State Dependent Priority Rules for Scheduling

Ari P. J. Vepsalainen

CMU-RI-TR-84-19

Graduate School of Industrial Administration
and
The Robotics Institute
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213

July 1984

Copyright © 1984 Carnegie-Mellon University

This research was sponsored, in part, by the Air Force Office of Scientific Research under contract F49620-82-K-00017, and the Westinghouse Corporation.

Table of Contents

1. Introduction and Summary 1
1.1 Introduction 1
1.2 Summary 4
2. Framework for State Dependent Priority Rules 11
2.1 Introduction 11
2.1.1 Weighted Tardiness Problems 11
2.1.2 Review of Priority Dispatching Rules 12
2.2 A Framework for Analysis of State Dependent Priority Rules 15
2.2.1 The Definition of State Dependent Priority Rules 15
2.2.2 New Distinctions of State Information in a Priority Index 16
2.3 An Adaptive Priority Rule for Single Machine Weighted Tardiness Scheduling 18
2.3.1 Some Properties of Apparent Urgency Priority Indexes 18
2.3.2 The Adaptive Apparent Urgency Rule 19
2.3.3 The Effects of Errors in Processing Time Estimates 23
2.4 Conclusions 26
3. Scheduling in Proportionate Flow Shops 27
3.1 Introduction 27
3.2 Permutation Schedules for the Proportionate Flow Shops 28
3.3 Schedules with No Job-passing 34
3.4 Schedules with no Job-waiting 35
3.5 Conclusion 36
4. Lead Time Iteration in Flow Shop Scheduling 37
4.1 Introduction and Summary 37
4.1.1 The Need for Lead Time Estimation 37
4.1.2 Related Heuristic Approaches 38
4.1.3 Chapter Summary 40
4.2 Lead Time Estimation in Flow Shops 41
4.2.1 Due Date Setting in Proportionate Flow Shops with Unit Jobs 41
4.2.2 Setting Operation Due Dates for the Apparent Urgency Rule 44
4.3 A Computational Study of Lead Time Estimation 46
4.3.1 The Scheme for Lead Time Iteration 47
4.3.2 The Design of the Simulation Experiment 48
4.3.3 Weighted Tardiness Performance of the Rules 50
4.3.4 Other Criteria of Performance 53
4.4 Conclusions 56
5. Slack Evaluation for Priority Dispatching in Dynamic Job Shops 57
5.1 Introduction 57
5.2 Estimation of Job Waiting Times to Evaluate Slack Priorities 58
5.2.1 Slack Evaluation in the Apparent Urgency and CoverT Rules 58
5.2.2 Derivation of Priority-Based Waiting Times 60
5.2.3 Waiting Time Estimation in a Dynamic Job Shop 61
5.3 Experimental Design for Dynamic Job Shops 65
5.3.1 The Job Shop Problems 65
5.3.2 Measures of Performance 66
5.3.3 Rules Tested 66

5.4 The Results of the Experiment 67
5.4.1 Weighted Tardiness and Portion of Tardy Jobs 67
5.4.2 Other Measures of Performance 69
5.5 Conclusions 73
6. Coordination of Job Priorities via Direct Load Information 75
6.1 Introduction 75
6.2 Load Estimates as Priority Coordination Parameters 77
6.2.1 Different Mechanisms of Priority Coordination 77
6.2.2 The "Self-Adjusting" AU Priority Index 78
6.2.3 Estimation of State Indicators 80
6.3 Experiments in Dynamic Job Shops 81
6.3.1 Job Shop Problem and Rules Tested 81
6.3.2 The Results of the Experiment 83
6.4 Conclusions 86
7. Conclusions and Future Research 87
7.1 Conclusions 87
7.1.1 Introduction 87
7.1.2 The Framework for Information in Priority Rules 87
7.1.3 Application of State Dependent Priority Rules 90
7.2 Directions for Future Research 92
7.2.1 Composite Priority Indexes 92
7.2.2 The Lot Scheduling Problem 94
7.2.3 Incentive-Compatible Priority Pricing Mechanisms 95
7.2.4 Diagnostic Analysis for Scheduling Expert Systems 97
8. References 99

List of Figures

Figure 1-1: A framework for classifying dynamic priority dispatching rules according to the information used in the priority index. The matrix shows some existing rules and the chapters discussing the new enhancements. 3

Figure 2-1: The classification of the priority dispatching rules according to the information used in the priority index 17

Figure 2-2: The optimal schedule of the example as a Gantt chart. 20

Figure 4-1: Illustration of the case in which both jobs are tardy even if started first. 45

Figure 5-1: The comparison of the CoverT, CoverT-2 and AU priority indexes for a job which has two operations remaining. 59

Figure 5-2: Three sets of a job's expected waiting time graphs as functions of the priority index π : 1) The approximation of the expected waiting time 2) The priority-based estimation function $W = \beta\bar{w}/\pi(\gamma) + 0.5$, 3) The priority based waiting time estimates, 62

Figure 5-3: The weighted tardiness for the different levels of load averaged over the three shop types. 70

Figure 5-4: The percentage of tardy jobs for the different levels of load averaged over the three shop types. 71

Figure 5-5: An example of combined inventory, tardiness and rush order costs for different load levels averaged over the three shop types. 74

Figure 7-1: The classification of the state dependent dispatching rules according to the information used in the priority index. The enhancements of the basic look-ahead rules AU and CoverT are shown along with some previous rules and the pertinent chapters of the thesis. 88

List of Tables

Table 2-1: The improvement of the performance of the new Apparent Urgency rule using the look-ahead adaptation procedure for the 20-job problems. 21

Table 2-2: The improvement of the performance of the Apparent Urgency rule using the look-ahead adaptation procedure for the 30-job problems. The improvement with a pairwise interchange procedure is given below in parenthesis for comparison. 22

Table 2-3: The effect of errors in processing time estimates on the normalized weighted tardiness performance of the WSPT, EDD and AU rules. 24

Table 2-4: The performance of the rules in terms of maximal weighted tardiness and the number of tardy jobs out of 20 (in parenthesis). 25

Table 4-1: Normalized weighted tardiness for different tardiness levels and ranges of due dates. Three shop layouts, 480 problems with 4 machines and 20 jobs. 51

Table 4-2: Normalized weighted tardiness for different tardiness levels and ranges of due dates. Three shop layouts, 480 problems with 4 machines and 60 jobs. 51

Table 4-3: Normalized weighted tardiness for different tardiness levels and ranges of due dates. Two shop layouts, 320 problems with 8 machines and 60 jobs. 52

Table 4-4: Rule performance in terms of portion of tardy jobs, maximal weighted tardiness, work in process (WIP) and work in shop (WIS) inventories for 4 machines and 20 jobs (three shop layouts, 480 problems). 54

Table 4-5: Rule performance in terms of portion of tardy jobs, maximal weighted tardiness, work in process (WIP) and work in shop (WIS) inventories for 4 machines and 60 jobs (three shop layouts, 480 problems). 54

Table 4-6: Rule performance in terms of portion of tardy jobs, maximal weighted tardiness, work in process (WIP) and work in shop (WIS) inventories for 8 machines and 60 jobs (two shop layouts, 320 problems). 55

Table 5-1: The assignment of the look-ahead parameter κ and the lead time estimation parameter β for the AU and CoverT rules corresponding to the load in the shop in terms of the expected capacity utilization. 67

Table 5-2: The normalized weighted tardiness averaged for the three shop types. Slack due dates. The average portion of tardy jobs is shown in parenthesis. 68

Table 5-3: The normalized weighted tardiness averaged for the three shop types. Tight due dates. The average portion of tardy jobs is shown in parenthesis. 69

Table 5-4: The normalized WorkIn-Process (WIP) inventory averaged for the three shop types, slack due dates. The Work-In-Shop (WIS) is shown as well. 72

Table 5-5: The normalized Work-In-Process (WIP) inventory averaged for the three shop types, tight due dates. The Work-In-Shop (WIS) is shown as well. 72

Table 6-1: The performance of the self-adjusting rules in terms of normalized weighted tardiness with slack due dates, averaged for the three shop types. The average percentage of tardy jobs is shown in parenthesis. 84

Table 6-2: The performance of the self-adjusting rules in terms of normalized weighted tardiness with tight due dates, averaged for the three shop types. The average percentage of tardy jobs is shown in parenthesis. 85

Abstract

The purpose of this thesis is to enhance the priority setting procedures for job shop scheduling systems. The new state dependent priority rules extend the concept of a myopic dispatching heuristic by allowing a wide choice of forecasting and planning horizons and by encompassing indirect or direct load information, even performance feedback, while maintaining the flexibility and robustness of the dispatching approach. Preliminary results are proven in the special case of proportionate flow shops with pre-emption. Many optimal rules for lateness and tardiness problems are extended from the single machine case to flow shops. Appropriate lead time estimation used in setting operation due dates can be shown to guarantee the achievement of a global optimum when applying a myopic rule locally. In more general job shop environments, we study scheduling with due dates when jobs have different tardiness penalties. Advanced slack evaluation methods have been developed for our Apparent Urgency rule and for the modified CoverT rule. First, waiting line analysis furnishes the use of indirect load information, such as the distribution of the jobs' weights and processing times, in assigning static priority-based waiting time estimates for each operation. Second, the waiting time estimation and look-ahead parameters of the rules are further adjusted on the basis of direct, periodically updated state information, such as the anticipated queue lengths in the shop. Third, an iterative scheme is used to revise new lead time estimates based on the jobs' realized waiting times in successive schedules. This lead time iteration provides also feedback from the performance of the rule for the coordination of the priority assignments. The latter two enhancements of the myopic dispatching rules are implemented using rolling forecasting and planning horizons.

In our large scale tests in static flow shops and dynamic job shops, the Apparent Urgency and CoverT rules with the static lead time estimates surpassed the competing rules in weighted tardiness performance. The new priority-based lead time estimates improved both rules vis-a-vis the conventional method, making CoverT best in heavily loaded shops. The periodic adjustment of the slack evaluation parameters did not, however, lead to a consistent improvement of the rules' performance. The coordination of the Apparent Urgency rule, achieved through the lead time iteration procedure, proved paramount for a superior performance in weighted tardiness and combined schedule costs. The experiment with continuously updated information, the average anticipated urgency of the jobs in the machine queues, as a part of the AU priority index was not as successful. This probing of job's relative priority on the next machine should prevent temporary congestion, but it was ineffective in long run. The robustness of the new rules was tested against errors in the processing time estimates and also in terms of other criteria such as the number of tardy jobs, work-in-process inventory, and maximal tardiness cost with extremely good results.

Further extensions of state dependent rules are specified to multi-objective, multi-resource scheduling using composite priority indexes and to dynamic lot sizing problems in closed job shops. Applications in developing incentive-compatible priority rules and in supporting diagnostic routines of scheduling expert systems are discussed.

Acknowledgements

I wish to express my appreciation to the individuals and institutions who made this thesis research possible. Special thanks are due to Professor Thomas Morton, my mentor and friend. Tom's insight to operations management provided the starting point for this research and his discipline ensured its completion. Professors Charles Kriebel and Herbert Simon contributed invaluable ideas, criticism and mind-expanding discussions throughout the process of my professional training. Dr. Mark Fox offered his vision and other resources of the Intelligent Systems Laboratory to challenge the conventional design of scheduling systems. My stay at Carnegie-Mellon University as a student and as a faculty member was enjoyable in the inspiring atmosphere created by many friends and colleagues at the Graduate School of Industrial Administration and the Robotics Institute. I am ever so grateful to my parents for continuous encouragement, and to Mr. and Mrs. John Cohen for giving me the warmth and recreation of their home to cherish. The invaluable support from Anne made the effort worthwhile, her patience and love defeated my perpetual tardiness.

1. Introduction and Summary

1.1 Introduction

The production control activity has, generally speaking, three major responsibilities: timely service of customers, avoidance of unnecessary work in-process and end product inventories, and efficient utilization of the plant capacity. Possible conflicts of these goals have to be resolved by the job shop scheduling activity, but the traditional scheduling rules have had only limited success [17]. A rule that favors the most expensive and short jobs, such as the Weighted Shortest Processing Time first (WSPT) rule, guarantees reasonably efficient use of the capacity but often leaves some longer jobs very late whereas rules that emphasize the due dates, such as the Earliest Due Date first (EDD) rule, fail if the shop gets congested. The combinatorial complexity of the scheduling problem with due dates and job specific delay penalties, or the weighted tardiness problem, prohibits the optimal solution of all but unrealistically simple cases.[1] Academic interest has been directed to other problems for which exact solution procedures can be found while the practically important ones, including the minimization of weighted tardiness costs, are still by and large unexplored [15, 33].

The purpose of this thesis is to improve the use of information concerning the load of the shop in the scheduling process. We analyze the marginal costs and benefits of scheduling decisions under weighted tardiness criterion to understand the interactions of job attributes, shop layout and loading of the machines and their effects upon the performance of a given sequencing rule. In order to reduce the complexity of the search for efficient sequences of the jobs we use a dispatching approach: when a machine becomes idle, one of the waiting jobs is started next according to the priority index assigned to each job. Since every machine can be loaded as soon as there is at least one schedulable job in the queue in front of it, the dispatching discipline eliminates any unnecessary idle time of the machines. Timely delivery of products and low level of in-process inventories can be achieved by using a priority index function that coordinates the different operations of a job with respect to its due date and processing time requirements, taking into account the competing jobs and resource availability. A dispatching rule can also be used, in a simulation mode, to generate and test schedules for longer time periods, such as weekly or monthly shop schedules and quarterly production plans. The dispatching approach should work like a good manager: use simple and efficient reasoning, be flexible in different load situations, and adapt the rules when the situation changes. This can be achieved using a *state dependent priority rule*. First, a priority index evaluation should be a tangible, computationally efficient procedure and easy for the managers to understand. Second, a flexible priority rule can be applied in several different scheduling problems without major changes in the priority index function or the dispatching procedure. An empirical study by Jones [42] suggests that simple rules do not achieve this kind of long-term robustness without some additional information to allow more explicit trade-offs among the different goals. Third, the scheduler has to adapt continuously to changing problem situations. If the rule is not flexible

---

[1] Even the single machine case with weighted tardiness criterion is NP-hard, see [45]. Fisher [22] has developed an enumerative solution procedure for small resource-constrained scheduling problems using Lagrange multipliers.

enough to handle efficiently an anticipated change of the load, the priority index parameters can be adjusted temporarily for more appropriate response. Coordination of the parameter changes can pose major problems, however. Incorrect or incomplete forecasts can further jeopardize the possible gains from the reaction to the anticipated changes. An advantage of a myopic dispatching rule is that the performance achievable by the rule can be maintained despite the uncertainty of the future.

Extensive research for over twenty years has pursued a flexible, efficient and intuitively appealing priority rule.[2] A simple form of a state dependent rule is the (Anticipated) Work In Next Queue, or (A)WINQ rule [17, 58], whose priority index is the work load (or anticipated work load) on the machine the job will visit next. A more elaborate state dependent dispatching rule, the Dynamic Composite Rule (DCR) discussed by Conway et. al. [17], incorporates into its priority index the operation due date, the processing time of the operation, the work in the current queue, and the work in the next queue relative to the total load in the shop. Carroll [16] developed and tested a parametric family of CoverT rules which incorporate a "look-ahead" feature: the global slack, evaluated against a standard waiting time allowance, determines the job's expected marginal tardiness cost and its priority index. Both DCR and CoverT rules performed well in computational tests for several tardiness related measures such as number of tardy jobs and average job tardiness, but these rules has been found sensitive to the values of the parameters used in the priority index calculation [16, 40]. Also, these rules are not meant for weighted tardiness scheduling.

The development of state dependence of priority dispatching rules should address the question of distributing the planning, implementation and control of the scheduling activities in the organization by establishing the necessary coordination for efficient local decisions. The quality of information in the scheduling system should be re-examined, including the use the expectations concerning the load of the shop in the near future, the feedback from the anticipated performance of the rule, and the sensitivity of the scheduling mechanism to incorrect and incomplete data. Within our extended framework of state dependent priority rules, we distinguish two principal ways to add more information and coordination to a dynamic priority rule:

1. The extension of the horizon and type of information feedback.
2. The extension of the scope and detail of the status information.

The state variables and parameters of the priority index are based on some *horizons of information feedback* (see the framework in figure 1-1). Static rules (WSPT, EDD) use only the initial data of the problem, whereas dynamic rules include time dependent terms, such as slack of the job and work load in the next queue. Myopic dynamic rules use the observable status of the job shop without any forecasting. If a forecast of the future status of the job shop is available, current dispatching decisions can be made dependent on the anticipated state indicators. Finally, the anticipated performance of the rule over some forecasting horizon can be used to adjust its parameters and to choose the best estimates of the state variables over a planning horizon.

The *scope and detail of the load information* is another dimension that differentiates the

---

[2]Detailed surveys of scheduling heuristics can be found in [3, 17, 33, 58, 66], and chapters 2 and 4 below.

SCOPE AND DETAIL OF STATUS INFORMATION:	HORIZON OF INFORMATION FEEDBACK: 1. Observable status	2. Anticipated status	3. Performance feedback
A. Local	Slack/Operation Chs. 2, 4 and 5.	(Ch. 2.)	Ch. 2.
B. Indirect Global	CoverT Chs. 4 and 5.	Chs. 4 and 5.	Chs. 4 and 5.
C. Direct Global	WINQ, DCR Ch. 6.	AWINQ Ch. 6.	Emery's rule (Ch. 6.)

Figure 1-1: A framework for classifying dynamic priority dispatching rules according to the information used in the priority index. The matrix shows some existing rules and the chapters discussing the new enhancements.

priority indexes. Local rules use only the information concerning the current machine and the jobs in its queue. In addition to the local information, some rules use observable or anticipated information of queues of the other machines and the general characteristics of the load as well. This global load information can be indirect. The estimates of expected waiting times, for example, can be derived from aggregate load indicators, such as the distribution of processing times and delay penalties without explicit reference to the attributes of the specific jobs to be scheduled on the subsequent machines. Direct global information would include observed or anticipated measures of the jobs to be scheduled on the other machines, such as the queue lengths or the opportunity costs of the machines.

This classification shows the principal categories of information in state dependent dispatching rules. It does not implicate that the use of some global or anticipated load indicator would make a rule perform better. The contrary is sometimes true: the consistency of priority assignments over time can be more important than the exact ranking of local opportunities. Hence the value of state information depends on the scheduling problem at hand and the specific priority rule implemented. It also depends on the quality of the data available in terms of completeness, correctness, accuracy and timeliness. The most representative rules reported previously are shown in our new framework in figure 1-1. The priority index of the Slack per Operation rule depends on the current slack of the job, or the time remaining until its due date and the number of remaining operations. The CoverT index is based on the expected waiting time of the job on the subsequent machines, which can be estimated more accurately if the aggregate utilization of the shop is known. Direct load information, the current work in the machine queues, is used in WINQ and DCR indexes. The anticipated load information included has been the

anticipated work in the next queue, as in the AWINQ rule. Emery's dispatching procedure [20] is the only rule that uses iterative evaluation of the performance of the rule in search of optimal parameter values. This multi-stage screening procedure is, however, more complex than a priority index evaluation. Several interesting classes of potential state dependent rules have not been formulated at all as indicated in figure 1-1. The development of the new kinds of priority rules, possibly based on the previous rules, is the challenge of this thesis.

A state dependent priority index is derived from some load indicators, such as queue lengths, waiting times or due date distribution, using adjustable parameters. The previous studies reported in Conway et. al. [17], have used an experimental search to determine the best constant parameter values for the priority index function. One simple way to adapt the rule to the problem is to run simulations with several different rules and then choose the rule that performs best [55]. A more systematic way if a weighted priority index approach. The priority index of the scheduling rule can have several components, for example SPT and Slack/Operation priority indexes, or SPT and AWINQ. The weights of the component indexes can be set according to the experience with the rule in some previous problems [3, 10, 17]. The rule can thus, in principle, "learn" the best weights for the composite index in a given job shop and load situation. Iterative heuristic search procedures have been applied in the resource constrained project scheduling system by Wiest [70] and the job shop scheduling program by Holloway and Nelson [40]. These systems start with some feasible schedule and improve it locally by applying heuristic change rules. Emery [20] provides an interesting attempt to design an optimum-seeking procedure for the adjustment of the parameters of a five-component dispatching rule. Actually, Emery used a two-stage method by screening first the less urgent jobs from the final dispatching decision that was based mainly on the CoverT rule. Kriebel and Fox [25] show that the weights used in a composite dispatching rule can be estimated from the past decisions of a human scheduler, in the same way Bowman estimated efficient linear decision rules for aggregate planning [13]. In the development of expert systems for managerial applications, the terms of a state evaluation function are usually based on rules imitating the reasoning of a human expert. Fox [24] gives an example of a heuristic search procedure in which experienced schedulers can provide the evaluations of the different states of the job to furnish the choice among alternative machines for an operation, or among more specific queue positions. Haley and McDermott [36, 37] augment the critical path method with local rules which incorporate some additional constraints and expert knowledge. An incremental schedule is constructed for each job via opportunistic search which is guided by several global rules or "demons". The role of demons in the knowledge based scheduling process is similar to that of shadow prices in coordinated hierarchical systems. These attempts to incorporate load indicators and managerial expertise into a scheduling rules have been problem dependent and difficult to transfer from the test-bed into different shop environments. We hope that the results of this thesis will provide a step toward more rational, cost based derivation of the priority rules and their parameters.

