line to quasi on-line with the smallest time window value. The slope of the graphs is largest when AGV and machine loads are high, due dates are tight, buffer capacity is low and flexibilities are low. For this factor combination, computational requirements of the scheduling methods are also plotted in Figure 5.3. As expected, the computation time increases as the scheduling horizon is lengthened.

In summary the quality of the schedules generated is improved as the length of scheduling horizon (the extend of information usage) is increased in a static and deterministic environment. This shows that a quasi on-line scheduling scheme can utilize the potential trade-off between the quality of the schedule and the computational requirements. However, in a dynamic and stochastic environment in which most manufacturing system operate, the results may be completely different according to the level of interruptions in the system. In the next section, we study this problem by considering processing time variations and machine breakdowns.

5.3. Static and stochastic environment

5.3.1. Processing Time Variation

In a typical real manufacturing environment estimates of processing times are used in the scheduling process. However, actual processing times may be different than these estimates due to changing machining conditions and other factors. This uncertainty can easily degrade the quality of scheduling decisions made.

In this section, impacts of processing time variation on the scheduling decisions are investigated. For modeling processing time variations, it is assumed that estimated processing times come from a truncated normal distribution with a mean of the estimated value and a certain coefficient of variation (cv) to be specified. Actual processing times can be higher or lower than the estimated value with the same

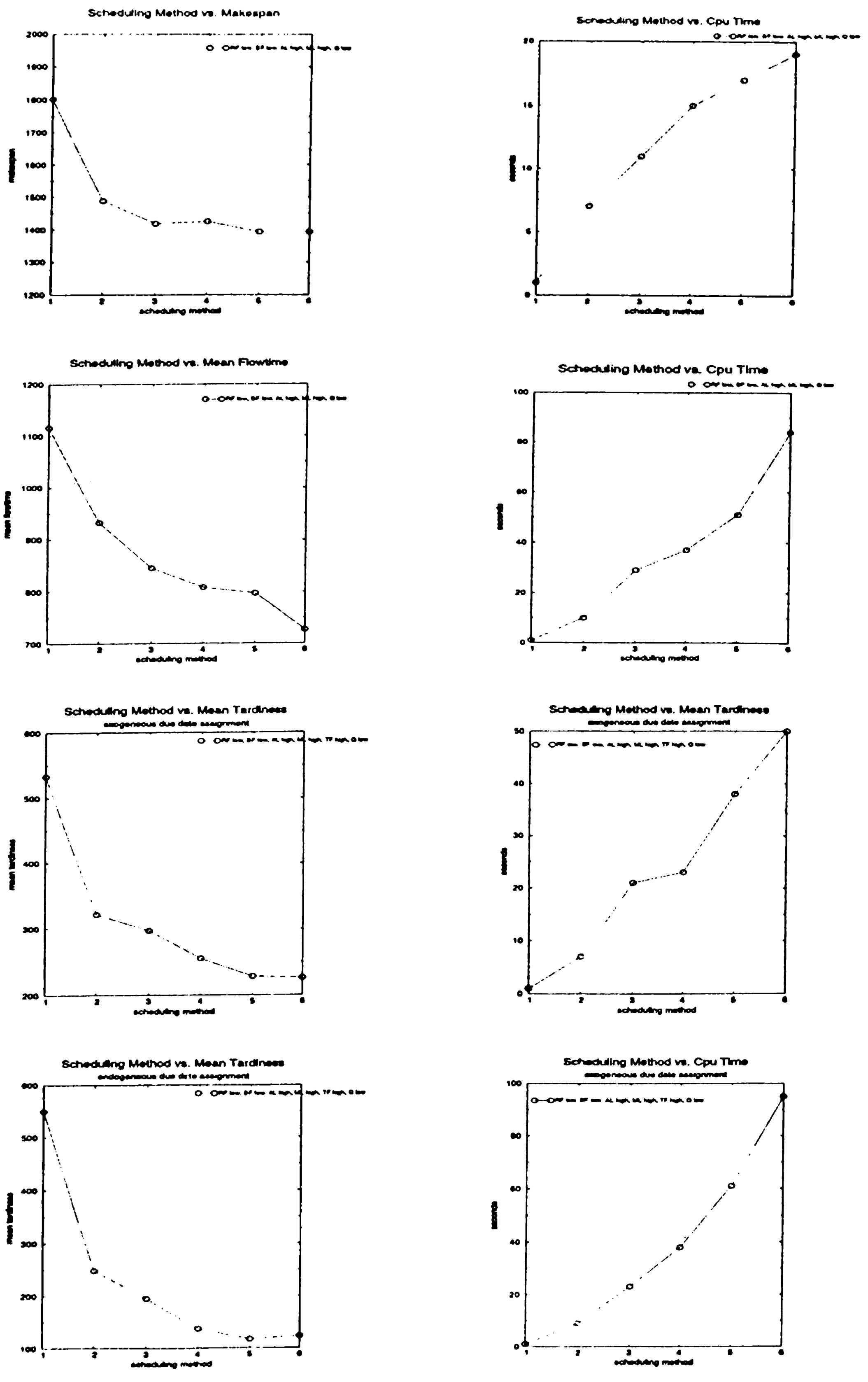


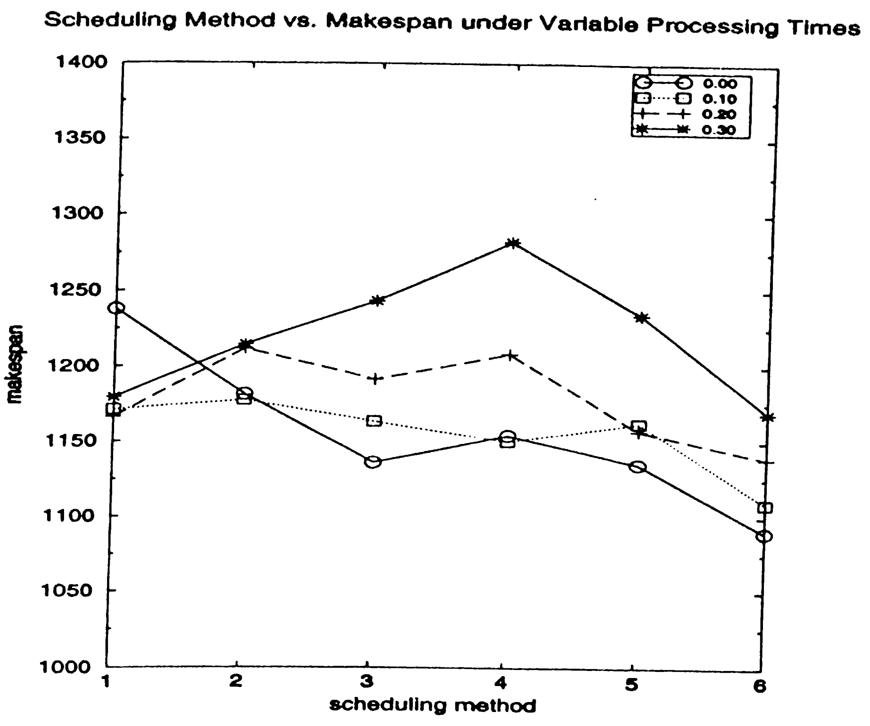
Figure 5.3 Cpu time requirements of scheduling schemes: deterministic and static environment

probability. The algorithm uses estimated operation times for generating schedules. However, during execution of schedules by the simulation model realized processing times are used. In the experiments, levels of scheduling factors are fixed such that flexibilities are low, machine and AGV loads are high, tardiness factor is high, and queue capacity is high. The three different levels of coefficient of variation is set to the values of 0.1, 0.2 and 0.3 respectively.