1.2 Summary

The contribution of the thesis falls into three areas. First, we have proven several properties of optimal schedules in flow shops with unit jobs and different machine speeds. Although the special structure allows us to avoid some of the complexity of general flow shop problems, the

results concerning the role of bottleneck machines and lead time estimates in the successful decentralization of scheduling decisions have important implications to the development of state dependent priority rules for general flow shops and job shops.

Second, we have improved priority rules for weighted tardiness problems in single machine shops, static flow shops and dynamic job shops. The large scale tests indicated that the CoverT rule [15, 16], after modification for the weighted tardiness problem, can outperform the other standard rules. Our new Apparent Urgency rule is consistently better than CoverT using the traditional local information for lead time estimation. Appropriate use of indirect global information (expected utilization of the shop, distribution of the delay penalties of the jobs) improves both rules, making CoverT best in extremely congested shops. Successive adjustment of lead time estimates through an iterative procedure allows the use of performance feedback in the search for improved estimates. This lead time iteration yields the best performance in weighted tardiness and combined schedule costs, whereas additional direct load information (anticipated queue lengths, average anticipated urgency of the jobs on the next machine) with periodic adjustment of the rule parameters does not improve the average performance of the Apparent Urgency or CoverT rules. The new state dependent rules have been extremely robust. The same rules work with minor modifications in different job shops and varied load conditions, the performance has been insensitive to erroneous data where tested, and the rules are superior to the previous rules (including the original CoverT) in terms of other important criteria such as number of tardy jobs and maximal weighted tardiness of any job while maintaining low levels of work-in-process inventories.

Third, the formulation of the new state dependent priority rules is based on our extended framework of information and coordination of the dispatching process. The distinctions between indirect and direct global information, on one hand, and between status and performance feedback, on the other, are shown to be significant in determining the minimal information requirements for a desired level of schedule performance. Furthermore, the experiments with the self-adjusting priority indexes verify the previously intuitive setting of the slack evaluation parameters for Apparent Urgency and CoverT rules depending on the anticipated load. Hence our framework serves as a basis for documenting and developing the expertise of state evaluation in scheduling support systems. A summary of the thesis is given below covering the main topics and results of the remaining chapters. The state dependence studied with the new rules in each of the chapters is also indicated in the framework in figure 1-1.

Chapter 2: A Framework for State Dependent Priority Rules

This chapter provides an extensive review of the dispatching rules used for scheduling in job shops under tardiness related measures of performance. A rule that involves a job's processing time, due date and delay penalty is the look-ahead rule, the Apparent Urgency rule, first tested in the single machine case by Rachamadugu and Morton [61, 62].[3] Based on this survey, we classify the information that could be used in a state dependent rule according to the forecasting horizon to observed status, anticipated status, and feedback from anticipated performance. According to its scope and detail, the load information used in the priority index is

[3]We modify also the CoverT rule [16] to accomodate the job specific tardiness penalty in chapters 4 and 5.

either local to the machine, global but indirect such as the distribution of job attributes, or direct global information such as average queue lengths on different machines. This general framework forms a basis for the development and testing of state dependent rules.

Some properties of the locally optimal Apparent Urgency (AU) rule, for example the form of the look-ahead function are discussed. We revise an iterative heuristic procedure for the AU rule to adjust the length of the look-ahead for each job according to its relative earliness or tardiness in a previous schedule. The adjusted look-ahead parameter incorporates feedback of the anticipated performance into the otherwise myopic rule. In a simulation study with some of the hard problems studied in [61], the look-ahead adaptation saved 30-60% of the margin between the optimal weighted tardiness and the solution with the constant look-ahead, in par with a locally optimal pairwise interchange procedure. The AU index values are shown to be insensitive to the errors in processing time estimates used in the iterations. The indicated robustness of the rule was tested in a one-machine simulation study with varying the maximal error in processing time estimates from 30% to 90% of the actual. The performance of the constant AU rule deteriorated only by 2% to 36% from the cost of the schedule in the deterministic case. The adaptation of the look-ahead parameters of the jobs improved the performance in all cases. Good performance of the basic and adaptive AU rules, in terms of other criteria such as the number of tardy jobs and the maximal weighted tardiness, persisted despite the estimation errors. Hence the iterative look-ahead adaptation constitutes a planning procedure that can successfully utilize even erroneous data in setting essential control parameters for the detailed scheduling.

Chapter 3: Scheduling in Proportionate Flow Shops

The development of state dependent dispatching rules for multi-stage processes or flow shops is started by studying scheduling problems in proportionate or uniform flow shops: the job processing time on any machine is proportionate to the processing time on the first machine. Though flow shop problems are a special case of job shop problems. even these problems have proven themselves so far to be too complex to provide any analytical solutions. Except for the makespan results in the case of two machine flow shop by Johnson [41], Gilmore and Gomory [29] and the two results characterizing optimal solutions by Conway et. al. [17], there are no known analytic solutions for flow shop problems.

In this chapter, the optimal solutions of the pre-emptive version of the problem, i. e. with unit time jobs in the proportionate flow shops, are characterized. Some of these properties are used in developing iterative procedures for the weighted tardiness problems in general flow shops in the next chapter. First, we show that permutation schedules constitute a set of dominant schedules for minimizing any regular measure of performance.[4] This characterization by itself reduces the search space for seeking optimal solutions. Second, we show that makespan is minimized by any permutation schedule. The derivation of this result shows also how the makespan depends critically upon the bottleneck machine (the machine with the largest job processing times), a result which makes intuitively sense and explains why practitioners tend to be very much concerned about bottleneck machines. Further it is shown that, analogously with the

[4] In the case of ordinary flow shops, this statement is valid only on the first two machines for any regular measure of performance and also on the last two machines in the case of makespan problems [17].

single machine case, the weighted mean flowtime is minimized by the Weighted Shortest Processing Time rule and the maximum lateness is minimized by the Earliest Due Date rule. The weighted tardiness problem can, in the special problem discussed here, be reduced into a linear assignment problem. It is shown that the necessary condition of optimality in the case of weighted tardiness criterion is similar to the condition previously obtained in [61]. These results are further extended to the cases in which job-passing is not permitted. It is also shown that when job-waiting is forbidden, i. e. when a job can not wait between subsequent operations,[5] all the above results hold good except that the start times on the first machine have to be delayed appropriately.

Chapter 4: Lead Time Iteration in Flow Shop Scheduling

Though no computational results have been reported in flow shops for the weighted tardiness criterion, extensive experimental work has been done by Carroll [3, 15, 16], for the average tardiness problems in job shops. He developed a parametric family of look-ahead rules called CoverT which performed better than other rules such as SPT and Slack per Operation in dynamic job shops. The performance of the rule was found to be sensitive to the setting of a parameter adjusting the "length" of the look-ahead (as measured in the units of standard waiting time in the subsequent queues). Some other priority dispatching rules include setting "operation due dates" based on the remaining processing time, simple estimates of the waiting times, or some other considerations.

In this chapter, the existence of optimal operation due dates is established in the preemptive version of the proportionate flow shop. In this case it is shown that if the lead time estimates are chosen optimally, then the scheduling problem can be decomposed into single machine optimization problems. Local optimal solutions to these problems correspond to a global optimal solution. The derivation of this result also indicates how the optimal operation due dates are influenced by the bottleneck machine and the external job due dates. The optimal operation due dates are shown to be robust against errors in the lead time estimates thus providing promising ground for the application of approximate procedures. Using these results, an iterative lead time estimation scheme is developed for the flow shop problems as follows: we start with an initial lead time estimate and decompose the problem into several single machine problems. These can be solved in dispatching mode using the Apparent Urgency rule and applying the appropriate operation due dates. Then in the subsequent iterations, the due date setting uses the lead times realized in the previous iteration. The process is repeated a finite number of times and/or until no further improvement takes place in the measure of performance. In a large computational experiment consisting of 1280 problems with sizes varying up to 60 jobs and 8 machines, the AU rule outperformed other competing heuristics such as CoverT and four other rules by at least 10% in the average. The adaptation of the lead time estimates through iteration further improved the weighted tardiness performance of the AU rule, additional savings ranging from 10% to 25% depending on the shop load conditions. The results proved to be relatively insensitive to the variation of the parameters of AU rule and CoverT over a wide range of problem specifications. The coordination achieved through the use of better state information in the adaptive rule is advantageous also on the basis of some other important measures, such as the number of jobs tardy, the work in process inventory, and the maximal weighted tardiness.

---

[5] Job waiting is naturally not possible in continuous process industries such as glass production.

Moreover, the fact that these rules are dispatching rules makes them rather easy to implement in practice.

Chapter 5: Slack Evaluation for Priority Dispatching in Dynamic Job Shops

Waiting time estimates are used in the CoverT rule to evaluate a job's priority index by comparing its global slack to the sum of standard (worst case) waiting times. The slack evaluation of the Apparent Urgency rule has two parts: the global estimates of the actual (resource constrained) waiting times determine a local operation due date, and the local slack, compared to the processing times of the competing jobs, constitutes the look-ahead. In dynamic job shops studied in chapter 5, jobs arrive continuously on the machines and the simulation can extend long time periods. Since the jobs have a random routing, their expected waiting times can be studied with queueing theory. The new estimates of the waiting times for a job with a given natural index (weight/processing time) value, can be obtained through a numerical approximation of the results of waiting line analysis. These "priority-based" waiting time estimates are used to derive two new waiting time estimation functions, an inverse function of the natural index and a composite function of the weight and the processing time. These approximations are then tested in weighted tardiness performance against the traditional multiples of processing times of the AU and CoverT rules. The parametric lead time estimation methods require some indirect load information, such as the capacity utilization of the shop and the weight and processing time distributions of the jobs.

The iterative estimation of job waiting times introduced in chapter 4 can be implemented in the dynamic job shop via rolling forecasting and planning horizons. A forecasting horizon is the time period over which the performance of the rule is simulated. The best waiting time estimates, found during the successive simulations, are implemented over a planning horizon which is shorter than the forecasting horizon. However, the lead time iteration proved rather insensitive to the variations in the forecasting and planning horizons.[6] We tested the weighted tardiness performance of the constant and iterative AU and CoverT rules in large job shop problems with 10 machines and 2,000 jobs. Expected capacity utilization varied from 80% to 97% of bottleneck capacity, and the random due date allowances had the average of 3 or 6 times the average total processing time. Three different shop structures were studied: general, proportionate, and bottleneck job shops. The results with the conventional static waiting time estimates were similar to the results in flow shops. The AU rule was best in the weighted tardiness and in the number of tardy jobs, followed by CoverT and the other heuristics. More accurate lead time estimates, either through iteration or through the new priority-based lead time estimates, improve the AU rule 3%-10% in the weighted tardiness performance. However, the same lead time estimates helped CoverT to even better performance in some heavily loaded shops, due to the better smoothing of job priorities over the global slack. The AU rule can balance machine loading and job tardiness through the lead time iteration, resulting in lowest combined costs of tardiness, inventory-holding and late shipments with CoverT lagging up to 10% behind. These results establish the new priority-based waiting time estimation method as a new standard for the slack evaluation by the look-ahead rules.

---

[6] The possibility of determining dynamically the most appropriate length of the horizons, or to obtain weak horizons [53], will not be studied here.

Chapter 6: Coordination of Job Priorities via Direct Load Information

In the previous tests of the look-ahead rules, constant slack evaluation parameters have been applied in the simulation through a long busy period including widely varying shop conditions. In this chapter, we experiment with a periodic adjustment of the look-ahead and waiting time estimation parameters of the Apparent Urgency rule, based on the anticipated critical queue lengths on the machines. This anticipated direct load information can be obtained by simulation over a forecasting horizon. The results obtained after extensive experimentation with several plausible parameter adaptation functions indicate that the parameters of the AU priority index can be automatically adjusted without any significant loss of performance when compared to the best static rules. But no consistent improvement can be achieved either, due to insufficient coordination of the local priorities. The tests with several "self-adjusting" mechanisms verify, however, the best parameter setting and other coordination requirements of the previous dispatching rules.

The complexity of job routings in a job shop suggests also that the immediate priority of a job could be adjusted according to the load on the next machine. We use the projected relative priority of the job on the next machine as an argument of the priority index on the current machine, instead of the more conventional queue length or work contents of the next machine used in AWINQ [3, 17, 58] and the Dynamic Composite Rule [17]. The "probing" term of the AU priority index function keeps the machines loaded but allows the urgent jobs to be rushed through the shop. Hence the two-stage AU rule should improve the performance with respect to most of the criteria used above by avoiding congestion and idle time on any bottleneck machines. The results of test simulations proved rather disappointing: the probing of the projected urgency differential on the next machine lead only occasionally to a significant improvement of the weighted tardiness criterion.

Chapter 7: Conclusions and Future Research

The last chapter summarizes the contributions of the thesis and discusses the issues related to the implementation of the state dependent priority rules. We have studied new state dependent dispatching rules for weighted tardiness scheduling, starting with a constant parameter rule on a single machine and building gradually more complex applications in static flow shops and dynamic job shops. The new Apparent Urgency rule has been tested in these applications, and the CoverT rule has been modified for better performance in weighted tardiness problems. We have enhanced these rules by incorporating indirect and direct load information to the priority index valuation, and by using lead time iteration to obtain performance feedback for parameter adjustment. The results of extensive testing have justified the added complexity of the state dependent rules on grounds of reduced weighted tardiness costs, low inventory holding costs and fewer tardy jobs.

Future research should extend the applications of the state dependent dispatching rules. First, scheduling under combined costs, such as inventory-holding, tardiness and rush-shipping costs could be tried in dispatching mode using a composite priority index, the expected marginal benefits of a decision per the marginal costs of implementing it. We specify the most promising composite priority index functions for several cost components. The allocation of multiple resources along the dispatching process and the procedures for the evaluation of their

opportunity costs are discussed. Second, the scheduling of the production lots of several products with finite productions rates on a single facility leads to an inventory tardiness problem with dynamic lot sizing. The costs for inventory holding and tardiness (backordering) are convex, presupposing a modification of the AU priority index. In an iterative two-level solution procedure, capacity evaluation could be used to find the relevant production and availability constraints which determine the appropriate setup costs and dynamic lot sizes. Given the lot sizes, slack times (inventory availabilities) and quadratic penalty functions, a priority index can be derived for dispatching the next lot.

Third, the application of priority rules in service centers can create incentive problems since some customers would have to wait much longer in the optimal schedule than under the "fair" FCFS discipline. The delay penalties needed for priority assignment are unobservable and often private information, complicating the equitable assignment of service priorities. We propose to modify Dolan's [19] centralized taxation schema for lateness problems into a dynamic, decentralized transfer-pricing mechanism that eliminates possible gains from strategic cheating. The nonlinearity of the tardiness costs breaks down any transfer-pricing mechanism in the case of different due dates. Several plausible bidding strategies could be tested for sequential priority auctions on a single facility. Finally, we believe that priority rules provide an appropriate way to accumulate expertise for the design of more sophisticated search systems, such as scheduling expert systems [23, 24, 37]. A primary area for application would be the diagnosis of schedule and capacity problems, a neglected area of scheduling research [60, 69]. Among the constraints to be considered in state evaluation are precedence relations among the jobs, allowing alternative routings and sequence dependent setup times, and other resources. Linking aggregate capacity and materials planning to detailed scheduling is a potential area for heuristic search, with few attempts with empirical [9] and optimizing [26, 27] approaches. The trade-off between the dynamic tardiness costs and the fixed costs of facilities could be studied in a hierarchical planning framework similar to the closed shop case in [32, 34]. The diagnostic study of the status of a job shop could be based on its approximate loading using a state dependent rule. A cost-oriented priority index is essential for the effective integration of the diagnostic routines and the incremental scheduling of implicated improvements. Hence the knowledge representation of the expert system [24, 36, 68] should reflect the required state information, including indirectly derived parameters and opportunity costs of resources.

Co-authorship

The results of chapter 2 were first reported in [67]. The research reported in chapters 3 and 4 was joint with Professor Thomas Morton and Dr. R. V. Rachamadugu, references [63], [66], and also [62]. The research for chapter 5 was joint with Professor Morton.

2. Framework for State Dependent Priority Rules

Summary

Scheduling with due dates and weighted tardiness criterion is a common business problem for which there is no adequate theory of optimal solution. Hence the development of new heuristic approaches for solving problems of practical size and with realistic information availability is warranted. State dependent priority rules use information concerning the anticipated load in the shop or the simulated performance of the rule in order to adapt to the local opportunities for improved dispatching decisions. The trade-off between the locally enhanced performance of the rule and the long run coordination is discussed. The information feedback in priority index evaluation extends the framework for state dependent rules and suggests new practical methods for load estimation. The new concepts are demonstated in single machine weighted tardiness problems. We tested a new "look-ahead" rule called Apparent Urgency as the prototype of a state dependent rule.[1] The priority index value of a job is reduced expontially with longer slack. The factor that determines the length of the look-ahead can be adapted in a heuristic fashion, using iterated performance feedback, to a particular shop load situation. This look-ahead adaptation improved the worst case performance of the of the Apparent Urgency rule to the level of pairwise interchange procedure. Moreover, the new rule is shown to be robust against errors in jobs' processing time estimates.

2.1 Introduction

2.1.1 Weighted Tardiness Problems

Scheduling against due dates and job specific delay penalties, or the weighted tardiness problem, is common in most job shops [15, 33]. In heavy equipment industry, some plants use timely shipment of orders for strategic positioning of their products. For example, the capital tied in generators, turbines and in the operations dependent on this equipment justifies the high premiums for prompt spare part service of the customized components needed on short notice, classified as forced outages. Sometimes the promised delivery date, or manufacturer's due date, brings along contractual penalties for late shipments. Tardiness causes also potential losses in terms of bad reputation, higher shipping costs, etc. These problems encompass complex technical and organizational information processes, relating the scheduling activity directly to the strategic management of the plant. Communication between marketing and manufacturing calls for better understanding of the sensitivity of a production schedule against capacity changes (machine breakdowns) and new rush orders. These characteristics of the weighted tardiness problems, dynamics, complexity and uncertainty, have rendered the use of exact methods and existing dispatch rules insufficient in practice. Consequently, most tardiness related scheduling is done manually by experienced schedulers who can cope with the process information and unpredictable organizational responses in technologically intensive service operations. Formal

[1] The Apparent Urgency priority index was first derived by Morton and Rachamadugu in [61]. See also [62].

scheduling systems, where implemented, have often been abandoned by the schedulers due to the additional data preparation required, errors, inadequate reporting and inflexibility of problem formulation.

Weighted tardiness problems are known to be NP-complete even in the one machine case [45]. Enumerative methods can solve static scheduling problems consisting of up to 30 jobs on a single machine [33, 61].[2] Related to the branch and bound techniques, there has been some interest in sequencing methods, i.e., heuristic methods for generating optimal solutions in some special problems, and statistically good solutions to general static scheduling problems[3] [5, 21, 64, 65]. These sequencing procedures define sufficient dominance properties between the jobs in order to eliminate some of them as them as starting or ending jobs from the sequence enumeration. Examples of sequencing methods are the Schild-Fredman method [64], and the Montagne method [5]. These sequencing methods have not been applied in dynamic problems of realistic size.

The failure to find exact solutions has motivated our search for efficient heuristic approaches. The problem is formulated as a dispatching process in which machine loading decisions are made dynamically when there are schedulable jobs on an idle machine. A dispatch rule is used to assign a value, priority index, for each job. The jobs are then ranked according to their priority indexes, and the most urgent is started. The dispatching discipline generates non-delay schedules without inserting idle time [3]. Dispatching constitutes often a single pass search in which the start time of every operation is determined only once. However, some adjusting procedures, such as the look-ahead iteration proposed in this chapter, can be used to improve the schedule after several jobs have first been scheduled tentatively.[4] The purpose of our study is to develop more efficient dispatching methods for weighted tardiness problems by incorporating more status information (load, capacity, relative urgency of the jobs, etc.) into the priority evaluation. The results of the experimental simulations have been encouraging.

2.1.2 Review of Priority Dispatching Rules

The reported research in the heuristic rules for weighted tardiness problems is scarce, but other related measures of performance, such as total or average (unweighted) tardiness and fraction of tardy jobs, have been investigated quite extensively. The research of dispatching rules, and the simulation methodology used in this research, was established by the RAND studies in the early 60's [17]. Those results and more recent survey articles on the development priority rules, e. g. [3, 17, 51, 58] are summarized here. The Weighted Shortest Processing Time first -rule (WSPT) can be shown to minimize the weighted tardiness criterion in a single machine shop if all jobs are necessarily tardy [5]. WSPT, or SPT in average tardiness problems, performs consistently in most problems by expediting several short jobs while delaying the processing of few long and cheap jobs. Especially in high load situations, this principle seems to more than

---

[2]For a detailed review of enumerative methods in weighted tardiness scheduling, see [33, 61].

[3]This kind of methods are called "algoristics" in [52].