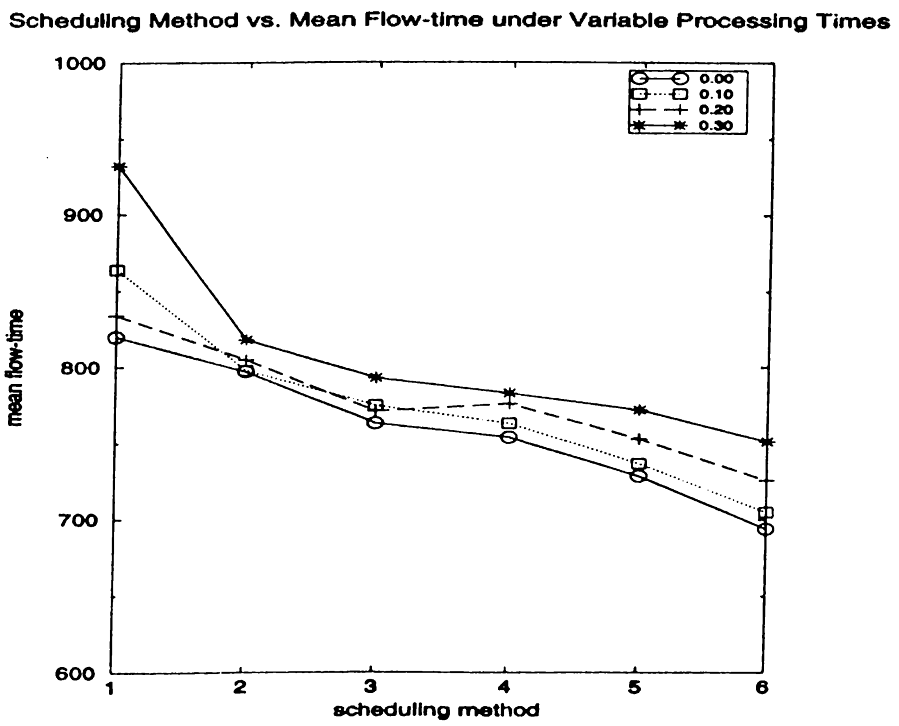
As can be seen in Figure 5.4 which displays the results for each scheduling criterion, the performance of scheduling methods detoriates as the level of processing time variability increases. Except for the makespan criterion, the relative ranking of the methods is preserved when compared with the deterministic environment. However, with the makespan criterion increasing the value of time window parameter leads to an increase in the makespan up to a certain point, beyond which makespan decreases again. Nevertheless, the off-line scheme is the best performer for all scheduling criteria.

The interesting behavior observed for the makespan criterion needs further discussion. This phenomenon can be explained by the high rate of jobs entering the system, which is a typical characteristic of schedules generated with minimum makespan objective. In general, machines operate with little slack to minimize the makespan and this leads to high resource utilization rates when compared to other performance measures. For that reason, schedules generated for the makespan criterion are more sensitive to variations of processing times. However, when the time window parameter is large enough, the total processing time of a machine will be close to its estimated value due to the negative and positive deviations in processing times. This smoothes out the adverse effects of variations of processing times in the long run.

However, if buffer capacity of machines is small, the smoothing effect can not be easily realized. In low buffer capacity cases, the synchronization and dependencies among machines and AGVs are more important. Specifically, delivery sequence of parts to machines will be invalidated by variable processing times. Consequently, a

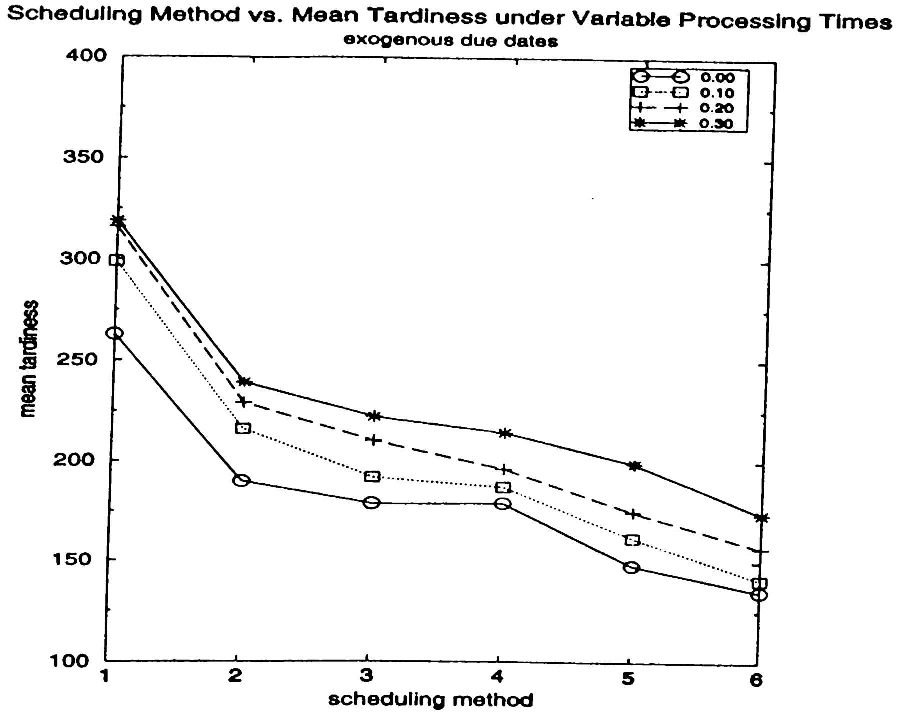


(a)

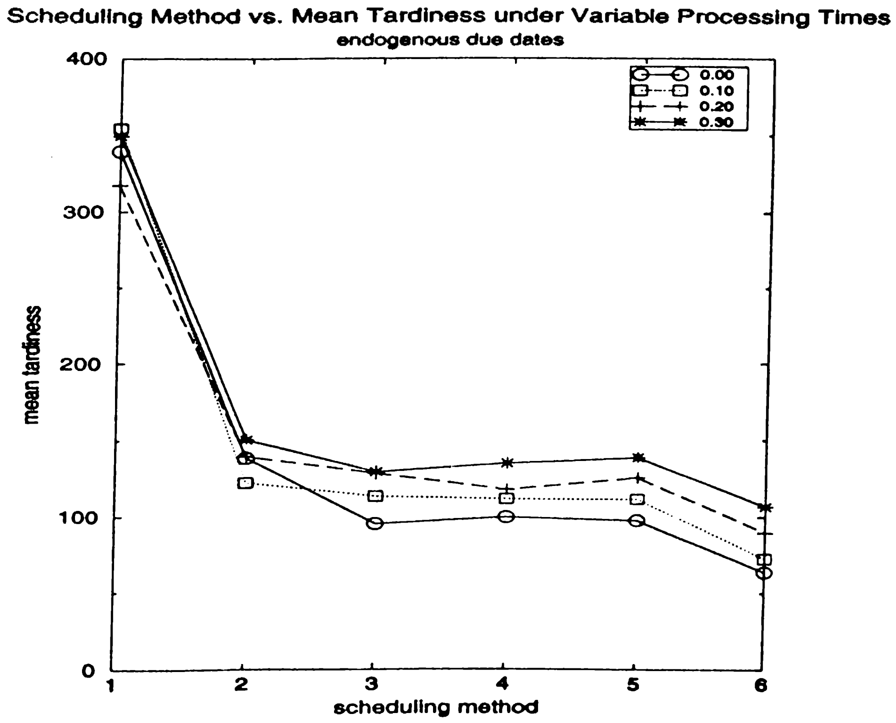


(b)

Figure 5.4 Comparision of scheduling schemes: varaible processing times in static environment



(c)



(d)

Figure 5.4 (Cont'd)

machine may have to wait longer for the next part in its processing sequence and can not compensate for the changes in processing times in the long run.

In order to confirm this conjecture, the same experiment is repeated for the makespan criterion with a small buffer capacity. As shown in Figure 5.5, the performance of the schedules degrade more with large values of time window parameter. However, increasing the time window up to some point decreases makespan. In this case, information usage becomes more important for the algorithm to avoid deadlocks in the low buffer capacity case. This also indicates the importance of a trade off between the amount of information usage and the degree of responsiveness for the makespan criterion.

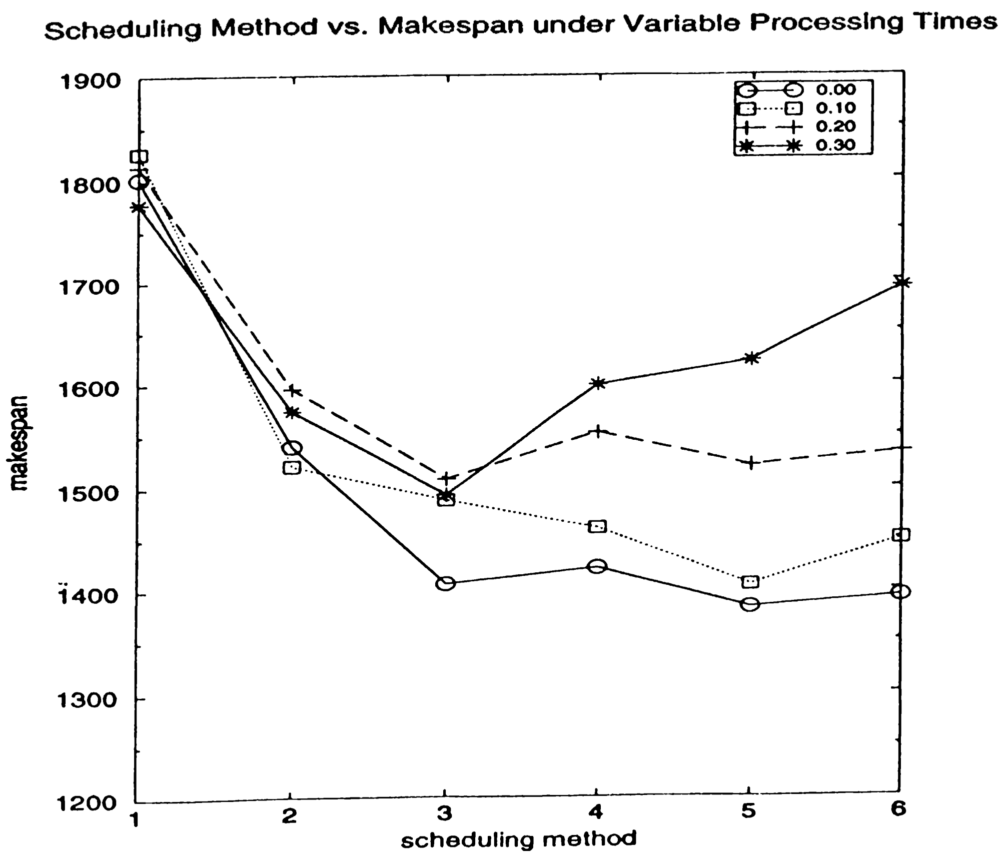


Figure 5.5 Scheduling schemes vs. makespan, tight queue capacity: variable processing times in static environment

5.3.2. Machine Breakdowns

Machine breakdowns are modeled by the busy time approach (Low and Kelton [18]). With this approach a random uptime is generated from a busy time distribution. The machine is considered as up until its total accumulated busy (processing) time reaches the end of this uptime. Then it fails for a random down time, after which an uptime will again be generated.

Low and Kelton [18] recommends that in absence of real data busy time distribution is most likely to be a gamma distribution with shape parameter alpha equal to 0.7 and scale parameter to be specified. The authors also propose a relationship between scale parameters and mean busy and down times, by which the model for machine breakdowns can be completely specified. In this framework, the level of machine breakdowns is measured by efficiency level which gives the long run ratio of a machines busy time to busy plus down time. The parameters and breakdown levels used in this study are depicted in Table 5.2.

Breakdown Level (efficiency)	Distribution Parameters (Gamma) Busy time	Down time

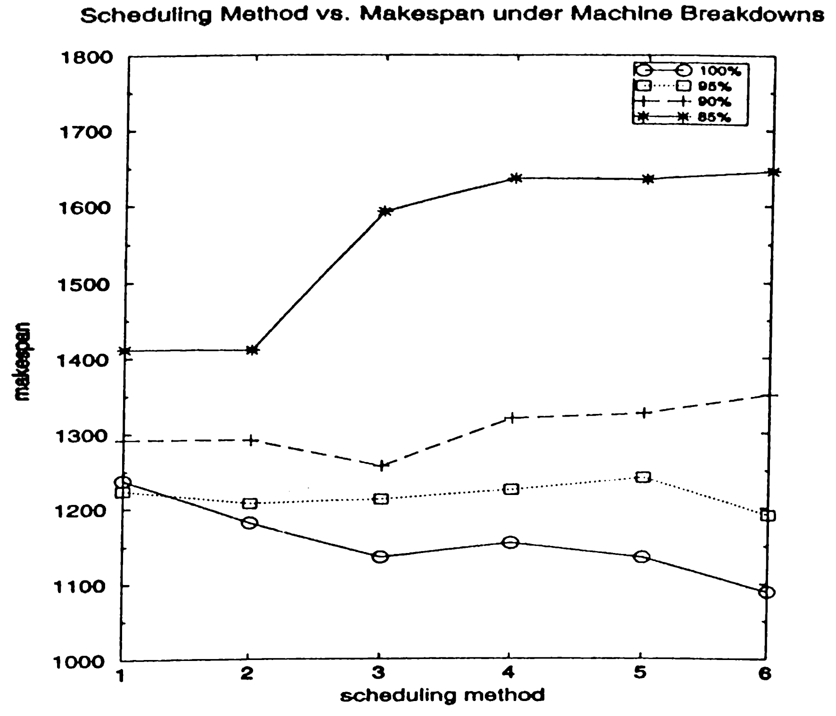
95 %	alpha =0.7
	beta=407
	mean=300
		alpha=1.4
		beta=10
		mean=15

90 %	alpha =0.7
	beta=450
	mean=300
		alpha =1.4
		beta=25
		mean=35

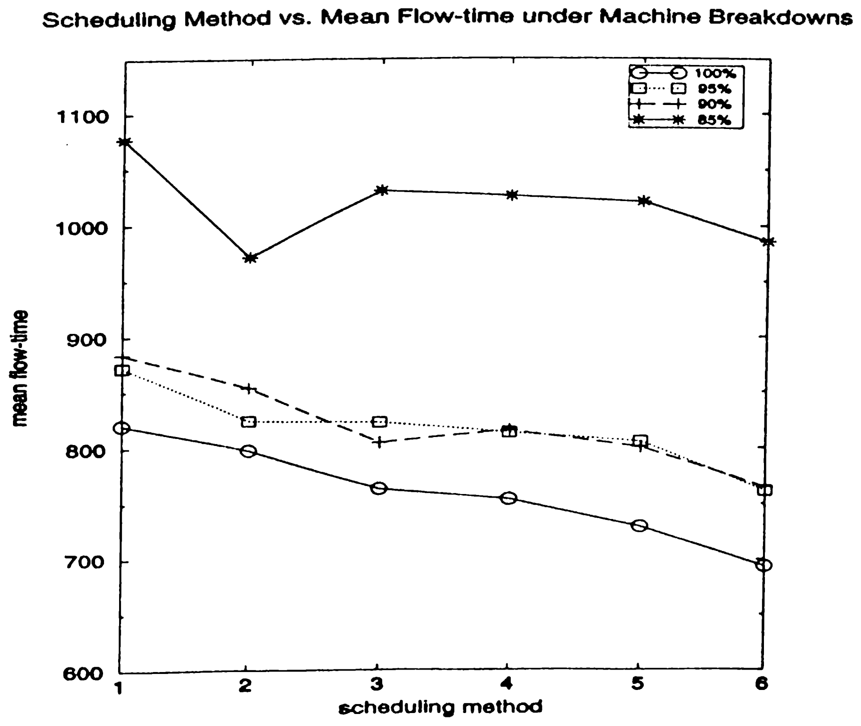
85 %	alpha =0.7
	beta=445
	mean=300
		alpha =1.4
		beta=40
		mean=55