[4]Some notable examples of adjusting, or improving, procedures are described below, see also [18, 28, 40].

compensate for the complete lack of due date information in the WSPT priority index. Another intuitive heuristic is the Earliest Due Date (EDD) rule that can generate an optimal schedule whenever it is possible without making any job tardy [64]. Consequently, one would expect it to be a reasonable rule for lightly loaded shops. Its performance deteriorates, however, even with moderate increase of load. A multi-machine modification, the earliest Operations Due Date (OPNDD) produced high variance of the job flow times [17]. Under OPNDD discipline, the initial time allowance before job due date is divided evenly (or according to the processing times) into operation lead times to assign operation due dates. These artificial operation due dates are then used as a basis for myopic EDD dispatching. These rules ignore job's weight (relative importance or delay penalty) and their processing time information, leading to the inefficiency in congested shops with short lead time allowance. Three dynamic variants of the OPNDD rule use no operation due dates: total Slack remaining, the Slack per Operations remaining (S/OPN) and the Slack per Remaining Processing Time S/RPT [3]. These rules determine the remaining slack, i. e. the time until the due date less the lead time of remaining operations, on the arrival to the machine queue [3, 9, 66]. Dynamic slack rules work well in light load situations. Critical Ratio, CR, is the ratio of dynamic slack to the standard remaining lead time used as priority index in many MRP systems [9]. Gere [28] and Baker [4] have proposed a dynamic version of the EDD rule that seems to work well for the unweighted tardiness problems. This modified due date method changes the due date of any late job to be the earliest possible date it could be completed; it then applies the normal EDD.

The relative total slack constitutes look-ahead information in Carroll's CoverT rules [3, 15, 16, 51]. The CoverT priority index is based on the projected (unweighted) tardiness cost, c_j , and the processing time, p_j, of job j on the machine in question, or "Cost OVER Time". If d_j denotes the job due date, the priority index of job j at time t, π_{CoverT}, would be:

$$\pi_{CoverT}(t) = c_j/p_j = (1/p_j)[Q\,W_j - (d_j - r_j - t)^+]^+/(Q\,W_j)\,, \tag{2.1}$$

where W_j denotes the anticipated waiting time and r_j the remaining operation time.[5] If the slack $d_j - r_j - t \leq 0$, the anticipated tardiness cost is set to $c_j = 1$, and if the total slack exceeds the worst case waiting time QW_j, the cost is set to $c_j = 0$. The parameter Q can be adjusted to control the look-ahead period of the rules. CoverT outperformed standard rules SPT and OPNDD in an average tardiness problem with an appropriate selection of parameter Q.[6] When jobs have different routings through the shop, the priority of the job on the current machine can be lowered by the length of the queue on the next machine to avoid congestion and uneven use of capacity. To prevent overreaction, the rule usually anticipates new jobs entering the next queue by the completion of the current job, hence the name Anticipated Work In Next Queue, AWINQ, [3, 17]. Another way to balance the conflicting objectives is adopted in linear combination rules, which use the weighted average of the component priority indexes, for example $a\pi_{SPT} + (1-a)\pi_{WINQ}$ or $b\pi_{SPT} + (1-b)\pi_{S/OPN}$ reported in [17], Often the combination rule outperforms the constituent rules, contingent to appropriate parameter value selection. The estimation of best parameter values has been seen as an empirical problem without theoretical results for guidance [3, 17, 40].

---

[5] Here we use notation $(x)^+ = \text{Max}\{0, x\}$.

[6] In the test with a light load, the CoverT rule with value Q = 0.5 performed better than Q = 1, see [3, 15].

A complex 3-parameter state dependent rule, Dynamic Composite Rule or DCR, was also tested in the average tardiness studies [17]. The priority index at time t, $\pi_{DCR}(t)$, was determined by the operation due date, o_j, and the operation processing time, p_j:

$$\pi_{DCR}(t) = o_j - p_j + b\,[\Sigma\, p_i]^r\, p_j + h\, W_{nq}\,, \qquad (2.2)$$

where b, r and h are constant parameters, $\Sigma\, p_i$ is the sum of the processing times in the current queue, and W_{nq} is the ratio of the work contents in the next queue to the total work in the other queues. Thus DCR dependes on the due date, the processing time of the job and the congestion of the current as well as of the next queue in choosing the next job to be started. With proper parameter values, such as b = 0.3, r = 1 and h = 160, the DCR rule achieved the lowest average tardiness and conditional tardiness. However, the efficient choice of one parameter was entirely dependent on the values of the others [17].[7]

Some classification of the schedulable jobs and the scheduling state can be used to select one rule among several rules for priority index valuation. For example, a truncated SPT rule switches from SPT to EDD when the machine queue falls below a critical length, or when the waiting time of some jobs exceeds some limit. Swapping between several rules is subject to the same criticism as the linear combination rules: the procedure is not always consistent in different load situations, and its performance is sensitive to the values of the parameters involved. The dispatching discipline can also be extended by including multiple passes through the loading procedure. Gere [28] suggested several "second-pass" adjustment procedures, and Holloway and Nelson [40] gave an example of a multipass heuristic search procedure for lightly loaded shops.[8] Emery [20] provides an interesting attempt to design an optimum-seeking procedure for the adjustment of the parameters of a state dependent dispatching rule. Actually, Emery used a two-stage method for screening the less urgent jobs from the final dispatching decision that was based mainly on the CoverT rule.

In summary, only few heuristic rules are capable of reasonable average tardiness scheduling in widely varied load situations. This deficiency results, we hypothesize, from ignoring relevant state information in the priority index evaluation. No previous rule has all three attributes value, processing time, and due date in the priority index.[9] The use of state information, e. g. the lead time estimates, has been unsatisfactory in many experimental simulations [3, 9]. The effects of the local environment, such as the jobs in the next queues, have been included in some priority rules, but more global estimates of the future load are missing. Rachamadugu and Morton tested a new look-ahead heuristic for weighted tardiness dispatching, the Apparent Urgency rule [61], that performed close to the optimum in single machine problems. A parametric family of Apparent Urgency rules forms the basis for our state dependent rules.

---

[7] Unfortunately, DCR was not tested against CoverT. Conway et. al. [17] concluded in their study that the slight improvement of the performance of the state dependent rules over the static combination rules does not warrant their use in job shops having a manual scheduling system and relatively modest delay penalties. The availability of computer based information systems in shops with major tardiness penalties motivates our closer analysis of the state dependent rules.

[8] Dannenbring [18] discusses the difference between one-pass, or "schedule generating" procedures and multi-pass, or "schedule improving" procedures in the case of minimum makespan problems.

[9] Adding a missing attribute to some of the priority indexes is sometimes straightforward, see the "weighted CoverT" in chapter 4, but it can also be less obvious as in the case of DCR, analyzed in chapter 6.

The balance of this chapter is organized as follows. In the next section, the concept of state dependent dispatching rule is defined technically, providing a new framework for further classification of the information feedback in priority rules. A state dependent priority rule is developed for one-machine weighted tardiness scheduling in section 3. An iterative scheme is used in extracting state information for the basic Apparent Urgency rule. We discuss and test also the sensitivity of the new rule against errors in processing time data. Concluding remarks can be found in section 4.

2.2 A Framework for Analysis of State Dependent Priority Rules

2.2.1 The Definition of State Dependent Priority Rules

The concept of priority dispatching rule can be formalized as follows [3, 58]:

Definition 2.2-1: A *priority dispatching rule* consists of a procedure for determining a value of *priority index*, π, for any job in a given scheduling situation, and of a *ranking procedure* using the priority indexes for selecting one of the schedulable jobs waiting on an idle machine to be started next.

Different dispatching rules have been discussed above. The priority index of the WSPT rule is the tardiness penalty of job j, w_j, over its processing time, p_j, for the imminent operation, or $\pi_{WSPT} = w_j/p_j$. The ranking procedure generates a sequence of non-increasing indexes and the highest priority is given to the first job in this sequence. For the EDD rule, the priority index is the due date of the job, $\pi_{EDD} = d_j$, and the jobs are ranked in nondecreasing order of π_{EDD}. Different rules have been characterized according to the following distinctions between static and dynamic rules on one hand, and between local and global rules, on the other [3, 58].

- In *static rules*, the priority index is constant over time: a static rule assigns a constant value of the priority index to each operation of a job when it enters the shop. In a multi-machine shop, the static priority index of a job can differ on individual machines, and hence over time, but the values of these indexes can be assigned from the outset. For *a dynamic rule*, the priority index changes over time even on a single machine. One example of a dynamic rule is the S/OPN rule: the dynamic slack per operation changes when the job proceeds in the shop.

- The rule is *local*, if the priority index is based on local information, typically the attributes of the jobs in the current machine queue.
  A global rule uses information concerning the status of the shop, such as the length of the present queue relative to the average queue length in the shop.

The dynamic and global dispatching rules reported in the literature have been parametrized, and the parameters values have been kept constant for all jobs over each simulation run. This practice is justified on the grounds of simplicity and lack of efficient parameter adjustment methods. However, we can include job-specific and dynamically adjusted parameters in the definition of state dependent dispatching rules:

Definition 2.2-2: A *state dependent priority rule* is a dynamic rule whose priority index for each job is a parametrized mapping from the state variables of the shop to real numbers.

More formally, we can define a state dependent priority index π of a job as a mapping from the job attributes and other state variables, such as the attributes of the other jobs and average load on the machines, and time t, $s = (s_1, s_2,...,s_n)$, and parameters $b = (b_1,..., b_m)$ that are free or depend on some state indicators, to the real numbers:

$$\pi: \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}. \tag{2.3}$$

Best values of the parameters are not necessarily functions of the current state of the dispatching system even for a regular measure of performance but might require the estimation of the anticipated status of the shop. In practical dispatching systems, we specify an approximation of the priority index mapping in Equation (2.3) using estimates of only few state variables, or anticipated state variables, $\tilde{s}$, and some parameters, $\tilde{b}$. This approximation can be called the priority index function, say g:[10]

$$\pi_j = g(\tilde{s}, \tilde{b}). \tag{2.4}$$

A state dependent rule is by definition a dynamic rule. The arguments $\tilde{s}$ of a state dependent priority index function are usually explicit measures of the load, such as the lengths of queues in the Dynamic Composite Rule in Equation (2.2) above. Job specific parameters, $\tilde{b}$, can also reflect the load in the shop. The information available for estimating the (aggregate) state variables and in setting the parameter values is determined by the forecasting and planning horizons of the dispatching process.

2.2.2 New Distinctions of State Information in a Priority Index

The classification the priority dispatching rules given above distinguishes two principal ways to add state information into the priority index evaluation: either by improving the quality of the information through extension of the forecasting horizon from static to dynamic, or by extending the scope and detail of the information from local to global. We propose to refine both of these dimensions to obtain a more useful framework for developing scheduling systems. First, the initial data of the problem used in static rules (job attributes) can be augmented by observed or anticipated state indicators. More specifically, the dynamic priority index formula and its parameters can be based on the following horizons of information feedback (see figure 2-1):

1. The observable status of the job shop, such as the slack of a job or the length of a machine queue.[11]

2. The anticipated status of the job shop in the future, for example average queue lengths estimated through aggregate forecasts or detailed forward simulation.

3. The anticipated performance of the rule and parameters in the problem under study. The performance feedback can be obtained through an iterative simulation or by solving an aggregate formulation of the problem. The parameter values and state estimators are selected based on the best performance of the incremental schedule.

---

[10] It is possible to reduce any multi-stage priority screening scheme, such as truncated SPT or Emery's method, to a priority index function.

[11] Instead of updating the state information upon each dispatching decision, the priority index is evaluated just at the arrival to the queue or the shop in simple rules.

SCOPE AND DETAIL OF LOAD INFO:	HORIZON OF INFORMATION FEEDBACK: STATIC: 0. Initial data	DYNAMIC: 1.Observable status	2.Anticipated status	3. Performance feedback
A. Local	EDD WSPT	Slack/RPT		
B.Indirect Global		CoverT		
C. Direct Global		WINQ DCR	AWINQ	Emery's rule

Figure 2-1: The classification of the priority dispatching rules according to the information used in the priority index

Second, the state information can also be analyzed based on the scope and detail of the load estimates used in the priority index function. Local information refers, as above, to the attributes of the jobs on the current machine. Global information, however, can further be divided into indirect and direct load information. The (anticipated) load can be used indirectly to infer the probable waiting times of the jobs or average queue lengths can be used directly in estimating the expected waiting times. Hence the spatial information used in the priority index function can fall into the following categories shown in figure 2-1:

1. Local information concerning the current machine and the jobs to be scheduled on it.

2. Indirect global information: In addition to the local information, the rule uses information that either characterizes the distribution of the problem parameters or can be inferred from the (anticipated) solution of the scheduling problem. An example is a lead time estimate obtained through simulation before priority index valuation.

3. Direct global information: the priority index uses explicit, observable or anticipated measures of load on several machines. Such direct information is the queue length on the next machine, or the implicated opportunity cost of the machine.

The adoption of this framework is justified if the new distinctions suffice to identify significant differences of the efficiency of the implicated rules vis-a-vis the costs of the

information systems needed. Some other issues pertaining to the development of state dependent priority rules are addresses in the thesis as well. First, bottleneck facilities require special attention to avoid any idle time and excessive lines. Can this be achieved using a priority rule? If the rules are adjusted for the bottleneck facilities, how do we ensure the coordination of priority assignments across machines? Second, can the rules take an advantage from a special layout of the shop, or other known structure of the load? How sensitive the performance of a particular rule is to the inaccurate specification of the shop? Do some errors in the status information degrade the performance of the rule? Third, can the estimation of the parameters of a state dependent priority index be decomposed, or does the change of one parameter require some internal adjustment of the others[12] ? Finally, how do we extract more information concerning the future status of the shop, such as a forecast of the lead time requirements of a job on the subsequent machines? We propose the use of *forward iteration* to estimate the values of any job specific parameters and load indicators of the priority index. This can be accomplished through diagnostic load analysis, through locally improving procedures, or through global performance feedback. Also, we hope that the rule parameters giving the most efficient performance convey some useful information about the load to operations management. Consequently, the new framework would provide guidelines for the integration of the detailed scheduling activity into the overall management information system.

2.3 An Adaptive Priority Rule for Single Machine Weighted Tardiness Scheduling

2.3.1 Some Properties of Apparent Urgency Priority Indexes

The objective in a weighted tardiness problem is to minimize the total tardiness cost, $\Sigma\, C_i(t_i)$, when the penalty associated with the completion of job i at time t_i, $C_i(t_i)$, is: $C_i(t_i) = w_i \max\{0, (t_i - d_i)\}$, where w_i and d_i are its value and due date, respectively. In a single machine weighted tardiness problem, a locally optimal condition for scheduling job i before an adjacent job j can be expressed as follows [61]:

$$(w_i/p_i)\,[1 - (d_i - t - p_i)^+/p_j]^+ \geq (w_j/p_j)\,[1 - (d_j - t - p_j)^+/p_i]^+ \ , \qquad (2.5)$$

where t is the current time, p_i and p_j are the processing times of the jobs. The derivation of equation (2.5) is based on the minimization of the marginal cost of scheduling one of jobs i and j in the current position. Since equation (2.5) gives a necessary condition of optimality, an approximation of this condition can be used as a priority dispatching rule. The priority index, called the "apparent priority", was first given in [61]:

$$\Pi_i(t) = (w_i/p_i)[1 - (d_i - t - p_i)^+/(k\,\bar{p})\,]^+. \qquad (2.6)$$

Here $\bar{p}$ is the average processing time of the schedulable jobs. The parameter k changes the length of the "look-ahead", or the slack which starts the anticipation of tardiness, that is measured in terms of the average processing time. In a large simulation study, the following exponential modification outperformed the linear priority index:

---

[12] The variability of parameter values in DCR and Emery's rule indicate their interdependence [17, 20].

$$\pi_i(t) = (w_i/p_i)\, e^{-(d_i - t - p_i)^+ / \kappa \bar{p}} \quad . \tag{2.7}$$

Again, κ determines the length of the look-ahead that is usually shorter than in the linear case. In order to understand the convex shape of the look-ahead specified by equation (2.7), we can find the locus of points of indifference for two jobs with identical delay penalties $w_i = w_j = 1$. Assume that in equation (2.5), the slack of job i, $(d_i - t - p_i) > 0$, we would be indifferent in starting this job or a longer job j that is already tardy (having the value $1/p_j < 1/p_i$ of the Apparent Urgency index) at time t_b if $d_i - t_b = p_j$. Hence the following property holds for slack evaluation of the shorter job i on the machine with two jobs:

Property 1: The unweighted Apparent Urgency index of the shorter job, i, having slack $p_j - p_i$, is $1/p_j$.

The convention of using the same length of the look-ahead period, $\kappa * \bar{p}$, for all schedulable jobs, often forces the more expensive jobs to be done before they are due. One possible precaution for this is to make the look-ahead parameter of job i, κ_i, inversely proportioal to its weight, w_i. This property can be tested in lightly loaded shops. The length of an efficient look-ahead has been shown in experimental simulations to increase with higher load [62]. This simple property is, however, quite difficult to establish analytically.

2.3.2 The Adaptive Apparent Urgency Rule

Since the optimal length of a job's look-ahead depends on the number and parameter values of the competing jobs suggests that the performance of the Apparent Urgency rule could be improved by adjusting the parameter κ in equation (2.7) for each job separately. In the following, we use an iterative search to adapt the look-ahead parameter κ to the problem data.[13] We illustrate the problem of finding an efficient combination of *static* κ_i-values with a static one-machine example:

Job# i	proc.time	due date	value	Optimal k values (average of 5 runs)
1	2	3	3	.7
2	2	5	1	2.2
3	2	6	3	.6
4	2	9	1	2.6
5	1	9	3	.7
6	4	10	4	.9
7	4	17	3	.5

The weighted tardiness of a WSPT schedule is 18, and application of EDD rule results in 12. The weighted tardiness with the Apparent Urgency rule, applying the priority index in equation (2.7) with look-ahead parameter $\kappa = 2$ is 14. Shortening the look-head to $\kappa = 1$ improves the performance to 10. The best weighted tardiness achievable is 9 with a constant $\kappa = 0.8$ for all jobs. The optimal solution 8 can be found by inspection. The Gantt chart for the optimal schedule is

[13] The properties discussed above indicate how the look-ahead should be adjusted in an average sense. These results will be tested in chapter 6 in general job shop scheduling by adjusting the κ parameter directly on the basis of the average queue length and the weight of the job on a machine.

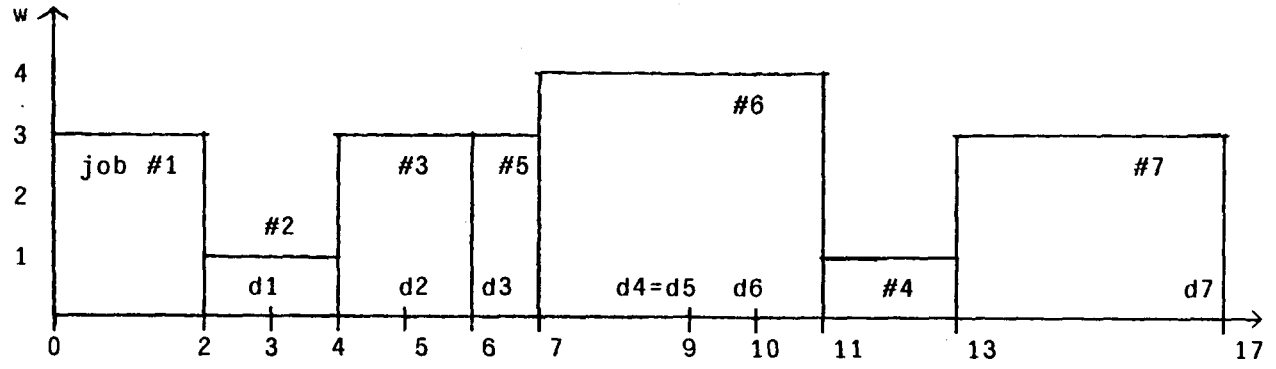


Figure 2-2: The optimal schedule of the example as a Gantt chart.

shown in Figure 2-2.This solution can also be found through an *adaptive apparent urgency rule*. This rule is based on an iterative adjustment of the look-ahead parameter κ as follows[14] :

1. Run one simulation run, using some initial value $\kappa_k = \kappa_{int}$ for all jobs.

2. Examine the resulting schedule: For each job that is late, increase the look-ahead, and for the jobs that are extremely early, decrease the κ value, both proportionally to the tardiness/earliness of the respective jobs.

3. Repeat the simulation with these new κ-values assigned to the jobs.

With some rather arbitrary values for the adjustment parameters, this "look-ahead adaptation" found the optimal schedule within 1 - 8 iterations, starting from values $0.5 \leq \kappa_{int} \leq 2.0$.

We tested the look-ahead adaptation with the constant AU rule (with $\kappa = 2.0$) in some hard problems whose weighted tardiness was at least 5% higher than the optimal in an earlier experiment [61]. Our sample of 30 hard problems with both 20 jobs and 30 jobs was further classified according to the expected portion of tardy jobs.[15] The portion of tardy jobs in an average schedule, denoted τ below, was set to 0.2, 0.4 or 0.6, that is, 20%, 40%, or 60% of the jobs were expected to be late. The adaptation procedure was in detail as follows:[16]

- A tentative correction of the κ_i parameter for job i is determined after each dispatching simulation run from the following equation:

$$\Delta\kappa_i = |\kappa_{ext} - \kappa_i| \times s_i / s_{ref} , \qquad (2.8)$$

---

[14] Notice that in this adaptive schedule, the κ_i-parameters are constant for each job throughout an iteration.