Table 5.2. The parameters and breakdown levels used in the experiments.

Figure 5.6 shows the simulation results for each performance measure. For the makespan criterion, larger time windows tend to produce worse solutions as the efficiency level decreases. Especially, at the lowest efficiency level the makespan increases as the length of time window increases. as the length of time window

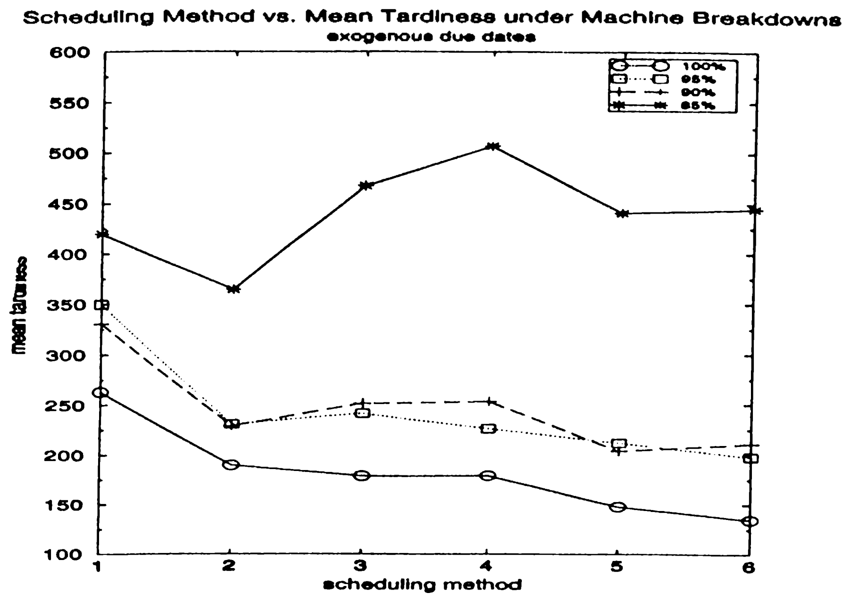


(a)

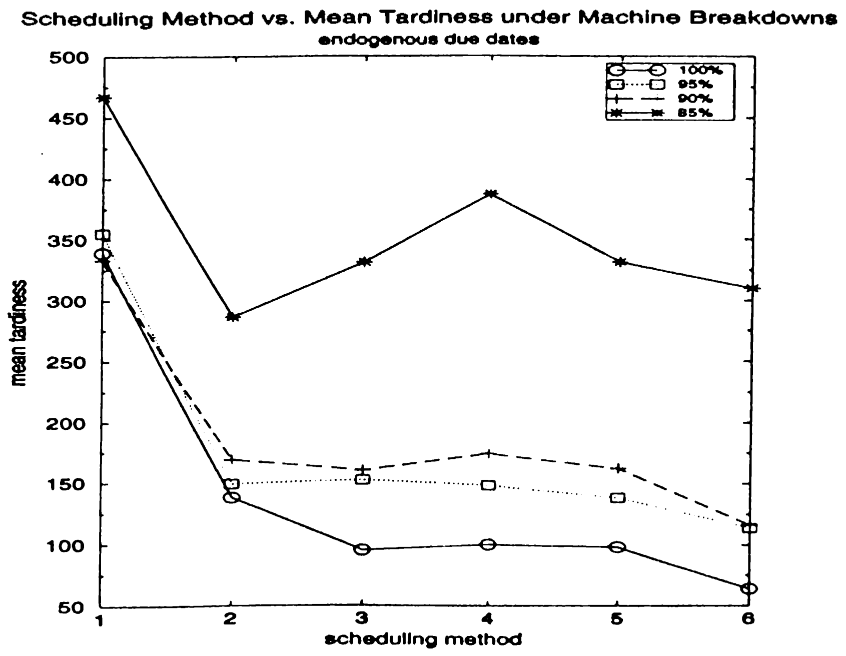


(b)

Figure 5.6 Comparision of scheduling schemes: machine breakdowns in static environment



(c)



(d)

Figure 5.6 (Cont'd)

increases. The best makespan values are obtained with the on-line method and the quasi on-line method with the smallest time window parameter value. This shows that the degree of responsiveness is more important than the information usage for the makespan criterion. For other performance measures the following behavior is observed. Increasing the value of time window parameter decreases the performance up to a certain point, beyond which any further increase improves the performance.

The behavior of performance curves for mean flowtime and mean tardiness criteria can again be explained by the amount of slack in machine and AGV schedules. Unlike the makespan criterion, there exists some slack in machine and AGV schedules generated by minimum mean flowtime and mean tardiness objectives. These slacks can absorb the effect of machine breakdowns to some extend when the value of time window parameter is large enough.

In summary, the results indicate that in all of the performance measures except makespan, off-line method performs better than the on-line. However, at lowest efficiency level, the best performing method turned out to be the quasi on-line scheme with the smallest time window value for all performance criteria.

5.4. Dynamic Environment

5.4.1. Deterministic case

Generally, in a dynamic environment an off-line scheduling scheme is used on a rolling horizon basis. That is at each scheduling point a static problem is generated by taking into account all unscheduled operations and other relevant information. This static problem is then solved entirely and the resulting schedule is implemented until a new job arrival, upon which a new schedule is generated. With this approach the dynamic problem is decomposed into a series of static problems which are implemented dynamically on a rolling basis. It has proved to give superior results than implementations of on-line scheduling schemes. This approach utilizes all available information at a scheduling point and has high degree of responsiveness because new job arrivals are considered immediately. On the other hand its computational requirements are rather high.

However, the information used at a scheduling point is not perfect because new job arrivals and other exceptions such as varying processing times and machine breakdowns that are not known in advance. Therefore, limiting the use of information at a scheduling point may provide competitive results with less computational requirements. This new approach can be realized with the implementation of the quasi on-line scheme.

In a dynamic environment the quasi on-line scheme can be applied in two ways. Firstly, it can be applied as in the static environment: that is, new job arrivals are ignored until a resource finishes its assigned tasks and triggers the scheduling process. Secondly, new job arrivals can also trigger scheduling. Throughout this section the first (second) approach will be referred to as Policy 1(Policy 2). With the first policy degree of responsiveness is determined by the value of the time window parameter. Moreover, the use of this parameter provides a job release mechanism. On the other hand, Policy 2 basically provides a trade-off between computational requirements and information usage at a scheduling point and its degree of responsiveness is the same as an on-line scheme in deterministic environment.