[15] Less than 5% of the problems were hard. For details concerning the test load generation, see [61].

[16] In the preliminary test runs, the adaptation procedure was found to be quite insensitive to wide range of the parameter values explained in the text. As a possible modification of the procedure, we could adjust the look-ahead factor according to the relative weighted tardiness of the job in question. In our test, this modification had a quite small but improving effect on the performance of the rule.

where s_i is the lateness[17] of the job i from the results of the run, s_{ref} is a reference level for earliness or tardiness, experimentally set at 60*(average processing time). κ_{ext} is an upper bound of κ for a job that is late, or a lower bound for an early job (set to 3.0 and 0.3, respectively).

- The adjustment step size, $\Delta\kappa$, is further constrained by an upper bound of 0.2 for increasing and 0.1 for decreasing the κ's.

- The AU rule has two tie breakers, the due date and the value (penalty on being late).

We used a normalized measure of weighted tardiness, or the total weighted tardiness divided by the total processing time and the average weight of the jobs. This measure is independent of the problem size and the units of measurement of processing times and weights. Test results for the 20-job problems are displayed in table 2-1. The following notation has been used: WT_{opt} is the optimal weighted tardiness from [61], WT_{con} the weighted tardiness with constant κ, from [61], and WT_{ad} is the weighted tardiness obtained after the adaptation. From these measures, the following ratios were computed:

Average absolute improvement = $Av(WT_{con} - WT_{ad})$

Average % improvement from margin = $Av((WT_{con} - WT_{ad})/(WT_{con} - WT_{opt}))$

Average improved value relative to the optimal = $Av((WT_{ad})/ WT_{opt})$.

The average optimal normalized weighted tardiness measures were approximately 0.035, 0.30, and 0.84 for the three different load levels, respectively. Another 30 problems with 30 jobs each were selected similarly from the test material in [61]. The results are shown in table 2-2.

Table 2-1: The improvement of the performance of the new Apparent Urgency rule using the look-ahead adaptation procedure for the 20-job problems.

τ (#of probl.)	Av. optimal tardiness	Av. absolute improvement	Av. % improv. from margin	Av. improved rel. tardiness	Av. # of iter.
0.2 (10)	0.035	0.034 (100%)	78%	1.31	7.4
0.4 (12)	0.30	0.058 (20%)	65%	1.14	6.7
0.6 (8)	0.84	0.076 (10%)	66%	1.05	7.1
Total (30)		0.055	69%	1.17	7.0

Based on this small sample, the look-ahead adaptation improves relatively more the solutions of the slack problems, but the absolute saving is more in the tardier problems with higher τ values. Often some of this improvement could be captured through an appropriate constant value of κ, using a shorter look-ahead, $\kappa = 0.5 \cdot 1.0$, in a slack shop and a longer one, $\kappa = 1.5 \cdot 3.0$, in a

[17] The earliness of an early job is first reduced by one average processing time in order to avoid overreacting.

Table 2-2: The improvement of the performance of the Apparent Urgency rule using the look-ahead adaptation procedure for the 30-job problems. The improvement with a pairwise interchange procedure is given below in parenthesis for comparison.

τ (#of probl.)	Av. optimal tardiness	Av. absolute improvement	Av. % improv. from margin	Av. improved rel. tardiness	Av. # of iter.
0.2 (9)	0.028	0.021 (75%) (0.019)	62% (56%)	1.61 (1.76)	5.3
0.4 (17)	0.189	0.015 (8%) (0.031)	21% (45%)	1.41 (1.25)	4.8
0.6 (4)	1.71	0.040 (3%) (0.020)	25% (17%)	1.06 (1,067)	5.0
Total (30)		0.020 (0.027)	34% (44%)	1.42 (1.36)	5.0

heavily loaded shop. In table 2-2, we have also given the corresponding results using a pairwise interchange procedure that improves the same initial sequence. In this sequence, the condition (2.5) is checked for any adjacent jobs. If it fails, the jobs are interchanged. The exact interchange algorithm is as follows:

1. Start with the initial sequence of the n jobs generated by the Apparent Urgency rule with some constant value of look-ahead. Set $i = 1$.

2. Number the jobs according to the present sequence.

3. Check the condition (2.5) for jobs i and $i + 1$.

   a. If the condition is satisfied for the present sequence, set $i = i + 1$. Go to 4.

   b. If the condition is not satisfied, interchange the jobs i and $i + 1$. Set $i = i - 1$, and go to 2.

4. Test if done:

   a. If $i < n$, go to 3.

   b. If $i = n$, compute the weighted tardiness of the present sequence. Stop.

The improvement using the interchange algorithm is approximately the same as it is using the look-ahead iteration in these hard problems.

2.3.3 The Effects of Errors in Processing Time Estimates

Conway et. al. [17] discovered that the WSPT rule maintained 80% of its power in reducing the average flow time of a FCFS sequence when using processing time estimates with errors up to 100% of the actual processing times. Muth [55] studied the consequences of the errors in processing times for minimum makespan problems. Holloway and Nelson [40] designed a multi-pass adjusting procedure for minimizing the average tardiness in static job shops. Their heuristic search procedure, which allowed non-delay schedules, outperformed SPT, truncated SPT, S/OPN and EDD rules in three deterministic problems. The introduction of errors in processing time estimates affected most their heuristic multi-pass procedure, although a non-delay version of it was more robust in the stochastic problems. The other rules had less pronounced deterioration of performance.[18] When scheduling with the Apparent Urgency rule, several factors can be affected if some erroneous processing time estimates, $\tilde{p}_i$, are applied instead of the actual processing times, p_i, in computing the index values in equation (2.7). First, the w_i/p_i part of the index changes the same way as in WSPT scheduling, and this bias is opposite to the error in the processing time estimate. Second, a change of the slack term, d_i-t-p_i, changes the priority index of a slack job. Third, the accumulation of the errors in processing time estimates can shift the time t of the simulated dispatching decisions biasing all slack estimates. However, in a one-pass dispatching routine the actual processing times will be observed before the next decision is made. More formally, the effect of processing time estimation errors, denoted dp, upon the numerical value of the exponential version of the Apparent Urgency index can be derived through differentiation. From equation (2.7), $\pi = (w/p)e^{-(d-t-p)^+/\kappa\bar{p}}$, we get by total differentiation, assuming that $d\text{-}t\text{-}p > 0$ and $\bar{p}$ = constant:

$$d\pi = (\partial\pi/\partial p)\,dp = (1/\kappa\bar{p} - 1/p)\,\pi\,dp\,, \tag{2.9}$$

The relative error of the AU priority index, $d\pi/\pi$, amounts to:

$$d\pi/\pi = (p/\kappa\bar{p} - 1)\,dp/p\,. \tag{2.10}$$

These errors in the Apparent Urgency indexes are rather small for processing times relatively close to the average look-ahead period, $\kappa\bar{p}$. For a tardy job, the relative error of π is -dp/p. We ran a simulation study to see the real effect of errors in processing time estimates on the performance of the AU rule and its adaptive modification in the single machine case. The problems were generated randomly. The actual processing times were taken from a uniform distribution $p_i \sim U[5, 25]$. The weights of the jobs were then generated based on the processing time p_i from the uniform distribution $w_i \sim U[1, 2p_i]$. The due dates were set with the tardiness factor $\tau = 0.5$. This means, that the due dates, d_i, are distributed as $d_i \sim U[\bar{d}(1\text{-}r), \bar{d}(1+r)]$, where the parameter r, $0<r<1$, determines the dispersion of the due dates around the average due date $\bar{d} = \Sigma p_i(1\text{-}\tau)$. In the study we used a value of r = 0.8.

The experimental design consisted of five different loads of 20 jobs each. We generated erroneous data on three levels of deviations from the actual processing times. The erroneous processing times, p_i^e, were taken from a uniform distribution $p_i^e \sim U[(1\text{-}E)p_i, (1+E)p_i]$, where E,

---

[18] Holloway and Nelson rejected the use of parametrized rules, such as COVERT, Dynamic Composite rule, etc. because of the difficulty to choose robust parameter values in the static problems. Most of the parametrized rules have been tested in stationary dynamic shop environments.

the maximal absolute error, was given the values 0.3, 0.6 and 0.9, i.e. the maximal errors were allowed to be 30%, 60% or 90%, respectively, from the actual processing times. Five sets of garbled processing time data were generated for each actual load, a total of 25 problems. Two common rules, WSPT and EDD, were used as benchmarks. The WSPT rule is known to be rather robust with erroneous processing time data, and the EDD rule does not use processing time information. The simulation procedure was as follows. First we solved a problem with the actual processing times to get a deterministic benchmark. The resulting average normalized tardiness for the basic 5 problems is shown in the first column of Table 2-3 below. Then the dispatching decisions were made based on the erroneous estimates, and the actual processing times were implemented during the simulation. In the case of the iterative look-ahead adaptation explained in Section 2.3.2 above, actual processing times were used just during the final simulation. The performance with the three levels of maximal errors for each of the rules is also shown in Table 2-3: these normalized tardiness figures are averages of 25 runs.

Table 2-3: The effect of errors in processing time estimates on the normalized weighted tardiness performance of the WSPT, EDD and AU rules.

Rule	With the actual data	With errors of: Max. 30%	Max. 60%	Max. 90%
WSPT	1.155	1.147	1.312	1.422
EDD	.646	.646	.646	.646
AU w/constant look-ahead	.254	.259	.304	.344
AU w/look-ahead adaptation	.225	.240	.270	.318

From these results we can see that the rules are quite insensitive to the errors in processing time estimates. The performance of the rules remained within 7%, 20% and 36% of the results with actual data for errors up to 30%, 60% and 90%, respectively. The absolute deterioration of performance was considerably less for the AU rules than for WSPT in the cases with large errors. The best rule, the AU rule with look-ahead adaptation, maintained its clear margin even for the highest level of errors. This result, which holds in small flow shops as well, is more encouraging than the results reported in [40]; the performance of the heuristic search procedure dropped in some cases even below the SPT and EDD levels.

We recorded also the performance of the rules in terms of two other measures, namely the number of tardy jobs (out of 20) and the maximal weighted tardiness cost for any one job. These results are shown in Table 2-4.

The new AU rules were again superior for the maximal weighted tardiness cost in the deterministic case and maintain their advantage despite the errors in processing time estimates. The AU rules were also better than the WSPT and EDD rules in terms of the number of tardy jobs

Table 2-4: The performance of the rules in terms of maximal weighted tardiness and the number of tardy jobs out of 20 (in parenthesis).

Rule	With the actual data	With the errors of: Max. 30%	Max. 60%	Max. 90%
WSPT	2142 (9.6)	2034 (9.8)	2338 (10.2)	2539 (10.0)
EDD	1487 (8.8)	1487 (8.8)	1487 (8.8)	1487 (8.8)
AU w/constant look-ahead	456 (7.0)	512 (7.0)	638 (6.7)	768 (6.7)
AU w/look-ahead adaptation	455 (7.0)	491 (6.7)	642 (6.3)	785 (6.4)

although these rules are known to be efficient with respect to this measure [3, 17]. Furthermore, the errors in processing time estimates did not impair the average performance of the AU rules.

The iterative adaptation of the parameters of a priority rule represents the planning of scheduling activity. We have shown that this planning can be consistently advantageous, even if the information available for planning is not perfect. The value of the incomplete and erroneous load information depends on the efficiency of the parameter setting for the final dispatching. But better information might be available later at the time of eventual dispatching. Hence in a stochastic environment, scheduling systems should address both the control problems and planning problems:

1. How well the rules perform assuming that there are some errors in the processing time estimates that cannot be detected before the job has been completed (the control problem of scheduling), and

2. How much the imperfect/incomplete load information that becomes more accurate after the job release dates can be used in the aggregate planning (the planning problem of scheduling).

The robustness of new AU rules against errors in processing time estimates indicated by our results is essential for the control of the scheduling activity in a dynamic environment. But the consistent performance of the AU rules could also help in linking the scheduling activity into the aggregate planning in a hierarchical production planning system, thus increasing the potential gains from scheduling information.

2.4 Conclusions

We have started with a new framework for systematic analysis of priority rules for scheduling. The search among the dispatching alternatives becomes more opportunistic when the rule uses more information of the status of the shop. But instead of explicit analysis of work contents or due date distribution of the load, we revise a heuristic, iterative procedure for adjusting the job-specific look-ahead parameter of the new Apparent Urgency rule. The performance feedback improves the weighted tardiness and portion of tardy jobs of the AU rule. The AU priority index is insensitive against errors in processing time estimates. The robustness of the basic AU rule and its look-ahead adaptation is demonstrated in a simulation experiment with stochastic processing times.

3. Scheduling in Proportionate Flow Shops

Summary

It is well known that except in the case of makespan problems, there are hardly any analytical results for flow shop problems. This chapter considers of a class of flow shop problems where job processing time at a machine is proportionate to the processing time on the first machine. We show that for the pre-emptive version of the problem, in order to minimize any regular measure of performance, it is sufficient to consider permutation schedules. Also, results for various other measures are derived. A characterization of the optimal solution for the weighted tardiness problem is derived which is analogous to its counterpart in the single machine case. It is indicated how this characteriation can be used to develop heuristics for flow shop problems.

3.1 Introduction

Flow shop problems have been the center of attention for researchers in Scheduling Theory for a long period of time. Though flow shop problems are a special case of general job shop problems, even these problems have proven themselves to be too complex to provide many analytic solutions. As has been established by Lenstra [45], most problems in this area fall in the NP-Complete class. There are no known polynomially bounded procedures for this class of problems and it is unlikely that there are any such procedures. Most prior research in the field of flow shop problems has been confined to makespan problems. The most widely quoted result is due to Johnson [41] to minimize makespan in two machine flow shop problem and its extension to a special case of three machine flow shop problem. Also, Gilmore and Gomory [29] devised an algorithm with a computational burden of $O(n^2)$ for the two machine flow shop problem where job waiting is not permitted. There are hardly any other known polynomially bounded procedures for the problems in flow shops. Another widely quoted result, due to Conway, Maxwell and Miller [17], proves the optimality of the same permutation sequence on the first two machines in a flow shop for any regular measure of performance and the additional result that the sequence on last two machines is the same for makespan problems. The fact that these results were discovered more than two decades ago and no further significant progress has been made in the case of flow shop problems in deriving analytical attests to the complexity of these problems. Most of the recent research in flow shops has been largely directed towards finding optimal solution using enumerative methods such as branch and bound or developing "good" heuristics for makespan problems [11, 12, 14, 18, 35, 47, 50, 57]. There is hardly any significant work done for other important measures of performance.

This chapter addresses scheduling problems in the context of a particular kind of flow shop where the task processing time of any job at a machine is proportionate to the processing time on the first machine. Results derived in this chapter relate to the problems where the jobs can be pre-empted or divided into parts of unit duration without penalty. We show that in such a case, permutation schedules constitute the set of dominant schedules for any regular measure of performance and we further derive results for performance measures based on completion times and/or the due dates of the jobs. These results hold good even in cases where job-passing is prohibited. In case of shops where intermediate queues are prohibited (once a job is begun on the

first machine, it has to be processed without interruption at any subsequent machine), these results hold good except that the start times on the first machine have to be appropriately delayed.

3.2 Permutation Schedules for the Proportionate Flow Shops

In this section, we consider the pre-emptive version of the general problem for the proportionate flow shop problem. We wish to schedule a set of jobs, $\{J_1,J_2,J_3,...j_n\}$ so as to minimize a regular measure of performance. It is not unusual to find jobs being pre-empted in practice in order to expedite them through the production system. Also, the pre-emptive case is an important relaxation of the original problem from the computational point of view. The following proposition holds good for the pre-emptive case.

PROPOSITION I : To minimize any regular measure of performance, it is sufficient to consider permutation schedules.

Proof : Consider an optimal schedule in which the ordering of jobs is not the same on the last two machines m-1 and m. Consider any two jobs J_i and J_j such that $J_i < J_j$ on machine m and $J_j < J_i$ on machine m-1. Since all jobs have the same processing time on any particular machine, pairwise interchange of any two jobs on a particular machine does not affect the completion times of any other jobs on that particular machine. So, pairwise interchange of jobs J_i and J_j on machine m-1 does not affect completion time of any other job on machine m-1. If such pairwise interchange on machine m-1 is forbidden by the schedule on machine m-2, we can switch jobs J_i and J_j on machine m-2 as well and so on back to the first machine. Thus, we can always form an optimal schedule in which machines m-1 and m have the same sequence and completion times of jobs on machine m are no greater than the original given optimal schedule. Now, we extend the same argument inductively between machines m-1 and m-2 , m-2 and m-3,..., 2 and 1. Since the completion times of the jobs are no greater than the completion times in the original schedule, permutation schedules constitute the set of dominant schedules for any regular measure of performance. □

Now we derive some results relating to the completion times of the jobs. Let p_k represent the processing time for any job(piece) on machine k. Consider any permutation schedule. Let $C^k_{[i]}$ represent the completion time for the piece in the i th position on machine k. The following result holds:

PROPOSITION II : For any piece,

$$C^j_{[i]} = \Sigma_{l=1}^{l=j} p_l + (i-1) \max_{q=1,2..j} \{p_q\}.$$

PROOF : In a permutation schedule, same sequence is used on all machines. Hence

$$C^{j}_{[1]} = \Sigma^{l=j}_{l=1} p_l.$$

The rest of the proof is by induction. Suppose that in a permutation schedule $C^{k}_{[i]} - C^{k}_{[i-1]} = D^k$ for some particular machine k (this is obviously true for k = 1 and i = 2,3,...,n). We show that $C^{k+1}_{[i]} - C^{k+1}_{[i-1]} = D^{k+1}$ where D^{k+1} is a constant and is given by $\max(D^k, p_{k+1})$.
We have two cases to consider: 1) $p_{k+1} \geq D^k$ and 2) $p_{k+1} < D^k$.

Case 1 : $p_{k+1} \geq D^k$

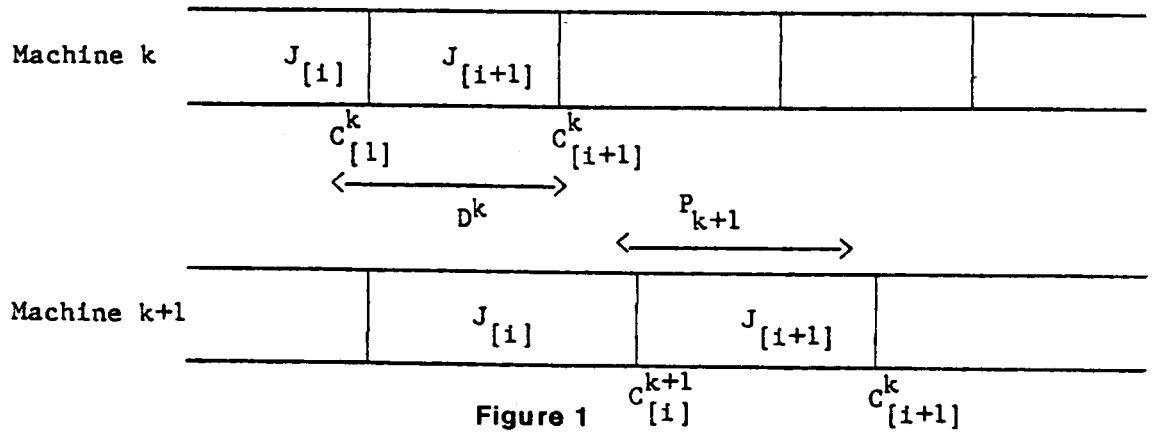


Figure 1

In this case, there is no idle time on machine k + 1. Therefore, $C^{k+1}_{[i]} - C^{k+1}_{[i-1]} = p_{k+1} = \max(D^k, p_{k+1})$.

Case 2 : $p_{k+1} < D^k$

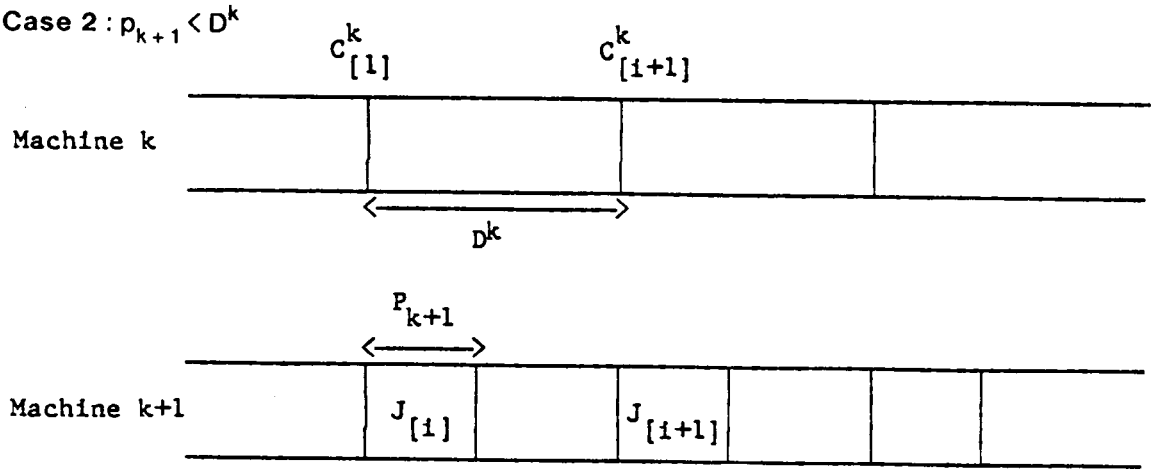


Figure 2

In this case, $C^{k+1}_{[i]} - C^{k+1}_{[i-1]} = D^k$

$= \max\{D^k, p_{k+1}\}$

Thus, $C^j_{[i]} = C^j_{[1]} + \Sigma^{i=j}_{i=2}\{ C^j_{[i]} - C^j_{[i-1]} \}$

$= C^j_{[1]} + (i-1) \max \{ D^{j-1}, p_j \}$

$= C^j_{[1]} + (i-1) \max_{q=1,..j} \{ p_q \}$ □

We had earlier indicated that makespan problems are the most widely researched area in the case of the flow shop problems. Further, it is well known that the optimal schedule need not necessarily be a permutation schedule except that the sequence is the same on the first two machines and also on the last two machines. However, when all jobs have equal processing times on the first machine, the following proposition holds good in the case of proportionate flow shop.

PROPOSITION III : Any permutation schedule provides the minimum makespan for the proportionate flow shop problem in the case of jobs with equal processing times on the first machine.

PROOF : Let p_{max} be the maximum processing time of a job on some machine. Work content at this particular machine is np_{max}. Also, every job has to undergo processing prior to and subsequent to this machine. Therefore, the minimum processing time for these operations is $\Sigma^{k=m}_{k=1} p_k - p_{max}$. Hence, the minimum makespan is given by

$$\Sigma^{k=m}_{k=1} p_k - p_{max} + np_{max} = \Sigma^{k=m}_{k=1} p_k + (n-1)p_{max} .$$

From Proposition II, it is clear that the minimum makespan is acheived by any permutation schedule and hence the result. □

Now we discuss some measures relating to the completion times of jobs in the case of the proportionate flow shop for jobs with equal processing times.

COROLLARY 1 : Any permutation schedule of the pieces minimizes F.

PROOF : F is a regular measure of performance and permutation schedules constitute the set of dominant schedules. From Proposition II, it is clear that all permutation schedules have same F.

$F = (1/n)*(\Sigma^{[i]=n}_{[i]=1} C^m_{[i]})$

$= \Sigma^{k=n}_{k=1} p_k + {}^1/_2(n-1)p_{max}$ □

COROLLARY 2 : F_w is minimized by scheduling the jobs according to the weighted shortest processing time rule.

PROOF : F_w is a regular measure of performance and we need to consider only permutation schedules. Completion times of jobs in a permutation schedule is given by

$$C^m_{[i]} = \Sigma_{k=1}^{k=m} p_k + (i-1)\, p_{max} \qquad \text{(application of Proposition II)}$$

$$F_w = (1/n)^* \{ \Sigma_{i=1}^{i=n} w_{[i]} C^m_{[i]} \}$$

It follows directly from basic algebra that the product of two series is minimized by arranging one in the ascending order and the other in non-ascending order. □

Just as in the single machine case, we can show in this case also that arranging the jobs in non-increasing order of the weights minimizes the weighted lateness as shown below :

COROLLARY 3: The Earliest Due Date rule minimizes maximum lateness and maximum tardiness.

PROOF: Consider any two adjacent jobs J_i and J_j in a given schedule such that $J_i < J_j$ and $d_i > d_j$. Let t be the completion time of J_i.

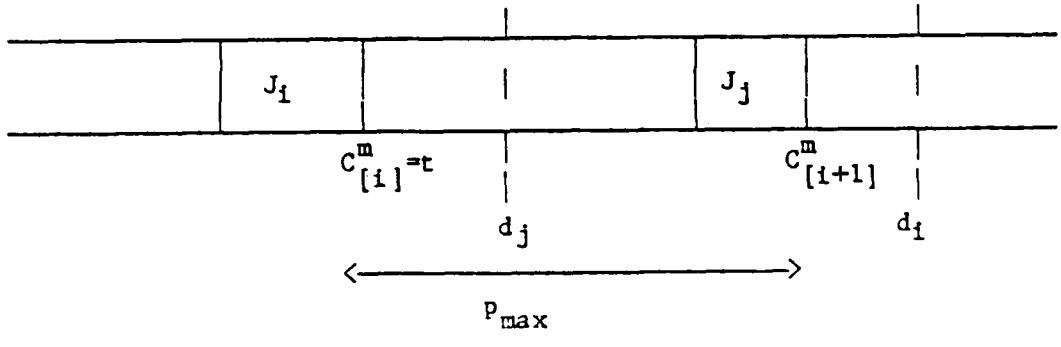


Figure 3

Maximum lateness among jobs J_i and J_j is given by

$$\max \{ t - d_i , t + p_{max} - d_j \} = t + p_{max} - d_j. \qquad (1)$$

Suppose we interchange J_i and J_j. Maximum lateness among J_i and J_j is given by

$$\max \{ t - d_j , t + p_{max} - d_i \}. \qquad (2)$$

It is clear that (1) > (2). Thus, by interchanging J_i and J_j, the schedule is no worse off and in fact, it would improve if the maximum lateness in the original sequence occured for J_j. Since T_{max} equals $\max(0, L_{max})$, the result holds good for maximum tardiness as well. □

Another important measure of performance is weighted average tardiness. Since this is a

regular measure of performance, it is sufficient to consider only permutation schedules. Following results relate to this measure of performance for jobs with equal processing times on the first machine in the case of proportionate flow shops.

PROPOSITION IV : The optimal pre-emptive solution to the $\Sigma w_i T_i$ problem is found by solving the linear assignment problem.

PROOF : It is clear from the Proposition II that $C^m_{[i]}$ is independent of the job occupying i^{th} position in the sequence. We can form the cost matrix tableau for the linear assignment problem (ψ_{ij} indicates the penalty incurred if J_j is in the i^{th} position in the sequence) as follows:

$$\psi_{ij} = w_j \max \{ 0, \Sigma_{k=1}^{k=m} p_k + (i-1)*p_{max} - d_j \}$$

Solving the linear assignment problem using the above cost tableau yields optimum solution. It may be noted that the solution procedure has a computational burden of the order of $O(n^3)$. □

In fact, the result in the Proposition IV can easily be generalized to any penalty function of the completion times of the jobs so long as they are nondecreasing functions of the completion times of the jobs and the performance measure is additive over the completion times of the jobs. Though Proposition IV provides us with a polynomially bounded procedure for solving the pre-emptive version of $\Sigma w_i T_i$ problem, the following characterization of optimal solution for the same problem is interesting from the point of view of developing heuristics for the flow shop problems.

PROPOSITION V : Consider an optimal sequence for $\Sigma w_i T_i$ problem for jobs with equal processing times on the first machine for the proportionate flow shop. Consider any two jobs, J_i and J_j, $i < j$ (without loss of generality, assume that job index is same as the locational index in the sequence under consideration). Then, the following property must be satisfied in an optimal sequence:

$$w_i \left\{ 1 - \frac{(di - C[i])^{m\ +}}{(j-i)\ p_{max}} \right\}^+ \geq w_j \left\{ 1 - \frac{(dj - C[i])^{m\ +}}{(j-i)\ P_{max}} \right\}^+$$

PROOF : The proof is similar to the proof provided in the appendix of an earlier paper on the myopic heuristics for the single machine tardiness problem [61] and is omitted here for the sake of brevity. □

This property can be considered to be valid for a relaxation of the general problem in proportionate flow shops where jobs are permitted to be preempted at unit intervals on the first machine and all such pre-empted pieces have the same due date as the original job. However, if all jobs have equal weights and equal processing times, then the earliest due date sequence provides an optimum sequence for the average tardiness problem as shown in the next proposition.

PROPOSITION VI : If all jobs (jobpieces) have equal weights, the earliest due date sequence minimizes the average tardiness.

PROOF : From Proposition I, it is clear that we have to consider only permutation schedules. Consider an optimal solution in which two successive jobs do not follow the earliest due date rule, i.e., $J_i < J_j$ and $d_i > d_j$.

Case 1 : Suppose that both J_i and J_j are early or on time. Since J_j is early or on time and $d_i > d_j$, pairwise interchange does not degrade the solution.

Case 2 : Both J_i and J_j are tardy. Pairwise interchange does not degrade the solution since the weights are equal.

Case 3 : J_i is tardy and J_j is early or on time. This is impossible since $d_i > d_j$ and $C^m_{[i]} < C^m_{[j]}$.

Case 4 : J_i is early or on time and J_j is tardy.

Subcase 4.1

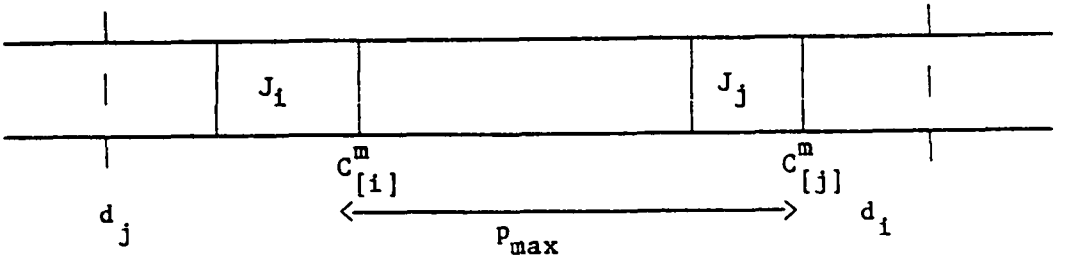


Figure 4

Clearly, pairwise interchange improves the solution.

Subcase 4.2

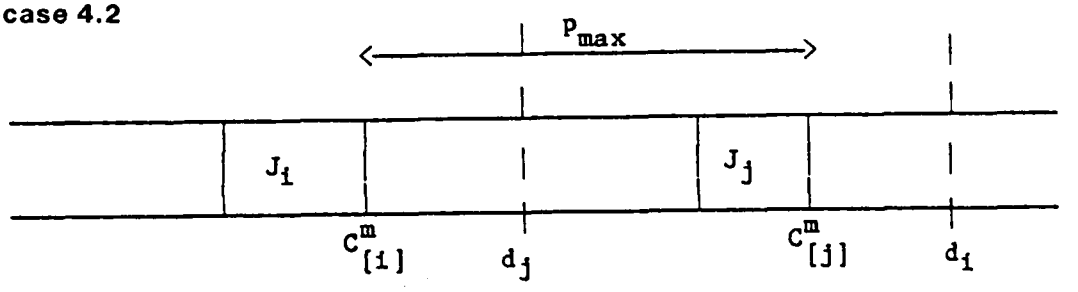


Figure 5

Clearly, pairwise interchange improves the solution.

Subcase 4.3

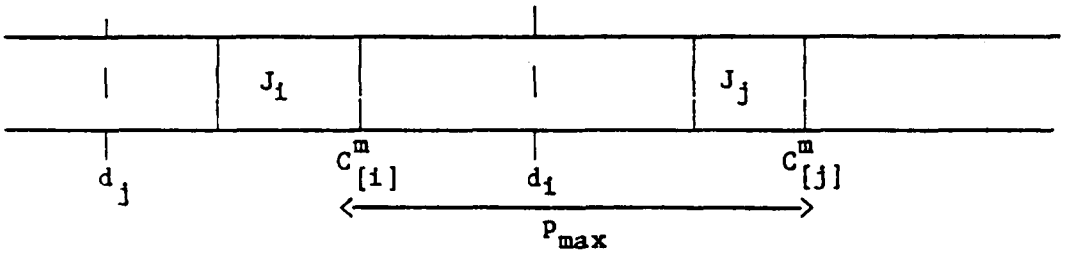


Figure 6

Cost of J_i and J_j in given schedule $= C^m_{[i]} + p_{max} \cdot d_j$ (3)

Cost after interchange $= (C^m_{[i]} + p_{max} - d_j) + (C^m_{[i]} - d_j)$ (4)

Subtracting (4) from (3), $= d_i - C^m_{[i]} > 0.$

Therefore, pairwise interchange results in an improvement.

Subcase 4.4

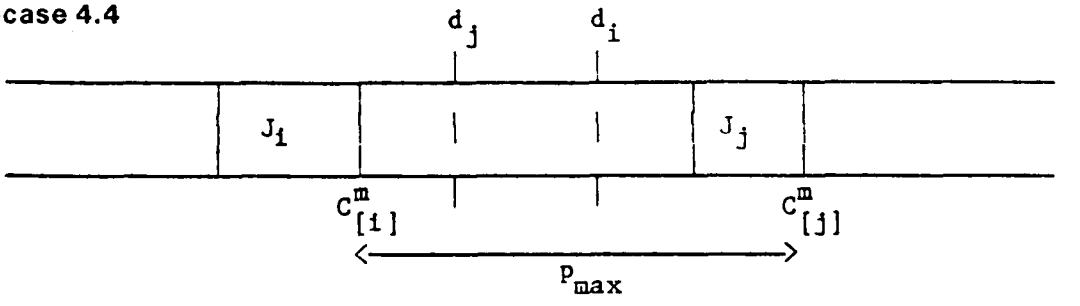


Figure 7

Cost of J_i and J_j in given schedule $= C^m_{[i]} + p_{max} \cdot d_j$ (5)

Cost after interchange $= C^m_{[i]} + p_{max} \cdot d_i$ (6)

Since $d_i > d_j$, (5) > (6). Therefore, pairwise interchange improves the solution.
Thus, in all cases, pairwise interchange does not degrade the solution and, in fact, may improve it. Since our arguments employ only information about the individual jobs and not the location, ensurance of local optimum at all locations in the sequence ensures global optimum and hence the earliest due date rule is optimal. □

3.3 Schedules with No Job-passing

There is a special class of flow shop problems where no job-passing is permitted. That is, once a job is begun on the first machine, it maintains same priority relative to other jobs for subsequent processing on any other machine. No job-passing is a matter of practical and design

expediency. As stated by King [43], *'this is typically the situation in many manufacturing plants where jobs are moved from station to station by conveyor'*. Even in Flexible Manufacturing Systems, due to problems involved in computation of optimal resouce utilization, not more than two or three jobs are permitted to pass the others in the sequence [38, 39]. Also, since technologically designing input buffers to machines to accomodate any scheme other than *First Come, First Served* is rather complex, in many situations no job-passing restriction is used. In case of proportionate flow shops, the following remark holds good.

REMARK 1: Permutation Schedules < = > Schedules with no job-passing

Hence all results derived in section 2 equally hold good for jobs with equal processing times in proportionate flow shops.

3.4 Schedules with no Job-waiting

Another special class of flow shop problems are those where job waiting is forbidden. Once a job is begun on the first machine, it must be processed with no waiting at any other machine. Steelmaking is an example of such a situation [43, 71]. It is clear that schedules with no job-waiting are a subset of schedules with no job-passing. So, here again, it suffices to consider only permutation schedules for optimizing any regular measure of performance. But, due to the no-wait condition, it would be necessary have inserted idle time on the first machine. An exact algorithm for minimizing makespan for the case of two machines with no job-wait is given by Gilmore and Gomory [29]. Wismer [71] has shown that the makespan problem for general flow shop problem with no job-waiting can be translated into an equivalent Asymmetric Traveling Salesman Problem. Lenstra [45] has shown that the Hamiltonian Path problem is reducible to makespan problem in flow shops with no job-wait, thus establishing the latter problem to be NP-Complete. King and Spachis [43] developed heuristics for this problem and tested them against random sequences and other heuristics.

However, in the case of jobs with equal processing times to be processed in proportionate flow shop, we can easily extend the results obtained in section 2 even for situations where job-waiting is not permitted.

PROPOSITION VII : Any permutation sequence for proportionate flow shop (all jobs with equal processing times on the first machine) can be scheduled so that completion times on the last machine are not changed and the jobs do not form queues at any machine.

PROOF : Consider two adjacent jobs, J_i and J_{i+1}. Suppose J_i starts on machine 1 at time t. Then $C^m_{[i]} = t + \Sigma_{q=1}^{q=m} p_q$.

J_{i+1} can start on machine 1 only at such a time that once its processing has begun, it does not have to wait at any other mchine. In order to determine when J_{i+1} complete on machine m, we simply left shift J_{i+1} such that its processing on machine m can begin immediately after J_i is complete on machine m and then right shift it to the minimum possible extent to make it feasible. Now $C^j_{[i]} = t + \Sigma_{q=1}^{q=j} p_q$.

Overlap of J_i and J_{i+1} on machine j $= \max \{ 0, C^j_i - [C^m_{[i]} + p_m - \Sigma_{q=j}^{q=m} p_q] \}$

$= \max \{0, p_j - p_m \}$

Therefore, time difference between completion times of two successive jobs on machine m is given by $p_{max} - p_m + p_m = p_{max}$. Thus, the completion time of i^{th} job on machine m, $C^m_{[i]}$, is given by $C^m_{[i]} = C^m_{[1]} + (i-1)*p_{max}$.

We note that this value is same as the one derived in Proposition II with no constraints on job-waiting. Thus, all the results derived in section 2 hold good even in the case when job-waiting is prohibited. However, the start times on the first machine will be delayed so that there are no queues at intermediate machines. The start time for the job in the i^{th} position is given by

$= C^m_{[1]} + (i-1)*p_{max} - \Sigma_{q=1}^{q=m} p_q$

$= (i-1)*p_{max}$ □

3.5 Conclusion

There are hardly any previous analytic results for flow shop problems except in the case of the makespan problem. We have derived such results for the situation where job processing times at any machine are proportionate to the time on the first machine. We considered the case where jobs are permitted to be preempted; these results may be used for developing lower bounds for non-preemptive cases. Also, the property developed for characterizing an optimal solution for the weighted tardiness problem can be used for developing heuristics for the flow shop problems. Our investigations in this direction have been promising.

4. Lead Time Iteration in Flow Shop Scheduling

Summary

This chapter studies the development of state dependent priority dispatching rules for flow shops. We demonstrate the advantage of using indirect state information, more specifically the lead time estimates of the jobs, to determine their priority index values. The theory of optimal lead time estimation is advanced in the special case of proportionate flow shops with with unit jobs. In this case, we can give an analytical characterization of the optimal sequence on the first machine using optimal lead time estimates. The same procedure can be repeated at all subsequent machines, using operation due dates computed from the optimal lead time estimates. Operation due dates can be used as coordination mechanism in the scheduling under any regular measure of performance, not only in scheduling against externally provided job due dates. Lead time estimates that are erroneous can be used insofar the errors do not exceed given bounds.

New versions of Apparent Urgency and CoverT dispatching rules are proposed for general flow shop scheduling. The new rules use performance feedback, obtained through an iterative procedure, in lead time estimation. In the computational study reported in this chapter, the Apparent Urgency rule performed better than any other rule tested even with constant lead time estimates based on multiples of the job processing times. Application of the lead time iteration procedure can improve considerably the weighted tardiness performance of the look-ahead rules. More accurate lead time estimation improves the overall coordination of the local dispatching decisions, resulting in robust performance also in terms of several important secondary measures of performance, including number of tardy jobs, maximum weighted tardiness, and work in process inventory.

4.1 Introduction and Summary

4.1.1 The Need for Lead Time Estimation

In a flow shop, jobs have operations on several machines, and consequently the total time a job spends in the process, or its *lead time*, can vary depending on the load in the shop and the priority rules used in scheduling. If the objective is to minimize some due date based criterion, such as weighted tardiness, it would be useful to know when a job should be completed on a particular machine in order to get through all its remaining operations on time, i.e., we need to determine some appropriate *operation due date* for each job on each machine.[1] The due date of an operation can be obtained by subtracting the remaining lead time, that is the sum of processing times and waiting times of the remaining operations after the current one, from the external due date of the job. Thus :

(operation due date) = (job due date) - (lead time for the remaining operations)

---

[1] The operation due date is not, however, the only way to determine the expected tardiness cost of a job as a function of time. Alternative formulations, such as CoverT rules, will be discussed later on.

A common practice in the dispatching approach has been to use some constant estimate for the lead time, based on experience and/or the attributes of the job such as the total processing time. The waiting time estimates, or the operation due dates obtained from the lead time estimates, can then be used in a myopic heuristic to determine the local sequencing priorities.

The purpose of this chapter is to develop a flow shop version of the Apparent Urgency rule tested earlier in scheduling one-stage processes under the weighted tardiness criterion, see [61, 62] and chapter 2 above. The new rule uses local operation due dates. The look-ahead of the AU rule means that a job's priority index, or its anticipated tardiness cost per its processing time, is increased gradually with diminishing slack until the job reaches its due date. In a flow shop, the due dates of the operations other than the last one are determined as discussed above. In order to provide good estimates of the lead times, we revise them by successive simulation over a forecasting horizon. By applying the best waiting time estimates obtained during this iterative procedure in the final dispatching, we can further improve the weighted tardiness performance of the myopic AU and CoverT rules by 5-20% in static flow shop problems.

4.1.2 Related Heuristic Approaches

Many commonly used scheduling heuristics avoid the necessity of determining the operation due dates and the use of lead time estimates. One of the most efficient scheduling rules even in tardiness related problems is WSPT, the Weighted Shortest Processing Time rule [3, 17]. WSPT is a static rule and does not use any information regarding the external due dates of the jobs. Some other heuristics use the time remaining until the due date, or the slack of job, as basis for the priority index. Examples of these rules are [3, 15, 17]:

1. External (Job) Due Date: The priority index of the job on any machine is the same as the job due date. The job with the earliest due date is started first.