A set of experiments is performed in order to compare the quasi on-line scheme with different time window parameter values and the on-line scheme in a dynamic environment. Policy 1 and Policy 2 are also compared. The experimental factors are set such that flexibilities are low, machine and AGV load levels are high and the buffer capacity is loose. The levels of experimental factors are chosen as a basis to examine the effects of changing these levels in future studies. New arrivals are generated from an exponential distribution with mean 50. For every different scheduling scheme simulation runs are taken until the completion of 5000 jobs. In order to avoid warm up period 10 jobs are initially created and placed in the system at the beginning of each simulation run.

The on-line scheme uses a job release mechanism. The significance of a job release mechanism in an on-line scheme is demonstrated by Sabuncuoglu and Hommertzheim [31]. As shown in Figure 5.7, limiting the number of jobs on the shop floor with 15 gives best results in terms of mean flow-time. This value is used in the comparison with the quasi on-line scheme. Also note that, without using this mechanism the flowtime of jobs goes to infinity, that is the system cannot meet the demand.

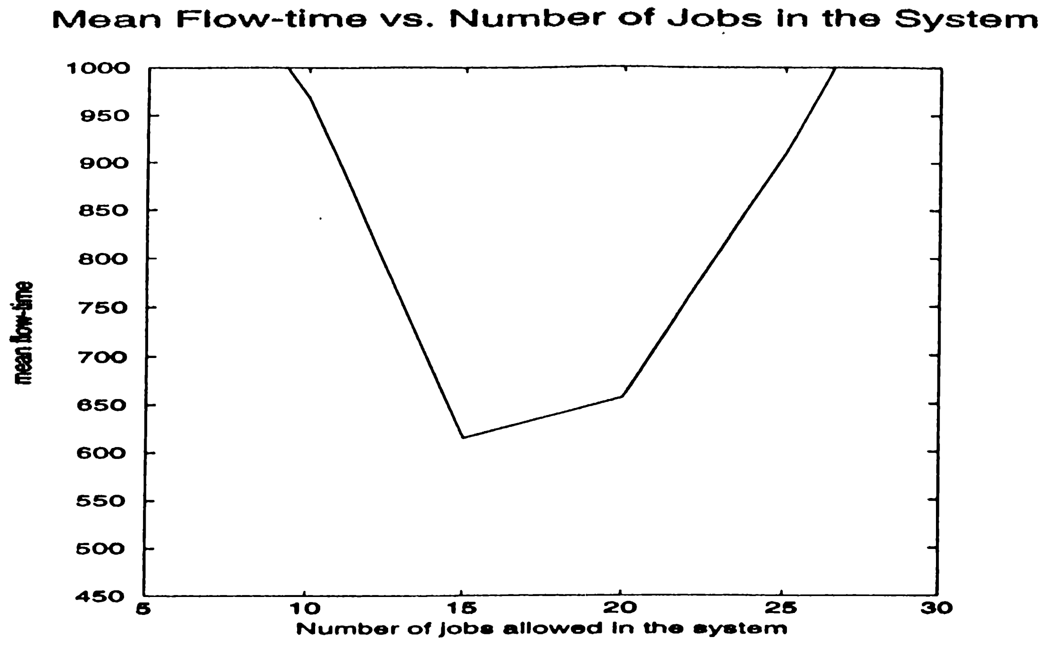


Figure 5.7. Effects of job release mechanism: on-line scheme

Such a mechanism is not used in the quasi on-line scheme. In experiments with mean flowtime criterion this was not needed. However, when the scheduling criterion is the mean tardiness, in most of the simulation runs the system could not meet the demand and simulation is terminated. This is due to the nature of the local evaluation function used in the scheduling algorithm. According to the algorithm, in mean tardiness case MDD rule determines the relative priority of jobs. Hence, new arriving jobs can immediately enter the system. In this section, experimental results with only mean flowtime criterion will be examined.

Figure 5.8 displays the simulation results. The scheduling method numbered as 1 in the figure corresponds to the on-line scheme. The other methods numbered 2 through 5, for both policies, correspond to the quasi on-line scheme with the values of time window parameter set to 2,4,6 and 8 times of the mean arrival rate (i.e. 100, 200, 300, 400). However, the last method numbered as 6 corresponds to the quasi on-line scheme with the time window parameter set to 10 times of the mean arrival rate for Policy 1, and to the off-line scheme (i.e. time window parameter set to infinite) for Policy 2. In Figure 5.9 cpu requirements of scheduling schemes with both policies are plotted.

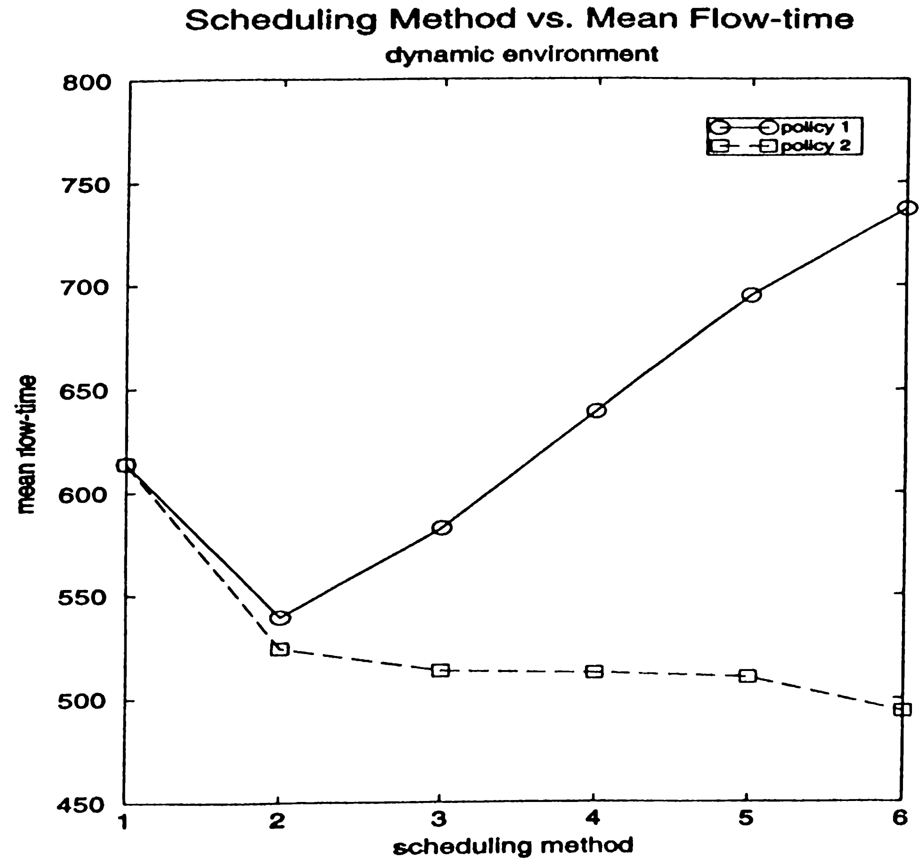


Figure 5.8. Comparision of scheduling schemes: deterministic and dynamic environment

As can be observed from the Figure 5.8, increasing the value of time window parameter in policy 1 results in a continuous decline in the performance of the system. This indicates that, responding to new arrivals is crucial in a dynamic system. However, a small time window value still performs better than the on-line scheme for policy 1. For policy 2, increasing the value of time window parameter improves the performance continuously. The application of off-line scheme gives best results in spite of its high computational requirements. This suggests that the off-line scheme achieves a better synchronization and coordination of resources and this is preserved even when it is applied on a rolling basis. The most important increase is observed when switching from on-line scheme to the quasi on-line with smallest time window value.