2. Operation Due Date: In this case, the the due dates for the operations are set by equally allocating the available time among all remaining operations when the job arrives at the machine center. The job with the earliest operation due date is started first.

3. Static Slack Time: The job priority index is the static slack of the job, i. e. the difference between the due date and the time of arrival at the machine center. The job with the least slack is started first.

4. Static Slack per Operation: This priority index that is the static slack of the job divided by the number of remaining operations. In the case of a flow shop in which all jobs go through the same sequence of machines this rule is effectively the same as Slack Time priority rule.

5. Static Slack per Operation Time: The priority index is the static slack of the job divided by the remaining processing time.

Dynamic rules use more up-to-date information about the time remaining until due date at the time of dispatching decision. Simple dynamic rules based on due dates often use the operation time remaining as a primitive lead time estimate. Examples of these rules are:

1. Dynamic Slack Time, Dynamic Slack per Operation, or Dynamic Slack per Processing Time remaining [15]: The priority index is the dynamic slack, or the time until due date less estimated processing time of the remaining operations, divided either by one, the number of the operations remaining, or the total processing time of the remaining operations.

From the discussion and experimental testing to follow we shall see that the rules listed above, although widely used in industry, do not in general provide good solutions for weighted tardiness problems. Some of the more elaborate heuristic procedures require an *a priori* estimate of the lead time on the remaining operations either for determining "normal" waiting times or for setting a local operation due dates. Examples of these rules are:

1. Critical Ratio: The priority index is the ratio of available time until due date over the standard lead time, i. e. the estimated queue and work time for the remaining operations [15].

2. CoverT rules: The relative length of global slack constitutes the basic information in a parametric family of scheduling rules called CoverT [3, 15, 16, 51]. Job j's priority index is based on the projected (unweighted) tardiness cost, c_j , and its processing time, p_j, on the machine in question. If d_j denotes the external due date, the priority index of job j at time t, π_{CoverT}, is:[2]

$$\pi_{CoverT}(t) = c_j(t)/p_j = (1/p_j)[k\,W_j - (d_j - r_j - t)^+]^+/(k\,W_j)\,, \qquad (4.1)$$

where W_j denotes the expected waiting time and r_j the processing time for the remaining operations. If the total slack $d_j - r_j - t \leq 0$, the anticipated tardiness cost is set to $c_j = 1$; if the slack exceeds the "standard" waiting time $k\,W_j$, the cost is set to $c_j = 0$. Hence the parameter k, $k > 1.0$, increases the length of the slack for which $c_j > 0$ to provide worst case tolerance beyond the average waiting times.[3] Note that CoverT does not set any operation due dates.

3. Dynamic Composite Rule: The most complex rule used for the average tardiness studies is called Dynamic Composite Rule (DCR), a 3-parameter state dependent rule [17]. The priority index is determined by the due date and the processing time of the operation in question, adjusted by terms dependent on the congestion the current and the next machine the job will need. The operation due date is set accordind to the standard lead time of the job class in question. See chapters 2 and 6 for more details.

A new myopic rule for weighted tardiness scheduling is the Apparent Urgency rule discussed in chapter 2. This rule applies a look-ahead feature, i. e. the anticipated tardiness cost of a job increases smoothly with decreasing positive slack. An exponential form of the look-ahead was found to be most efficient [61, 62]:

$$\pi_i(t) = (w_i/p_i)\, e^{-(d_i - t - p_i)^+/\kappa \bar{p}} \qquad (4.2)$$

---

[2] Here we use notation $(x)^+ = \max(0, x)$.

[3] Some standard waiting times, determined during a reasonably high load period, were used originally with CoverT. In the case of a light load, these standards had to be adjusted downward by using another parameter Q, $Q \leq 1.0$ [15, 16]. See also chapters 2 and 5.

Here $\bar{p}$ is the average processing time of the jobs, and w_i is the weight of job i. The slack of of a job is normalized by $\kappa\bar{p}$ that determines the rate of exponential decrease of the expected urgency value from a maximum of w_i/p_i (the WSPT priority index) for a tardy job. In the earlier experiments, the value of the look-ahead parameter $\kappa = 2.0$ performed close to optimal in a wide variety of one machine problems.

There are no previous studies in weighted tardiness scheduling in static flow shops [33]. From the literature reviewed above we know that due date based rules are efficient in lightly loaded shops in which due date setting is not very tight. However, the performance of these rules deteriorates quickly with increasing load or tightness of the due dates. WSPT has been a very robust rule in a wide variety of shop conditions, but it is usually inefficient in the problems in which due date rules are most efficient. CoverT works well in reducing the average (nonweighted) tardiness in job shop environments [3, 15, 16], but CoverT has not been tested previously in weighted tardiness problems, or in static flow shops, or against some state dependent rule, such as the Dynamic Composite Rule. In this chapter, we implement the original one-stage AU rule in a flow shop environment. The basic problem in modifying the rule for a multi-machine case is discussed above: how to set appropriate due dates for the operations preceding the last operation. In order to improve the due date setting in the actual shop situation, we use lead time estimates that reflect the effects of the load on job's realized waiting times in the subsequent queues. This is achieved by using an iterative procedure for adjusting the waiting time estimates according to the waiting times realized in a previous simulation with the same load. If the expectations concerning the required waiting times are consistent with realized ones, an equilibrium condition, the rule should exhibit a superior performance. In the terms of the framework developed in chapter 2, the lead time estimates serve as global coordination parameters that convey indirectly the feedback concerning the rule's performance.

4.1.3 Chapter Summary

The rest of this chapter is organized as follows. Based on the approach introduced in chapter 3, or [63], it is shown in the next section that the optimal lead time estimates for any regular measure of performance can be determined analytically in the case of a proportionate flow shop with unit jobs. Then, using estimates close enough to these optimal lead times to determine operation due dates, there exists a locally optimal dispatching discipline that yields also a globally optimal solution. Thus the problem may be decomposed and solved separately at each machine. The procedure is shown to be insensitive to small errors. In more general flow shop environments, optimal lead time estimation is impossible but efficient heuristics are developed for weighted tardiness scheduling based on the concepts of adaptive state dependent dispatching rules discussed in [67] and chapter 2 above.

In section 3, we report a large computational study testing the performance of the new AU rule, in its basic version and with the iterated lead time estimates, against several well-known myopic rules including FCFS, WSPT, EDD, Slack per Remaining Processing Time (S/RPT), and CoverT (a "weighted" version of the original rule in [16]), in static flow shop scheduling problems having up to 60 jobs and 8 machines. The results of this study indicate that AU, even without lead time iteration, outperforms the other rules clearly with respect to the weighted tardiness criterion

in an average of 1280 different randomly generated problems. The use of better lead time estimates, obtained through iteration, improved the performance of the new AU rule 5-10% in the average. The final weighted tardiness cost achieved by the iterative AU rule was in average 10-30% lower than that of CoverT for different problem sizes and shop layouts. By modifying CoverT to use the lead time iteration as well we can get within 5% of the average results achieved with the iterative AU rule. Furthermore, the lead time adaptation of the AU rule yields a robust performance in a wide variety of flow shop conditions and for several secondary measures of performance, including maximum weighted tardiness, number of tardy jobs and work-in-process inventories. In particular, the AU rule averages the lowest number of tardy jobs. Conclusions of this chapter can be found in section 4.

4.2 Lead Time Estimation in Flow Shops

4.2.1 Due Date Setting in Proportionate Flow Shops with Unit Jobs

In order to make the flow shop problem amenable to analytical study, we structure the scheduling environment under consideration with some simplifying assumptions. The concept of permutation schedule is useful in reducing the number of different sequences to be explored especially in flow shop problems [3, 17].

Definition 4.2-1: A *permutation schedule* is a schedule in which the jobs are processed in the same sequence on all machines.

For several special classes of problems, permutation schedules constitute a dominant set of schedules. We are particularly interested in a common flow shop structure that is called "proportionate" in chapter 3.

Definition 4.2-2: A *proportionate*, (or uniform[4]), flow shop is one in which the processing times of the jobs on different machines are constant multiples of some standard, e.g. the processing times of the operations on the first machine.

Proportionate flow shops represent an extreme of the rather realistic assumption that the processing time of an operation depends on the size of the job (a manufacturing order) in number of units and the relative speed per unit of the machine performing the operation.[5] In practice, we can expect to observe at least some uniformity in a job's operation times.

The concepts of lead time and operation due date are important for the development of myopic heuristics for scheduling. The "real" due date of the last operation of a job is the economically preferred completion date of the job in the context of production coordination. The first surrogate of this job due date is the external (promised) due date. Some dispatching heuristics use the surrogates of operation due dates for priority index computations as discussed in Introduction, but there has been also some experiments in applying the projected or possible completion date as the basis for the EDD priority index, see e.g. Baker [4, 28]. The basic idea of the look-ahead heuristics, such as AU and CoverT, is that the value of the priority index of a job

---

[4]The term "uniform" is used particularly in the problems with parallel machines, see [33].

[5]An "ordered" flow shop would be one in which the processing times can be rank ordered in the same order on each machine, but the requirement of exact proportionateness is relaxed.

increases smoothly *before* the job is due on a machine. Thus there is no need to assign any artificial job due dates.

Consider the proportionate flow shop problem for jobs with equal processing times on the first machine.[6] Suppose that the objective is to minimize the total tardiness cost, $\Sigma_i C_i(t_i)$, when the penalty associated with the completion of job i at time t_i, $C_i(t_i)$, is given as:

$$C_i(t_i) = w_i \max \{0, (t_i - d_i)\} . \tag{4.3}$$

Without loss of generality, assume that jobs are indexed according to the order in an optimal sequence for the problem that we have already shown to be a permutation schedule in chapter 3.

Proposition 4.2-1: If the lead times for setting the operation due dates are optimally chosen, then an optimal local assignment of priorities leads to a globally optimal schedule.

Proof: Consider the following problem. Jobs $J_1, J_2, \ldots, J_n$ have weights $w_1, w_2, \ldots, w_n$ and artificial due dates $p_1, p_1 + p_{max}, p_1 + 2\,p_{max}, \ldots$, where p_{max} is the processing time on the slowest machine. We want to show that there exists a sequence which minimizes $\Sigma_{i=1}^{i=n} w_i T_i$ on machine 1 alone which is same as a globally optimal sequence.

If the completion date of the job scheduled at i^{th} position on machine m is $c_{[i]}^m = \Sigma_{q=1}^{q=m} p_q + (i - 1)p_{max}$, as derived in chapter 3, then the optimal completion time at machine 1, t_c, is

$$t_c \leq c_{[i]}^m - \Sigma_{q=2}^{q=m} p_q \tag{4.4}$$

$$\leq (i - 1)\, p_{max} + p_1 .$$

From the definition of the problem, the optimal sequence is $J_1, J_2, \ldots, J_n$. Consider the value of the same sequence on machine 1 :

$$w_1 T_1 = w_1 \max \{0, p_1 - p_1\} = 0 ,$$
$$w_2 T_2 = w_2 \max\{0, 2\,p_1 - p_1 - p_{max}\} = 0 ,$$
$$\ldots$$
$$\ldots$$
$$w_n T_n = w_n \max \{0, n\,p_1 - (n-1)p_1 - p_{max}\} = 0 .$$

The sum is zero which is the minimum possible weighted tardiness value, and hence it is also optimal for the single machine case defined above.

The same argument can easily be extended to any other machine. The due date set for J_i on machine j is given by

$$c_{[i]}^m - \Sigma_{q=j+1}^{q=m} p_q = (i - 1)\, p_{max} + \Sigma_{q=1}^{q=j} p_q . \tag{4.5}$$

The proof is by induction. Suppose that the jobs complete by the set due dates at machine j (we have just shown that it is valid for machine 1). Then it can easily be seen

---

[6] A more general discussion of this problem is provided in chapter 3 above and [63].

that the jobs can be completed by the due date set for machine j + 1 if decisions were made independently for machine j + 1. It is clear that if the job is completed on machine j by the due date set by equation (4.5) in an optimal sequence, then the same optimal sequence is is generated on machine j + 1 as well (even with incorporation of the release dates for the jobs on machine j + 1). We have already shown that the due date setting scheme is optimal for machine 1 and hence the proof. □

Some points of practical importance are noteworthy in the context of the above proof. First, the latest possible optimal due dates are the artificial due dates set in the proof. Second, the optimal lead time for job i, L_i, is, assuming non-delay scheduling,

$$L_i = c^m_{[i]} - c^1_{[i-1]} \qquad (4.6)$$

$$= \Sigma^{q=m}_{q=1} p_q + (i-1)(p_{max} - p_1) .$$

This gives a unique optimal lead time for the first job, but for the other jobs the lead times determined by equation (4.6) are not necessarily the only optimal lead times when the weighted tardiness criterion is solely considered. This flexibility is due to the possible slack on each machine preceding a slower machine, and ultimately the operation due dates set according to the operation lead times derived similarly to Equation (4.6) are unique only on the slowest, i. e. the bottleneck machine (see chapter 3 for further elaboration of this point).

If the results derived in Proposition 4.2-1 are used for solving the optimal schedule in a proportionate flow shop with preemption, the solution procedure is insensitive to small errors in the optimal lead time estimation. This can be shown as follows. Let Δ_{ij} be the error in the lead time estimate of any job J_i at machine j. The estimated due date is given by:

$$(i-1)\, p_{max} + \Sigma^{q=j}_{q=1} p_q + \Delta_{ij} .$$

Note that the value of the sequence $\{J_1, J_2, \ldots\}$ on machine j is still

$$\Sigma^{i=n}_{i=1} w_i [(i-1)(\max_{q=1,\ldots,j}\{p_q\} - p_{max}) + \Delta_{ij}]^+ .$$

Since $p_{max} \geq \max_{q=1,\ldots,j}\{p_q\}$, for small values of Δ_{ij} the above expression tends to zero. Thus, in the neighborhood of the optimal lead time estimate, the solution procedure would be relatively insensitive to the errors in lead time estimation.

The minimal lateness for job J_i on machine j, M^j_i, is given by

$$M^j_i = c_{[i]}{}^j - d_i \qquad (4.7)$$

$$= (i-1) \max_{q=1,\ldots,j} p_q + \Sigma^{q=j}_{q=1} p_q - d_i .$$

From equation (4.7) it is clear that the operation due date depends on the due date, processing time on the bottleneck machine (even if the bottleneck machine precedes the machine under consideration), and processing time on the previous machines. Now we can compare the operation due dates generated by the heuristic procedures listed in Introduction to the optimal due dates derived above.

1. External (Job) Due Date: It can easily be seen that if all jobs have equal weights then this scheme gives the optimal result in a proportionate flow shop with unit jobs (Proposition VI in chapter 3)).

2. Operation Due Date: This scheme works well only if processing times are equal and jobs have identical weights.

3. Static Slack Time: This rule, as well as the two modifications of it (Static Slack per Operation and Static Slack per Remaining Processing Time) generate the same sequence as as the EDD rule in a proportionate flow shop with unit jobs.

4. The rules with Dynamic Slack calculation: A priori (standard) waiting time estimation is not likely to work optimally in static flow shop problems discussed above, since the jobs have different lead times depending on the position in the optimal schedule.

An important aspect of Proposition 4.2-1 is that it lets us to decompose the global scheduling problem into m local scheduling problems. In a practical situation, this implies that the scheduler of the factory or plant sets up internal due dates for the various operations of each job and the locally optimal decisions made at the machine centers automatically lead to a globally optimal sequence. Thus, the machine center supervisors not only have control on their resource management, but also on the scheduling within their own department as well. This possibly eliminates one major source of organizational friction between schedulers and line supervisors.

4.2.2 Setting Operation Due Dates for the Apparent Urgency Rule

In more general flow shops, it is not possible to solve analytically for the optimal due dates and the lead times of the jobs. Thus we have to use heuristic methods for the setting of operation due dates. Toward that end, we analyze a two machine flow shop. Considering the weighted tardiness criterion that is a regular measure of performance, we know from Theorem 5-1 in [17] that an optimal schedule in the two machine flow shop is a permutation schedule. Let w_k be the weight and p_{kr} the processing time on machine r of job k, ($r = 1, 2$ and $k = i, j$). The resource constrained slack of job i, if scheduled in the first position at time t, L_{i1}, can be expressed as follows (see figure 4-1 below):

$$L_{i1} = \max\{0, d_i - (t + p_{i1} + p_{i2} + W_i)\}. \tag{4.8}$$

Here W_i is the waiting time of job i on the second machine if it is started first, or:

$$W_i = \max\{0, (r - p_{i1})\},$$

where r is the processing time of the previous job on second machine exceeding time t. The correction in efficient slack of job i if scheduled second, due to possible waiting at the second machine, R_i, is given by:

$$R_i = (W_j + p_{j2} - p_{i1})^+ - W_i, \tag{4.9}$$

The corresponding definitions hold for job j.

Proposition 4.2-2: Consider two last jobs i and j in an optimal sequence in a two machine flow shop. Then in this sequence, at time t, job i precedes job j if the following condition is satisfied on the first machine:

$$(w_i/p_{i1})[1 - (L_{i1} - R_i)^+/p_{j1}]^+ > (w_j/p_{j1})[1 - (L_{j1} - R_j)^+/p_{i1}]^+ . \tag{4.10}$$

Proof: According to the observation above, we can restrict our attention to permutation schedules. Condition (4.10) has to be proven in six cases, determined by the tardiness of the jobs when scheduled at the current or the second position.

Case 1: Both jobs i and j would be tardy scheduled in either position. The cost of scheduling i first, c(i), is then (figure 4-1 below):

$$c(i) = w_i(t + p_{i1} + p_{i2} + W_i - d_i) + w_j(t + p_{i1} + p_{j1} + (W_i + p_{i2} - p_{j1})^+ + p_{j2} - d_j) .$$

Similarly, scheduling job j first has the following cost:

$$c(j) = w_j(t + p_{j1} + p_{j2} + W_j - d_j) + w_i(t + p_{j1} + p_{i1} + (W_j + p_{j2} - p_{i1})^+ + p_{i2} - d_i).$$

For $i < j$ in the optimal sequence, we must have $c(i) < c(j)$. After some manipulation of the terms, this condition yields

$$w_j(p_{i1} + (W_i + p_{i2} - p_{i1})^+ - W_j) < w_i(p_{j1} + (W_j + p_{j2} - p_{i1})^+ - W_i).$$

After dividing by $p_{j1}p_{j1}$ and rearranging, we get

$$(w_j/p_{j1})[1-(W_j-(W_i + p_{i2}-p_{j1})^+)/p_{i1}]^+ < (w_i/p_{i1})[1-(W_i-(W_j + p_{j2}-p_{i1})^+)/p_{j1}]^+ ,$$

which is the same as condition (4.10) since the slack is zero for both jobs by assumption.

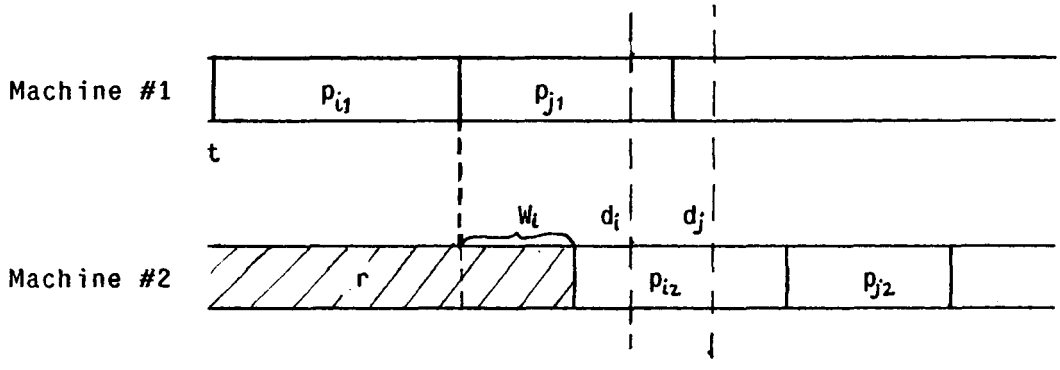


Figure 4-1: Illustration of the case in which both jobs are tardy even if started first.

Case 2: Both jobs are early in either position. Now the tardiness cost is zero for both jobs, and they can be sequenced so that condition (4.10) is satisfied.

Case 3: Both jobs would be early if started at time t, and both of them would be tardy if started second. The tardiness costs c(i) and c(j) are in this case:

$$c(i) = w_j [t + p_{i1} + p_{j1} + (W_i + p_{i2} - p_{j1})^+ + p_{i2} - d_j]^+ ,$$

$$c(j) = w_i [t + p_{j1} + p_{i1} + (W_j + p_{j2} - p_{i1})^+ + p_{j2} - d_i]^+ .$$

We can add and subtract W_i (or W_j) inside the square brackets in the expression for c(i) (for c(j)). Dividing by $p_{i1}p_{j1}$ and rearranging, we get from the condition $c(i) < c(j)$:

$$(w_j/p_{j1})[1 - (d_j - t - p_{j1} - W_j - p_{j2} + W_j - (W_i + p_{i2} - p_{j1})^+)/p_{i1}]^+$$

$$< (w_i/p_{i1})[1 - (d_i - t - p_{i1} - W_i - p_{i2} + W_i - (W_j + p_{j2} - p_{i1})^+)/p_{j1}]^+ ,$$

that is the same as equation (4.10).

Case 4: One job is early in both positions, and the other is tardy in both positions. Assume job i is tardy in both positions. Now the tardiness costs are:

$$c(i) = w_i(t + p_{i1} + p_{i2} + W_i - d_i)^+ , \text{ and}$$

$$c(j) = w_j(t + p_{j1} + p_{i1} + (W_j + p_{j2} - p_{i1})^+ + p_{i2} - d_j)^+ .$$

Obviously i < j in optimal sequence, as can be seen from the condition c(i) < c(j) :

$$0 < w_i(p_{j1} + (W_j + p_{j2} - p_{i1})^+ - W_i) ,$$

that can after dividing by $p_{i1}p_{j1}$ be rewritten as

$$(w_i/p_{i1})[1 - (W_i - (W_j + p_{j2} - p_{i1})^+)/p_{j1}]^+ > 0 ,$$

or

$$(w_i/p_{i1})[1 - (W_i - (W_j + p_{j2} - p_{i1})^+)/p_{j1}]^+ > (w_j/p_{j1})[1 - (d_j - t - p_{j1} - W_j - p_{j2} + W_j - (W_i + p_{i2} - p_{j1})^+)/p_{i1}]^+ .$$

which is the same as condition (4.10).

The proofs of two other cases, i. e. Case 5: one job is early if scheduled in the current position but tardy if second, and the other job is tardy in both positions, and Case 6: one job is early if scheduled in the current position but tardy if second, and the other job is early in both positions, are similar and have been omitted here for the sake of brevity. □

This result is valid only locally, since the completion time of the last job depends on the sequence. We suggest, however, the following myopic priority index for a two machine flow shop, reminicient of the linear single machine Apparent Urgency rule in chapter 2:

$$\Pi_i(t) = (w_i/p_{i1})[1 - (d_i - t - p_{i1} - p_{i2} - W_i - R_i)^+ / k\,\bar{p}]^+ , \tag{4.11}$$

where $\bar{p}$ is the average processing time of the schedulable jobs on machine 1 at time t. We can use the exponential form in Equation (4.2) above as well, except the operation due date for operation 1, d_{i1}, is used instead of the job due date. The operation due date is obtained by subtracting the expected lead time of the job i on machine 2 from its due date d_i, or[7]

$$d_{i1} = d_i - W_i - p_{i2} - R_i . \tag{4.12}$$

We can generalize the results in Proposition 4.2-2 and the implied operation due dates in equation (4.12) to the dynamic priority indexes in general multi-machine flow shops and general job shops. In the following, we work out an iterative procedure that uses lead times realized in the previous simulation run as lead time estimates for the next dispatching run.

4.3 A Computational Study of Lead Time Estimation

We tested the analytical results obtained above concerning the use of lead time estimates as state (load) dependent parameters in the Apparent Urgency rule, through a simulation study. This computational experiment had two major objectives: to examine the efficiency of the new AU

[7] The correction term due to possible queueing at the second machine, R_i, is usually determined by the bottleneck machines. To achieve global optimum, it is also important to avoid possible idle time of the second machine. Analysis of these issues in flow shops with bottleneck facilities has been started by Ow [59].

rule in flow shops with constant lead time estimates compared with some other commonly used dispatching rules, and to introduce an iterative procedure to improve the lead time estimates for setting the local due dates used in the AU rule. We ended up improving the waiting time estimation for CoverT as well. Before describing the study and its results, we specify the lead time adaptation procedure and motivate the selection of the iteration parameters.

4.3.1 The Scheme for Lead Time Iteration

The manufacturing (or internal) lead time estimate for job i after the operation at machine k, denoted L_{ik}, is an estimate of the time the job will spend in the process for the subsequent operations. Thus L_{ik} is given as:

$$L_{ik} = \Sigma_{q=k+1}^{m} (W_{iq} + p_{iq}) \ , \ k = 1,...,m, \qquad (4.13)$$

where W_{iq} is the estimated waiting time in the queue at machine q, p_{iq} is the corresponding processing time and m is the number of stages (machine centers) in the shop. The iterative procedure for lead time estimation consists of two stages: providing some initial values for the waiting time estimates W_{iq} and updating the estimates based on the realized waiting times in a subsequent simulation with the rule in question.

The a priori lead time estimates, or the initial values L_{ik}^{0}, can be determined from the initial waiting time estimates W_{iq}, q = 1,...,m. These estimates can be based on empirical observations, or they can be constant multiples of the processing times of the corresponding operations:

$$W_{iq}^{0} = \beta \times p_{iq} \ , \qquad (4.14)$$

where $\beta \geq 0$ is a multiplier whose efficient values depend on the load. In small shops values of $1.0 \leq \beta \leq 3.0$ work relatively well if the shop is not extremely congested. Alternatively, we can use as the initial waiting time estimates the realized waiting times when scheduling first with the WSPT rule (or with some other rule not using a priori lead time estimates).[8] The lead time estimates, L_{ik}^{n+1}, are then updated based on the realized waiting times, q_{iq}^{n}, recorded during the n^{th} simulation:[9]

$$W_{iq}^{n+1} = W_{iq}^{n} + \alpha \, (q_{iq}^{n} - W_{iq}^{n}), \qquad (4.15)$$

$$L_{ik}^{n+1} = \Sigma_{q=k+1}^{m} (W_{iq}^{n+1} + p_{iq}) \, .$$

Parameter α can be used for smoothing the effects of new waiting time changes. In the experiments in simple flow shops a wide range of values $0.3 \leq \alpha \leq 1.0$ work relatively well. In the main experiment we used the value $\alpha = 1.0$.

Now the n^{th} estimate of the operation due date for operation k of job i, d_{ik}^{n}, can be determined based on the job due date d_i and the lead time estimates as follows:

$$d_{ik}^{n} = d_i - L_{ik}^{n} \ , \ k = 1,...,m. \qquad (4.16)$$

This operation due date can then be substituted for the due date in the AU priority index in the single machine case in equation (4.2). Similarly for the iterative CoverT rule, the estimates of the

---

[8] In a dynamic job shop, the waiting time estimates can be derived analytically using Queuing Theory, see chapter 5.

[9] Here we assume that the processing times are known with certainty.

normal waiting times in equation (4.1) can be taken from the previous simulation. The weighted tardiness measures improve usually during the first few iterations, but since there is no reason to expect uniform convergence in the schedule cost or the lead time estimates we have to employ some stopping rule for the iteration. The stopping rule takes into account, besides possible convergence or fluctuation of the lead time estimates, the improvement in the weighted tardiness measure and the number of iterations.

4.3.2 The Design of the Simulation Experiment

We studied the performance of the AU rule and the iterative versions of CoverT and AU rules in a flow shop environment for different load conditions and compared them with some other well known scheduling rules. The controlled variables of the simulation study can be classified as follows: a) the layout of the shop, b) the type and parameters of the load, and c) the rules tested.[10]

1. *The layout* of the shop is flow shop with 4 or 8 machines, in which the jobs arrive simultaneously at some of the machines and then go through all the subsequent machines. All jobs may start at the first machine, or some of the machines may have an initial load (this initial load would correspond to "side load" of the machines in a dynamic flow shop). The processing speeds of the machines can have four different patterns: increasing, constant, decreasing or some random pattern. We choose to study three patterns in the case of 4-machine shop, one which displays decreasing processing speeds (relative processing times are increasing: 1.0, 1.5, 2.0 and 2.5) one with constant (all relative processing times equal to 1.0), and one with increasing processing speeds (relative processing times 2.5, 2.0, 1.5 and 1.0). The effect of lead time estimation is, a priori, more important in a shop with more queueing, i. e. in the case of increasing processing times. For the 8-machine shop there are two patterns: increasing (from 1.0 by 0.3 to 3.1) and alternating (1.5, 2.0, 1.5, 2.5, 1.0, 2.0, 1.5 and 1.0) relative processing times.

2. The major *load characteristics* are that the load is static and it consists of 20 or 60 jobs, generated randomly from the classes of loads specified below:[11]

   a. The processing times of the jobs depend on the speeds of the machines, the size distribution of the orders (jobs), and the assumption of the proportionateness of the machines. More specifically, the processing time of job i on machine j, p_{ij}, is taken from the following uniform distribution:

   $$p_{ij} \sim U\{s_i r_j (1-\rho/2),\ s_i r_j(1+\rho/2)\}\ ,\quad s_i \sim U\{5,\ 25\} \tag{4.17}$$

   For the size of the jobs or orders, s_i, we use a uniform distribution between 5 and 25. Here the machines can be proportionate to some degree, depending on the range of the random variation, ρ. In the experiment, we have almost proportionate shops (ρ = 0.5, or the maximum random variations in processing times are 25%), and shops with more random operation processing time distribution (ρ = 1.5 or maximum variations of 75%).

[10] This simulation experiment is the first for the Apparent Urgency rule, as well as for the other rules tested in static flow shops with the weighted tardiness criterion. The rules are tested in more general dynamic job shops in chapter 5.

[11] The use of a static load is justified as the first step in studying the iterative rules [40]. The same rules will be tested in dynamic job shops in chapter 5.

b. There are two types of side loads: no side load or 20% side load on each of the three trailing machines in a 4-machine shop, and respectively no side load or 10% side load in a 8-machine shop.

c. For due dates, we want to control the due date setting procedure, the tightness of the due dates, and the dispersion of the due dates. The due dates are randomly distributed over the region specified below. The random due dates (as opposed to the constant or proportional to the work contents) have been shown to make the most difficult problems to solve, see [3, 17]. The tightness of the due dates is determined by the same way as in the one machine case [61], using the bottleneck machine, or the machine with maximal expected total processing time (work contents), P, as the reference.[12] Thus the due date of job i is generated from the uniform distribution:

$$d_i \sim U\{P(1-\tau)(1-R/2),\ P(1-\tau)(1+R/2)\}, \tag{4.18}$$

Here $P(1-\tau)$ is the average due date, where the expected fraction of tardy jobs, τ, is set on two levels: $\tau = 0.3$ and $\tau = 0.6$. These levels of tardiness are the most interesting, because in more slack shops, the schedules would often have close to zero tardiness making the comparison of the rules less interesting and problems with higher levels of tardiness resemble weighted lateness problems for which WSPT is known to be very efficient. The dispersion of the due dates around the average due date is controlled by the range parameter R. We use two ranges, $R = 0.6$ and $R = 1.6$, giving max 30% and 80% random deviation from the average due date.

d. The values (or delay penalties) of the jobs are made correlated to the work contents (order size) of the jobs. The value is taken from the uniform distribution

$$w_i \sim U\{1, 2s_i\}, \tag{4.19}$$

where s_i is the order size. This assumption also hardens the problem because it makes the distribution WSPT priority indexes of the jobs flatter.

3. Finally, *the rules* to be studied in the experiment are:

a. The FCFS rule that serves as a "random" benchmark,

b. The WSPT rule, using the processing time of the imminent operation,

c. The EDD rule with global due dates,

d. The S/RPT rule, or slack per total remaining processing time.

e. The CoverT rule, which we make "weighted" by multiplying the "unweighted" priority index in Equation (4.1) by the value of the job j, w_j:

$$\pi_{CoverT,j} = (w_j/p_j)\,c_j = (w_j/p_j)[k\,W_j - (d_j - r_j - t)^+]^+/k\,W_j, \tag{4.20}$$

---

[12] For problems with high tardiness, $\tau > 0.4$, the total processing time P of the bottleneck machine was increased by (m/2)*(average processing time), where m is the number of machines in the shop, in order to compensate for possible idle time in the beginning and/or end of the run.

where the symbols are explained in section 1, equation (4.1). The CoverT rule is applied with constant waiting time estimates ($CoverT_{con}$) and as an iterative rule ($CoverT_{iter}$). The most efficient value for the look-ahead parameter in the problems studied was $k = 2.0$, and for the initial lead time estimation parameter $\beta = 2.0$.[13]

f. The AU rule using exponential look-ahead, equation (4.2). This state dependent rule is tested with initial constant lead time estimates (AU_{con}) and with iterated lead time estimates (AU_{iter}). Thus AU_{con} is the new look-ahead heuristic with constant $\kappa = 2.0$ look-ahead parameter and constant lead time estimates ($\beta = 2.0$ in Equation (4.14)), and AU_{iter}, or the same rule with lead time adaptation, updating the waiting time estimates according to Equation (4.15). The smoothing parameter α was given value 1.0, although other values ($\alpha = 0.5$ and 0.2) were tested with good results.[14]

A full factorial experiment of this design, for certain type of flow shop and given number of jobs, includes two types of side loads, two tightness classes of due dates (expected portion of tardy jobs $\tau = 0.3$ or 0.6), two kinds of dispersion of due dates (range $R = 0.6$ or 1.6) and two levels of proportionateness, or random variations in the operation processing times ($\rho = 0.5$ or 1.5), gives $2 \times 2 \times 2 \times 2 = 2^4 = 16$ cases for each of the rules. In this study, we ran samples of 10 replications of each load, altogether 160 experiments with each of the eight rules. This basic test was repeated for different shop layouts (three relative processing time distributions for 4-machine shops and two for the 8-machine shops) and for different number of jobs (20 jobs on 4 machines, and 60 jobs on 4 and 8 machines). Thus the total number of problems solved in the experiment is $(3+3+2) \times 160 = 1280$.

4.3.3 Weighted Tardiness Performance of the Rules

The primary criterion of performance in the study is weighted tardiness, ΣC_i, where the tardiness cost of job i is given in equation (4.3) as $C_i = w_i \max\{0, t_i - d_i\}$. When reporting the results of our simulations, we have normalized the tardiness measures by dividing the total weighted tardiness of a schedule by the mean weight, mean total processing time and the number of jobs. This normalized measure expresses the weighted tardiness in terms of how many average total processing times tardy an average job will be, thus permitting meaningful comparisons among problems which have different job characteristics as well as different number of machines and jobs. Because optimal solutions are not available to the problems, we use the average normalized weighted tardiness in excess over the measure of the best rule tested in each of the problems as a benchmark for the performance of the heuristics. The results of the simulation study with respect to the normalized tardiness criterion are summarized in the following tables

---

[13]The Dynamic Composite Rule in [17] was not included in our test because it is not "weighted" and thus resembles the other slack rules in weighted tardiness performance. The priority index would not have the next machine correction since the next machine in a flow shop is the same.

[14]The stopping rule for the lead time iterations for CoverT and AU was as follows: If the lead time estmates converge of fluctuate, stop. Otherwise, iterate 5 times and start counting the iterations since the last improvement in the weighted tardiness measure: iterate until there is no improvement in weighted tardiness during the last 5 iterations. This rule guarantees at least 10 iterations in the case of no convergence.

4-1, 4-2 and 4-3. We have shown a further break-down of the results according to the tightness of the due dates, determined by the expected level of tardiness $\tau = 0.3$ and $\tau = 0.6$ and further according to the range of due dates R = 0.6 and R = 1.6.

Table 4-1: Normalized weighted tardiness for different tardiness levels and ranges of due dates. Three shop layouts, 480 problems with 4 machines and 20 jobs.

Rule	Normalized Weighted Tardiness: $\tau = 0.3$ R = 0.6	$\tau = 0.3$ R = 1.6	$\tau = 0.6$ R = 0.6	$\tau = 0.6$ R = 1.6	Total	Excess over Best Rule
FCFS	.601	.753	1.455	1.588	1.129	.746
WSPT	.254	.441	.707	.883	.571	.188
EDD	.368	.218	1.251	1.240	.769	.386
S/RPT	.437	.264	1.265	1.253	.805	.422
$CoverT_{con}$	.195	.210	.706	.744	.464	.081
$CoverT_{iter}$	.162	.153	.650	.679	.411	.028
AU_{con}	.170	.166	.698	.714	.437	.053
AU_{iter}	.143	.128	.631	.658	.390	.007

Table 4-2: Normalized weighted tardiness for different tardiness levels and ranges of due dates. Three shop layouts, 480 problems with 4 machines and 60 jobs.

Rule	Normalized Weighted Tardiness: $\tau = 0.3$ R = 0.6	$\tau = 0.3$ R = 1.6	$\tau = 0.6$ R = 0.6	$\tau = 0.6$ R = 1.6	Total	Excess over Best Rule
FCFS	1.171	1.970	3.547	3.997	2.672	2.065
WSPT	.335	.850	1.565	1.919	1.167	.560
EDD	.349	.026	3.053	2.571	1.500	.893
S/RPT	.376	.039	2.948	2.472	1.459	.852
$CoverT_{con}$	.198	.043	1.513	1.069	.706	.099
$CoverT_{iter}$	.156	.027	1.380	1.016	.645	.038
AU_{con}	.132	.014	1.455	1.015	.654	.047
AU_{iter}	.112	.010	1.383	.971	.619	.012

The results show that the new Apparent Urgency rule with constant lead time estimates, AU_{con}, outperformed clearly the other simple rules in terms of weighted tardiness. The new

Table 4-3: Normalized weighted tardiness for different tardiness levels and ranges of due dates. Two shop layouts, 320 problems with 8 machines and 60 jobs.

Rule	Normalized Weighted Tardiness: $\tau = 0.3$ R = 0.6	$\tau = 0.3$ R = 1.6	$\tau = 0.6$ R = 0.6	$\tau = 0.6$ R = 1.6	Total	Excess over Best Rule
FCFS	.836	1.269	2.264	2.395	1.691	1.267
WSPT	.261	.625	1.017	1.242	.786	.362
EDD	.322	.054	1.904	1.661	.985	.561
S/RPT	.361	.075	1.921	1.658	1.004	.580
$CoverT_{con}$	.163	.071	1.002	.784	.505	.082
$CoverT_{iter}$	.131	.062	.930	.738	.465	.041
AU_{con}	.121	.033	1.004	.738	.474	.051
AU_{iter}	.104	.021	.912	.679	.431	.007

iterative AU_{iter} outperformed the weighted $CoverT_{con}$ by 10-30% in different classes of problem sizes and load characteristics. Lead time iteration improved the average weighted tardiness measures of the AU by 5-10%, and almost equally those of the CoverT rule. The new iterative $CoverT_{iter}$ was the only rule to come within 5% of the average performance of the AU_{iter} rule in some problem classes. The average number of iterations to reach the best solution was less than 5 for all different problem sizes.

The performance of AU and CoverT rules with constant setting for the look-ahead parameter ($k = 2.0$ for both) and the β parameter for obtaining the initial lead time estimate from the operation processing times ($\beta = 1.0$ for AU and $\beta = 2.0$ for CoverT) was similar across the problems with different number of jobs and machines. The shop layout in terms of the relative processing times of the machines had an obvious effect: lead time iteration was most efficient in a shop in which there were queues in front of some machines further down in the process, e. g. due to a bottleneck machine. Otherwise, the different shop layouts had no significant effects on the relative performance of the rules.

Among the load parameters, the level of tardiness τ had the following effects. For problems with low level of tardiness $\tau = 0.3$, AU rule was relatively more efficient than CoverT, and in very slack shops with almost zero average tardiness, also EDD rule was very good. High level of tardiness $\tau = 0.6$ made WSPT and CoverT relatively more efficient in problems with small dispersion of due dates, or with R = 0.6. Otherwise, the AU rule was relatively most efficient in the shops with large dispersion of due dates R = 1.6. The effects of initial side-loads or the random variations of processing times are not shown separately in the tables above. In general, initial side-loads on the machines reduced the positive effect of lead time iteration on the performance of AU and CoverT rules, AU_{con} being relatively more efficient than $CoverT_{con}$ in these problems. The magnitude of random variations in the processing times, ρ, determining the degree of proportionateness of the shop, did not affect the relative average performance of AU and CoverT

rules tested in this experiment. Of the other rules, EDD was better than $CoverT_{con}$ within the problem classes with low tardiness level $\tau = 0.3$ and large $R = 1.6$. A lower value of k-parameter can, however, improve CoverT in these cases. But otherwise, AU and CoverT were always the best rules in the study. As an example we can consider the largest problem size, 60 jobs on 8 machines. The AU rule dominated all the other rules, in the sense that AU_{con} was better than even $CoverT_{iter}$, in the average of all the problems with tardiness level of $\tau = 0.3$, of all the problems with larger dispersion of due dates, $R = 1.6$, and of all the problems with side-loads. Only for the problems having $\tau = 0.6$, $R = 0.6$ and no side-load, constant and iterative CoverT outperformed its AU counterparts. For these problems, WSPT comes close to CoverT and AU, but here a larger value for parameters κ and β used for the look-ahead and lead time estimation, respectively, would improve the performance of AU.