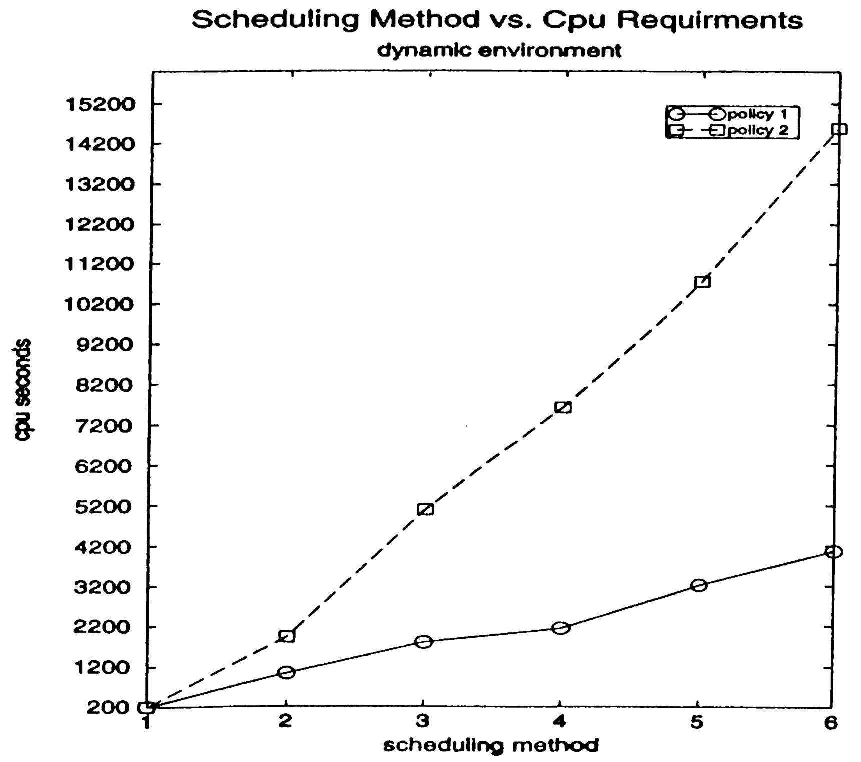


Figure 5.9. Cpu time requirements of scheduling schemes: deterministic and dynamic environment

Due to its relatively better performances, the quasi on-line scheme will be used according to the Policy 2 in the rest of this chapter.

5.4.2. Stochastic Environment

5.4.2.1. Variable processing times

In this section effects of variable processing times on the scheduling schemes are investigated in a dynamic environment. The experimental conditions mentioned in the previous section are adapted except that the three levels of processing time variations which are mentioned in section 5.4.1 are also added.

The simulation results are displayed in Figure 5.10. With all scheduling methods the performance deteriorates as the variability of processing times increase. The relative performance of each scheduling method is preserved when compared to deterministic case. Again the off-line scheme outperforms all other methods. The largest performance increases are observed when switching from scheduling method 1 to method 2 and from scheduling method 5 to 6. These observations suggest that information usage at a scheduling point is crucial even though the information is not perfect due to processing time variability.

5.4.2.2. Machine breakdowns

In this section the impact of machine breakdowns on the scheduling schemes is analyzed. The same experimental conditions defined in section 5.5.1. are adapted. Furthermore, the three machine efficiency levels described in section 5.4.2 are used.

The simulation results are depicted in figure 5.11. Similar observations are also made in this case. Additionally, at the lowest efficiency level, information usage becomes even more critical. Also notice that with both methods 1 and 2 the system cannot meet the demands and the flowtime of jobs goes to infinity.

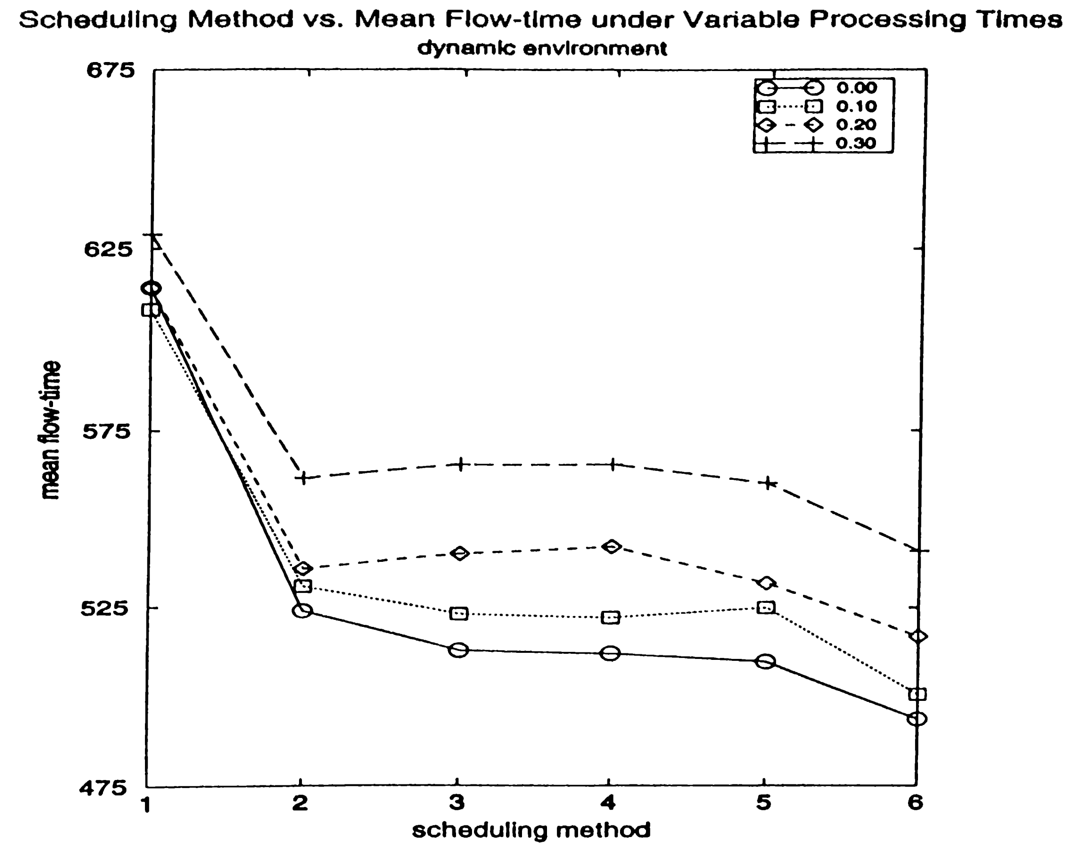


Figure 5.10 Comparision of scheduling schemes: varaible processing times in dynamic environment

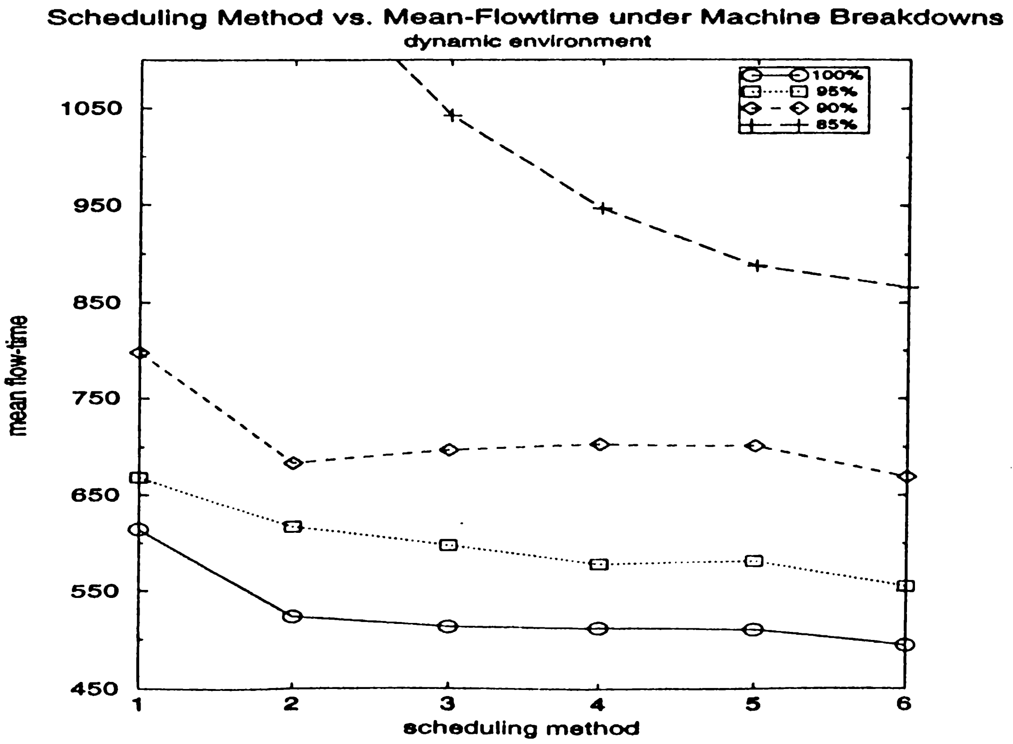


Figure 5.11 Comparision of scheduling schemes: machine breakdowns in dynamic environment

CHAPTER 6

CONCLUSION

In this thesis, essentially three issues are addressed in detail: comparison of on-line and off-line scheduling schemes in various operating environments; development and analysis of a new scheduling scheme called *quasi on-line* that makes a trade-off between superior and inferior characteristics of on-line and off-line schemes; and examination of effects of scheduling factors on FMS performance.

The first step in the course of this research was to develop a scheduling algorithm that considers a wide variety of scheduling related features of an FMS. The proposed algorithm generates a partial or complete machine and AGV schedule considering limited buffer capacities and routing and sequence flexibilities, for a given scheduling period. The scheduling period is specified by a parameter called a time window. This parameter determines both the number of scheduling decisions made at a time, thus the length of the scheduling period and the extend of information utilized at a decision point. The algorithm is designed in such a way that as the value of time window parameter increases the scheduling decisions are made in a more coordinated and integrated manner. The off-line scheduling scheme used in this study corresponds to a version of the algorithm with an infinite value of the time window parameter. Whereas the quasi on-line scheme utilizes the algorithm at each decision point with small parameter values. These decision points are triggered when a resource finishes its assigned tasks. With this scheme the scheduling problem is decomposed in time dimension and solved iteratively by concatenating partial solutions generated at each decision point. This approach has the following benefits. First the computational burden of the solution procedure reduces, second changing the frequency of scheduling points by varying the time window acts as a control policy by adjusting the degree of responsiveness of the scheduling scheme to unexpected events.

The operating environment of a manufacturing system is differentiated in two dimensions: demand pattern and uncertainty in the system. According to demand pattern a system may operate in a static environment (i.e. demand arrive periodically) or in a dynamic environment (i.e. demand arrive continuously over time). On the other hand, if unexpected events other than new arrivals occur over time (e.g. machine breakdowns, considerable variations in processing times etc.) the system operates in a stochastic environment as opposed to a deterministic environment. In this research, effectiveness of scheduling schemes under all these four different environments are investigated by carrying out a simulation study.

The second step of this research was to develop and implement a simulation based scheduling system in order to implement different scheduling schemes in different operating environments. The system is designed to have three components: the scheduler, the simulation model and the controller. The scheduler subsystem consists of the scheduling algorithm developed in this study and several dispatch rules to be used in the on-line scheme. The simulation model executes the scheduling decisions made by the scheduler. The controller acts as an interface with the simulation model and the scheduling algorithms and is the most crucial part of the system especially for the implementation of the quasi on-line and the off-line schemes.

In the experiments different experimental conditions are obtained by varying machine load level, AGV load level, buffer capacity level, sequence flexibility level, routing flexibility level and due date tightness. The performance measures were makespan, mean flowtime and mean tardiness with two types of due date assignment methods (i.e. exogenous and endogenous).

In deterministic and static environment, the off-line scheme performed the best among others. A more detailed comparison between on-line and off-line schemes showed that the performance difference between these schemes becomes more significant at high machine load and/or tight buffer capacity and/or high routing flexibility for all scheduling criteria and at tight due dates as well for tardiness criterion. This result suggests that information usage is more crucial when system resources are tight or there are alternative machines for operations. Also, the performance difference was relatively higher for mean flowtime and mean tardiness criteria than for makespan criterion.

The quasi on-line scheme also performs better than the on-line scheme in deterministic and static environment. Further examination of different versions of the quasi on-line scheme (obtained with different values of time window parameter) revealed that, as the length of the time window increases, so does the quality of the schedule. This situation is most apparent when resources are tight and flexibility levels are low. Therefore, in a static and deterministic environment the quasi on-line schemes makes a trade-off between quality of schedule and computation time.

In stochastic and static environment, however, the determination of the best performing scheduling scheme is effected by the levels of experimental factors and the type of the stochastic events. Another important factor is the performance criterion. The performance of the off-line scheme that generate schedules with the minimum mean flowtime and tardiness objectives is rather robust to unexpected events. This can be attributed to the slack present in the AGV and machine schedules. This slack is, to some extend, used to absorb the deviations in the schedule due to unexpected events. Nevertheless, in all cases either the quasi on-line or the off-line schemes performed the best.

In dynamic environment, in order to implement the quasi on-line and off-line schemes two approaches are examined. With the first approach new arrivals are ignored until a scheduling point, whereas they triggered scheduling in the second approach. It turned out that the second approach performed better than the first one. This shows that in a dynamic environment responsiveness to new arrivals is crucial in scheduling decisions. Another important issue was the use of a job release mechanism. The on-line scheme could not meet the demand without controlling the entry of jobs into the shop floor. Hence, it employed a job release mechanism. The quasi on-line and off-line schemes did not use such a mechanism. As a result, for mean tardiness criteria the system could not meet the demand. These results point out the importance of a job release mechanism in a dynamic environment.

In dynamic and stochastic environment the second approach is used to implement the quasi on-line and off-line schemes. Again the off-line scheme performed better than the others. This indicates that, global coordination of resources even with using imperfect information is important and beneficial.

For future research the experiments already conducted in stochastic and static environment can be extended to cover all other combinations of experimental factors. For dynamic environment different job release mechanisms and their effects on scheduling schemes can be investigated. Also, different implementation approaches for the quasi on-line and the off-line schemes in dynamic and stochastic environment can be investigated. Finally, it may be interesting to see an application of a job release mechanism in a static environment.

BIBLIOGRAPHY

[1] Aanen, E., Gaalman, G.J., and Nawijn, W.M., "A Scheduling Approach for a Flexible Manufacturing System", *International Journal of Production Research*, 31(10), 2369-2385, 1993.

[2] Blackstone, J. H., Phillips, D. T., and Hogg, G. L., "A state-of-the-art survey of dispatching rules for manufactruing jobshop operations", *International Journal of Production Research*, 20(1), 27-45, 1982.

[3] Chandra, J., and Talavage, J., "Intelligent dispatching for flexible manufacturing", *International Journal of Production Research*, 29(11), 2259-2278, 1991.

[4] Chang, Y.L., Sullivan, R.S., and Bagchi, U., "Experimental investigation of real time scheduling in flexible manufacturing systems", *Annals of Operations Research* 3, 355-377, 1985.

[5] Chang, Y.L., Matsua, H., and Sullivan, R., "A bottleneck based beam search for job scheduling in a flexible manufacturing system", *International Journal of Production Research*, 27(11), 1949-1963, 1989.

[6] Chang, Y.L., and Sullivan, R.S, "Schedule generation in a dynamic jobshop", *International Journal of Production Research*, 28(1), 65-74, 1990.

[7] De, S., "A knowledge based approach to scheduling in an FMS", *Annals of Operations Research*, 12, 109-134, 1988.

[8] De, S., and Lee, A., "Flexible manufacturing system (FMS) scheduling using filtered beam search", *Journal of Intelligent Manufacturing*, 1, 165-183, 1990.

[9] Egbelu, P.J., and Tanchoco, M.A., "Chracterization of automatic guided vehicle dispatching rules", *International Journal of Production Research*, 22, 359-375, 1984.

[10] Fox, M.S., "Constraint directed search: A case study of jobshop scheduling", *Ph.D. Thesis, Carnegie Mellon University*, U.S.A., 1983.

[11] Groover, M. P., "*Automation, Production Systems and Computer Integrated Manufacturing*", (New York: Prentice-Hall), 1987.

[12] Harmonosky, C.M. and Robohn, S.F., "Real-time Scheduling in Computer Integrated Manufacturing: A Review of Recent Research.", *Int. J. Computer Integrated Manufacturing*, 4(6), 331-340, 1991.

[13] Harmonosky, C., "Analysis of Two Key Issues for Using Simulation for Real-Time Production Control.", *In Proceedings of the 2nd Industrial Engineering Research Conference*, (L.A., California, May 26-28). IIE, Norcross, Georgia, 41-45, 1993.

[14] Hutchison, J., "Current issues concerning FMS scheduling", *OMEGA Int. J. of Mgmt. Sci.*, 19(6), 529-537, 1991.

[15] Hutchison, J., Leong, K., Synder, D., and Ward, F., "Scheduling approaches for random jobshop flexible manufacturing systems", *International Journal of Production Research*, 29(3), 1053-1067, 1991.

[16] Ishii, N., and Talavage, J. J., "A transient-based real-time scheduling algorithm in FMS", *International Journal of Production Research*, 29(12), 2501-2520, 1991.

[17] Kiran, A. S., and Smith, M. L., "Simulation studies in jobshop scheduling-performance of priority rules", *Computers and Industrial Engineering*, 8(2), 95-105, 1984.

[18] Low, A. M., and Kelton, W. D., "*Simulation Modelling and Analysis*", McGraw-Hill, 1992.

[19] Lowerre, B.T., "The HARPY speech recognition system", *Ph.D. Thesis, Carnegie Mellon University*, U.S.A., 1976.

[20] Manivannan, S. and Banks, J. "Design of a Knowledge-based On-line Simulation System to Control a Manufacturing Shop Floor." *IIE Transactions*, 21, 72-83, 1992.

[21] Montazeri, M., and Wassenhove, L.N., "Analysis of scheduling rules for an FMS", *International Journal of Production Research*, 28(4), 785-802, 1990.

[22] Morton, T.E., and Pentico, D.W., "*Heuristic Scheduling Systems with Applications to Production Systems and Project Management*", John Wiley and Sons, New York, 1993.

[23] Mukhopadhyay, S.K., Bibekananda, M., and Garg, S., "Heuristic solution to the scheduling problems in flexible manufacturing system", *International Journal of Production Research*, 29(10), 2003-2024, 1991.

[24] Nof, S., Barash, M., and Solberg, J., "Operational control of item flow in versatile manufacturing systems", *International Journal of Production Research*, 17(5), 479-489, 1979.

[25] Ow, P.S, and Morton, T.E., "Filtered beam search in scheduling", *International Journal of Production Research*, 26(1), 35-62, 1988,.

[26] Panwalkar, S. S., and Iskander, W., "A Survey of Scheduling Rules", *Operations research*, 25(1), 45-61, 1977.

[27] Rachamadugu, R. and Schriber, T.J., "Performance of dispatch rules under perfect sequence flexibility", *Proceedings of the Winter Simulation Conference*, 653-658, 1990.

[28] Rachamadugu, R. and Stecke, K., "Classification and review of FMS scheduling procedures", *Production Planning & Control*, 5(1), 2-20, 1994.

[29] Raman N., "A Survey of the Literature on Production Scheduling as it Pertains to Flexible Manufacturing Systems", *Technical Report No: NBS/GCR-85/499, Graduate School of Business Administration, The University of Michigan*, 1985.

[30] Raman N., Talbot, F.B., and Racmahadugu, R.V., "Due date based scheduling in a general flexible manufacturing system", *Journal of Operations Management*, 8(2),115-132, 1989.

[31] Sabuncuoglu, I and Hommertzheim, D.L., "An investigation of machine and AGV scheduling rules in an FMS", *In Proceedings of the Third ORSA/TIMS Conference on Flexible Manufacturing Systems: Operations Research Models and Applications, edited by K.E. Stecke and R. Suri*, 261-266, 1989.

[32] Sabuncuoglu, I and Hommertzheim, D.L., "Dynamic dispatching algorithm for scheduling machine and AGVs in a Flexible Manufacturing System", *International Journal of Production Research*, 30(5), 1059-1080, 1992.

[33] Sabuncuoglu, I and Hommertzheim, D.L., "Experimental investigation of FMS machine and AGV scheduling rules against the mean flow-time criterion", *International Journal of Production Research*, 30(7), 1617-1635, 1992.

[34] Sabuncuoglu, I and Hommertzheim, D.L., "Experimental investigation of FMS due-date scheduling problem: evaluation of machine and AGV scheduling rules", *International Journal of Flexible Manufacturing Systems*, 5(4), 301-324, 1993.

[35] Shanthikumar, J.G. and Sargent, R.G. "A Unifying View of Hybrid Simulation/Analytic Models and Modeling." *Operations Research*, 31, 1030-1052, 1983.

[36] Shaw, M. J., Park, S. and Raman, N., "Intelligent Scheduling with Machine learning Capabilities: The Induction of Scheduling Knowledge", *IIE Transactions*, 24(2), 156-168, 1992.

[37] Sriskandarajah, C., Sethi, S.P., and Ladet, P., "Scheduling methods for a class of Flexible Manufacturing Systems", *Annals of Operations Research*, 17, 139-162, 1989.

[38] Stecke, K. E., "Design, Planning, Scheduling, and Control Problems of Flexible Manufacturing Systems", *Annals of Operations Research*, 3, 3-12, 1985.

[39] Ulusoy, G., and Bilge, U., "Simultaneous scheduling of machines and automated guided vehicles", *International Journal of Production Research*, 31(12), 2857-2873, 1994.

[40] Wu, S.D. and Wsyk, R.A., "An application of discrete-event simulation to on-line control and scheduling in flexible manufacturing.", *International Journal of Production Research*, 27(9), 1603-1623, 1989.

[41] Yamamoto, M., and Nof, S.Y., "Scheduling/rescheduling in the manufacturing operating system environment", *International Journal of Production Research*, 23(4), 705-722, 1985.

[42] Yao, D. D., and Pei, F. F., "Flexible Parts Routing in Manufacturing Systems", *III Transactions*, 22(1), 48-55, 1990.

[43] Zanakis, S.H., Evans, J.R., and Vazacopoulos, A.A., "Heuristic methods and applications: A categorized survey", *European Journal of Operations Research*, 43, 88-110, 1989.