Some parametrized rules are sensitive to the values of the parameters making their use less convenient in practice. The parameters of the iterative AU and CoverT rules are the look-ahead parameter k, the parameter β used to determine the initial waiting time estimates from the operation processing times, and parameter α that can be used in smoothing the lead time iteration. As we have seen above, a poor choice of some of these parameters can hurt the performance of AU and CoverT in certain problem classes. In several tests not reported here, we found that AU rule is robust for the values $1.0 \leq k \leq 3.0$, in terms of average performance in all the criteria reported above. For problems with low tardiness factor, smaller values of k~1.0...2.0 tend to perform better in most of the problems and high tardiness favors higher values of k~2.0...3.0. Also the values of β can vary considerably in the range $1.0 \leq \beta \leq 3.0$ for AU_{iter} to yield relatively consistent performance. CoverT seemed to be more sensitive to the initial parameter selection. Nevertheless, the values $1.0 \leq k \leq 3.0$ and $1.0 \leq \beta \leq 3.0$ gave better average performance than any of the other myopic rules tested. Furthermore, the iterative method moderates the adverse effects of parameter selection for the rules by improving more the weighted tardiness performance of the rules with inappropriate parameter values. As to the smoothing parameter α, we used the value $\alpha = 1.0$ throughout the main experiment, i. e. the new waiting time estimates in an iteration were equal to the realized waiting times in the previous iteration. This procedure gave usually a rapid gradual improvement in the weighted tardiness measure, although the convergence of the lead time estimates was not uniform, causing the weighted tardiness to diverge or fluctuate during the later iterations. From a few tests we concluded that the parameter α is not very critical, especially in large shops. Values of $0.2 \leq \alpha \leq 1.0$ performed almost equally in the problems with 8 machines and 60 jobs. The number of iterations is usually larger (but under 8 in the average) for smaller values of α. Another way to avoid fast divergence is to limit the range of feasible lead time estimates.[15]

4.3.4 Other Criteria of Performance

The new AU rule appeared to be the best myopic dispatching rule for weighted tardiness scheduling among the rules tested above. However, the lead time iteration procedure can further improve the performance of the AU and CoverT rules. But are these results with respect to

---

[15] The tests with the range of $0.3p_{ij} \leq W_{ij} \leq 5p_{ij}$, where p_{ij} and W_{ij} are the processing time and waiting time estimate, respectively, of job i on machine j, gave positive results in larger problems.

weighted tardiness achieved at the expense of some other important scheduling criteria? We recorded also four secondary criteria in order to verify the overall performance of the rules tested: a) the portion of tardy jobs, b) the maximal (normalized) tardiness of a single job in the schedule, c) the normalized flow time from the start to the completion, weighted by job sizes s_i, (work-in-process, or WIP), and d) the normalized weighted shop-time, i. e., the time from the starting of the first operation to the completion or to the due date, whichever is later, weighted by job sizes s_i (work-in-shop, or WIS). The results for different problem classes are shown in tables 4-4, 4-5 and 4-6.

Table 4-4: Rule performance in terms of portion of tardy jobs, maximal weighted tardiness, work in process (WIP) and work in shop (WIS) inventories for 4 machines and 20 jobs (three shop layouts, 480 problems).

Rule	Norm.Weight. Tardiness	Max.Norm. Tardiness	Portion of Tardy Jobs	WIP	WIS
FCFS	1.129	.326	53.6%	2.24	3.11
WSPT	.571	.145	50.3%	1.91	2.84
EDD	.769	.176	56.5%	1.94	2.48
S/RPT	.805	.193	67.1%	2.00	2.51
$CoverT_{con}$	.464	.115	52.3%	2.04	2.70
$CoverT_{iter}$	.416	.101	52.0%	1.97	2.60
AU_{con}	.437	.108	48.5%	1.98	2.62
AU_{iter}	.390	.100	45.5%	1.93	2.57

Table 4-5: Rule performance in terms of portion of tardy jobs, maximal weighted tardiness, work in process (WIP) and work in shop (WIS) inventories for 4 machines and 60 jobs (three shop layouts, 480 problems).

Rule	Norm.Weight. Tardiness	Max.Norm. Tardiness	Portion of Tardy Jobs	WIP	WIS
FCFS	2.672	.367	47.8%	5.09	8.12
WSPT	1.167	.134	44.7%	3.72	6.99
EDD	1.500	.151	43.2%	3.69	5.58
S/RPT	1.459	.183	51.4%	3.80	5.52
$CoverT_{con}$	.706	.083	38.0%	4.45	6.53
$CoverT_{iter}$	.645	.075	37.1%	4.10	6.18
AU_{con}	.654	.071	34.6%	3.72	5.68
AU_{iter}	.619	.070	33.3%	3.79	5.80

From the results we can see that AU and CoverT rules performed well with respect to all of the secondary criteria. Furthermore, the lead time iteration did not have any adverse effects on their performance. In fact, it often improved these measures as well. In particular, AU was the best rule in terms of the average number of tardy jobs and also gained most from iteration. This

Table 4-6: Rule performance in terms of portion of tardy jobs, maximal weighted tardiness, work in process (WIP) and work in shop (WIS) inventories for 8 machines and 60 jobs (two shop layouts, 320 problems).

Rule	Norm.Weight. Tardiness	Max.Norm. Tardiness	Portion of Tardy Jobs	WIP	WIS
FCFS	1.691	.227	49.5%	4.52	6.33
WSPT	.786	.085	46.2%	3.80	5.69
EDD	.985	.097	47.5%	3.84	4.94
S/RPT	1.004	.123	57.1%	3.96	4.99
$CoverT_{con}$	.505	.055	42.1%	4.24	5.49
$CoverT_{iter}$	.465	.052	41.6%	4.05	5.34
AU_{con}	.474	.049	39.9%	3.87	5.03
AU_{iter}	.431	.048	37.4%	3.86	5.07

indicates that AU succeeds in making more accurate trade-offs between the urgency and the processing time than does CoverT.

4.4 Conclusions

We have discussed the use of lead time estimates as state dependent parameters in myopic dispatching rules for flow shop scheduling. For proportionate flow shops with unit jobs it is shown that appropriate lead time estimates can be applied in local heuristics to achieve a globally optimal weighted tardiness schedule. Thus the lead times estimates improve the coordination of distributed scheduling systems. In general flow shops, efficient lead time estimates can be generated through iterative procedures introduced in this chapter. The Apparent Urgency rule, developed originally for one stage problems [61, 62], as well as a "weighted " version of the CoverT rule [15, 16], are both amenable to the coordination. In a large computational experiment with static flow shop problems, the new AU rule outperformed CoverT and the other four rules tested in weighted tardiness. The AU and CoverT rules were robust even with constant parameter values, and the new iterative lead time estimation method further moderates the effects of inappropriate selection of the parameter values. The lead time iteration maintained the good average performance of the AU and CoverT rules also with respect to important secondary criteria such as the number of tardy jobs, maximum weighted tardiness, and work-in-process inventory.

The results are encouraging for the further application of the state dependent scheduling rules and the lead time iteration in more general job shop environments. We can also ask to what extent lead time iteration and some exogenous setting of the lead time estimates could be used to control the completion times of the jobs in practical scheduling situations. If the consistent multi-objective performance can be transferred from the simple test-bed to real job shops, the look-ahead rules could be used as a kernel of an interactive scheduling system.

5. Slack Evaluation for Priority Dispatching in Dynamic Job Shops

Summary

In this chapter, we extend the use of indirect load information and performance feedback for lead time estimation to dynamic job shops. The different roles of slack evaluation in the Apparent Urgency and modified CoverT rules are discussed. Appropriate "priority-based" waiting time estimates are derived for more efficient slack evaluation. Further improvement can be achieved through the lead time estimation procedure introduced in the previous chapter. In a dynamic setting this iterative procedure requires the implementation of rolling forecasting and planning horizons.

Despite the practical importance of scheduling with due dates, there are no previous studies in general job shops using the weighted-tardiness criterion. Our results in a large scale experiment are quite similar with those obtained in static flow shops. The new Apparent Urgency rule and our modified CoverT rule are superior to all the other rules tested. Moreover, the iterative Apparent Urgency rule has 5-30% lower weighted tardiness costs than the basic CoverT rule in widely varied load situations. This margin, and the portion of tardy jobs, can be reduced by using the new priority-based waiting time estimates or lead time iteration with CoverT. AU and CoverT also outperform the other rules tested when compared in terms of combined inventory-holding, rush-shipping and tardiness costs. The best rule in all load conditions is the AU rule with lead time iteration.

5.1 Introduction

Some dynamic dispatching rules which use a job's due date require also some estimates of the waiting times on its subsequent operations to evaluate the effect of any slack upon the job's priority index value. Since the priority of the job, and hence its expected waiting time at any machine, depends on the waiting time estimates used, an efficient waiting time estimation method should use the expected priority of the job to guarantee the consistency of the priority assignment. In this chapter, we develop efficient waiting time estimation methods for the Apparent Urgency rule [61, 62] and the CoverT rule [3, 16] in general job shop scheduling problems with dynamic job arrivals. The original CoverT uses a constant multiple of the processing time of the job on machine as a "standard" waiting time estimate which is then represents a worst case limit in assessing the probability of the job being tardy. This is appropriate in average (non-weighted) tardiness problems since the priority of a job is proportional to the inverse of its processing time. The same method is also used in many Materials Requirements Planning (MRP) systems. In the weighted tardiness problems, jobs have different tardiness penalties, or weights, and a job's priority is proportional to its "natural index", penalty over its processing time. Obviously the expected waiting time should, in this case, be estimated on the basis of the inverse of this natural index. This simple idea is made precise in section 2 by using waiting line analysis of the weighted lateness case. The AU rule evaluates a job's actual resource constrained slack. Hence, ideally, AU should use the expected waiting times and not the standard (worst case) estimates for setting its operation due dates. Any initial waiting

time estimates can be adjusted using the iterative procedures introduced in chapter 4. In a dynamic shop the application of the lead time iteration procedure requires rolling forecasting and planning horizons. We discuss the setting of these horizons so as to improve the consistency and convergence of the iteration in section 2.

In section 3, we specify a large experiment to test the effectiveness of the different waiting time estimation methods. The results reported in section 4 show that AU and CoverT are again far superior to the other rules tested. Furthermore, the new priority-based waiting time estimates consistently improve the weighted tardiness of the Apparent Urgency rule, up to 5% beyond traditional estimation methods. But more accurate waiting time estimates, obtained through lead time iteration, can still improve the performance by 3-10%. The fraction of tardy jobs is smaller in as well in most cases. Both the basic and iterated versions of AU perform better than those of CoverT, but the difference diminishes with higher load. In fact the new priority-based waiting time estimates improve CoverT's performance in weighted tardiness beyond that of lead time iteration. The portion of tardy jobs, previously CoverT's weak point, improves up to the level of the AU rule. The rules are also compared in terms of combined costs of inventory-holding, rush-shipping of tardy jobs, and variable tardiness penalties. The AU rule with lead time iteration is the best by a narrow, but consistent, margin over all the problems studied. The conclusions of this chapter are in section 5.

5.2 Estimation of Job Waiting Times to Evaluate Slack Priorities

5.2.1 Slack Evaluation in the Apparent Urgency and CoverT Rules

The slack of the job, or the time between the due date and present date which is not allocated explicitly for processing or waiting, is used as the dynamic term in the priority index function of many dynamic dispatching rules. Jackson (see e. g. [17]) used a due date based "urgency index" to study the statistical properties of the dynamic scheduling rules. The conventional slack-time rules give the highest priority to the job with least slack, often divided by the number of remaining operations or by the remaining processing time.[1] A common priority rule in MRP systems is the "Critical Ratio", or the ratio of the global slack to the remaining processing time. Carroll [16] elaborated the use of slack in his CoverT rules by mapping the slack to the expected tardiness "cost" of job i, c_{ij}, that varies from zero, for a very slack job, to a maximum of one, for a tardy job. The cost c_{ij} represents the probability of a job being tardy by its completion. The relative urgency of a job depends on the ratio of this probability over the processing time of the job on machine j, or c_{ij}/p_{ij}. In the case of job specific tardiness penalties, w_i, a job that is within the remaininig processing time of its due date (i.e. it will be tardy), can be assigned the "natural priority" or the WSPT priority index $\pi^{ij}_{WSPT} = w_i/p_{ij}$. This priority can be discounted linearly with the increasing slack until the "worst case" waiting time on the subsequent machines, or the length of the look-ahead, is reached. The jobs with a longer slack are assigned the priority

[1] The conventional rules compute the slack without the waiting times forced by resource constraints. The AU rule recognizes the resource constraints explicitly, see discussion below.

index value of zero as in the original CoverT, see Figure 5-1. Hence the "weighted" CoverT priority index represents the expected tardiness cost per a unit of machine processing time [15, 16]:[2]

$$\pi^{ij}_{CoverT}(t) = (w_i/p_{ij})[1 - (d_i - t - r_{ij})^+ / k\, W_i^{'j}]^+ \qquad (5.1)$$

where d_i is the due date of job i, r_{ij} is the remaining processing time of the job including the imminent operation j, and $W_i^{'j} = \Sigma_{k=j}^{n} W'_{ik}$ is the remaining standard waiting time or the maximal waiting time for the remaining operations in a normal load situation. Hence the total slack of job i, $d_i - t - r_{ij}$, is compared to a standard, $W_i^{'j}$, that is estimated either from empirical data or using a heuristic rule such as a constant multiple of the "crash time", or the remaining processing time [15]. The resulting look-ahead profile is shown in figure 5-1. Carroll [16] used the parameter **k** to shorten the allowed slack in a light load, setting **k** <1.0. Thus the length of the look-ahead period was determined by one parameter and the length of total standard waiting time.[3] However, the use of iterative lead time estimation introduced in [66], or in chapter 4 above, requires a separation of the look-ahead and waiting time estimation parameters. A new look-ahead parameter $\mathbf{k_c}$ determines the slack tolerance beyond the anticipated waiting time generated by lead time estimation; hence $\mathbf{k_c} > 1.0$. Another parameter β determines the initial estimate for the waiting time, for example, the multiple of the crash time.

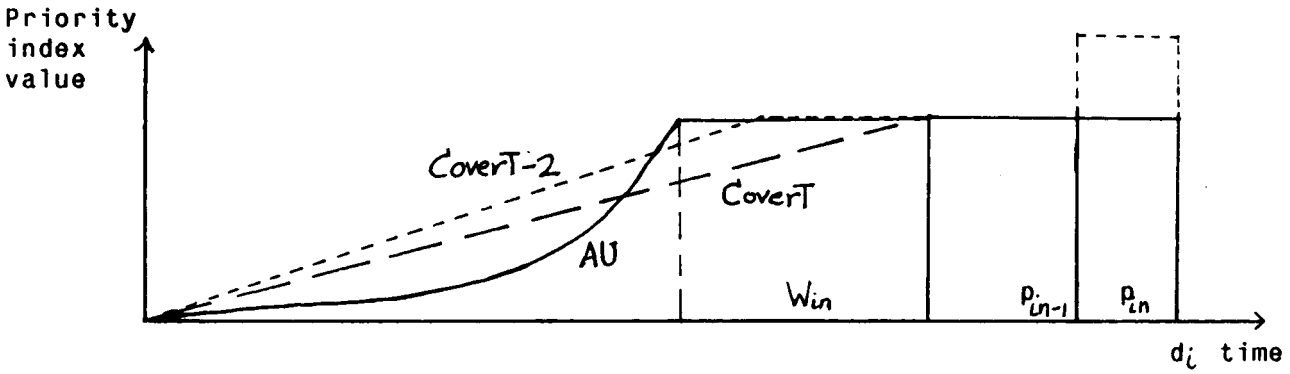


Figure 5-1: The comparison of the CoverT, CoverT-2 and AU priority indexes for a job which has two operations remaining.

In the earlier study in chapter 4 above, CoverT left 5-25% more jobs tardy than AU. We modified the look-ahead of CoverT in equation (5.1) by keeping the priority index at its maximal level over a portion σ, ($0<\sigma<1.0$), of the total look ahead period. The new priority index is $\pi^{ij}_{CoverT\text{-}2}(t) = (w_i/p_{ij})[1 - (d_i - t - r_{ij} - \sigma W_i^{'j})^+ / k\, W_i^{'j}]^+$. For this safety limit the value $\sigma = 0.2$ worked best, producing often a minor improvement of the portion of tardy jobs. However, this gain was offset by a deterioration of the weighted tardiness performance in many cases.

---

[2] The notation $(x)^+$ means that only the positive values of x are considered, i. e. $(x)^+ = \text{Max}\{0, x\}$.

[3] In the original CoverT studies, $\beta = 6.0$ and $k_c = 1.0$, for a load 90% of capacity, and $k_c = 0.5$ for a load 80% of capacity, see [16].

The slack evaluation of Apparent Urgency rule is decomposed into two parts: the estimation of machine constrained waiting times, or the global lead time down the route, and the determination of the length of the local look-ahead.[4] The look-ahead depicts the decreasing marginal tardiness cost of the job when its slack is normalized by the average processing time of the competing jobs. The time reference determining the slack is the (local) operation due date, that is the (external) job due date minus the estimated lead time on the subsequent machines. The form of the look-ahead is exponential discounting as shown in figure 5-1. Hence the priority index formula for AU is [61, 62]:

$$\pi^{ij}_{AU}(t) = (w_i/p_{ij})\, e^{-[d_i - t - p_{ij} - W_i^{j+1} - r_{ij+1}]^+ / \kappa \bar{p}} , \qquad (5.2)$$

where $W_i^{j+1} = \Sigma_{k=j+1}^{n} W_{ik}$ is the expected waiting time of the job on the subsequent machines. κ is the parameter that adjusts the length of the look-ahead scaled by the average processing time of the schedulable jobs, $\bar{p}$. Here the lead time estimate, or the sum of the processing times and the expected waiting times on the subsequent machines, is used to determine a realistic operation due date that takes into account job's waiting times forced by the contention over machines:[5] $d_{ij} = d_i - (W_i^{j+1} + r_{ij+1})$. Hence a resource constrained slack of the job, $d_{ij} - t - p_{ij}$, is used to determine its priority. In the following, we will discuss the estimation of the expected waiting times, W_{ij}, using two different methods: a waiting line analysis and an extension of the lead time iteration procedure introduced in chapter 4 above.

5.2.2 Derivation of Priority-Based Waiting Times

A conventional waiting time estimate for a job at a machine, used for AU and CoverT in the previous studies, is a multiple of the processing time of the corresponding operation [15, 16, 17]. In the following, we analyze the expected waiting times of the jobs in a job shop when scheduling with the weighted tardiness criterion. Assume a decomposition of the shop into individual machines. Each machine has an input process which is assumed to be stationary. The expected waiting times of the jobs are determined from the following waiting line analysis that captures jobs' different priority levels. To establish the required notation, let:

λ = future arrival rate of jobs,

$W(\pi_i)$ = the expected waiting time of job i with WSPT priority index π_i, arriving at the machine,

$Q(\pi > \pi_i)$ = the expected number of jobs in the queue with priority index value larger than or equal to π_i,

$P(\pi > \pi_i)$ = probability that a job arriving in the future will have a higher priority index value than job i,

---

[4] Some details of the look-ahead feature and the lead time iteration for the AU rule have already been discussed in chapters 2 and 4 above.

[5] When the expected waiting time estimates are used in CoverT's priority index, the following should hold: $kW_i^j = k_c W_i^j$.

U = the expected remaining processing time of the job currently being processed,

$\bar{p}_{\pi i} = E\{p_j | \pi_j > \pi_i\}$ = the expected processing time of jobs j for which $\pi_j > \pi_i$.

Notice that the dynamic priority index values will be equally proportional to the WSPT priorities π_i if all jobs i have approximately equal slack times. The arrival rate and the job parameters in a machine queue can be assumed to have the same distribution as the process arriving to the shop.[6] Then we can assume, following Dolan [19], that the expected waiting time of a job arriving at a machine is equal to the processing times of the jobs already in the queue having higher priority index values, or being processed, plus the sum of the processing times of the jobs arriving during the wait that have higher priorities:

$$W(\pi_i) = Q(\pi > \pi_i)\, \bar{p}_{\pi i} + U + \lambda\, W(\pi_i)\, P(\pi > \pi_i)\, \bar{p}_{\pi i}, \qquad (5.3)$$

if we can assume that the arrival rate λ is less than the service rate $1/\bar{p}$. Solving equation (5.3) for $W(\pi_i)$, we get:

$$W(\pi_i) = [Q(\pi > \pi_i)\, \bar{p}_{\pi i} + U] \,/\, [1.0 - \lambda\, P(\pi > \pi_i)\, \bar{p}_{\pi i}]. \qquad (5.4)$$

The expected waiting time is directly proportional to the relevant queue length and increases with the arrival rate of the higher-priority jobs. This result assumes constant slack; hence it holds more accurately in a shop in which the load is high and the due date allowance is short. The waiting time estimates indicated by the analysis above can be approximated numerically, given the distributions of the processing times, p_{ij}, and the delay penalties, w_i. One approximation is shown in figure 5-2 for the case of uniform distributions of p_{ij} and $w_i \sim U(0.1, 1.9)$, with the mean values of $\bar{p} = \bar{w} = 1.0$. Although the priority rules coordinate the jobs so as to avoid too early starts, some jobs in the queues will have rather long slack since the due date setting is random. The effects of increased load and slack on the expected waiting times of different jobs are:

- the length of expected waiting time increases with increasing load, since the arrival rate λ increases and the expected queue length $Q(\pi > \pi_i)$ increases. This changes mainly the waiting time estimates for jobs which have $\pi \leq \bar{\pi}$, see Figure 5-2.

- longer due date allowances decrease the number of critical jobs in the queues, thus shortening the waiting time of the jobs which have relatively low WSPT priority. On the other hand, jobs which have high WSPT priority might stay in the queue longer than indicated above due to slack and lower relative urgency. As a result, the waiting time distribution will be flatter with longer due date allowances.

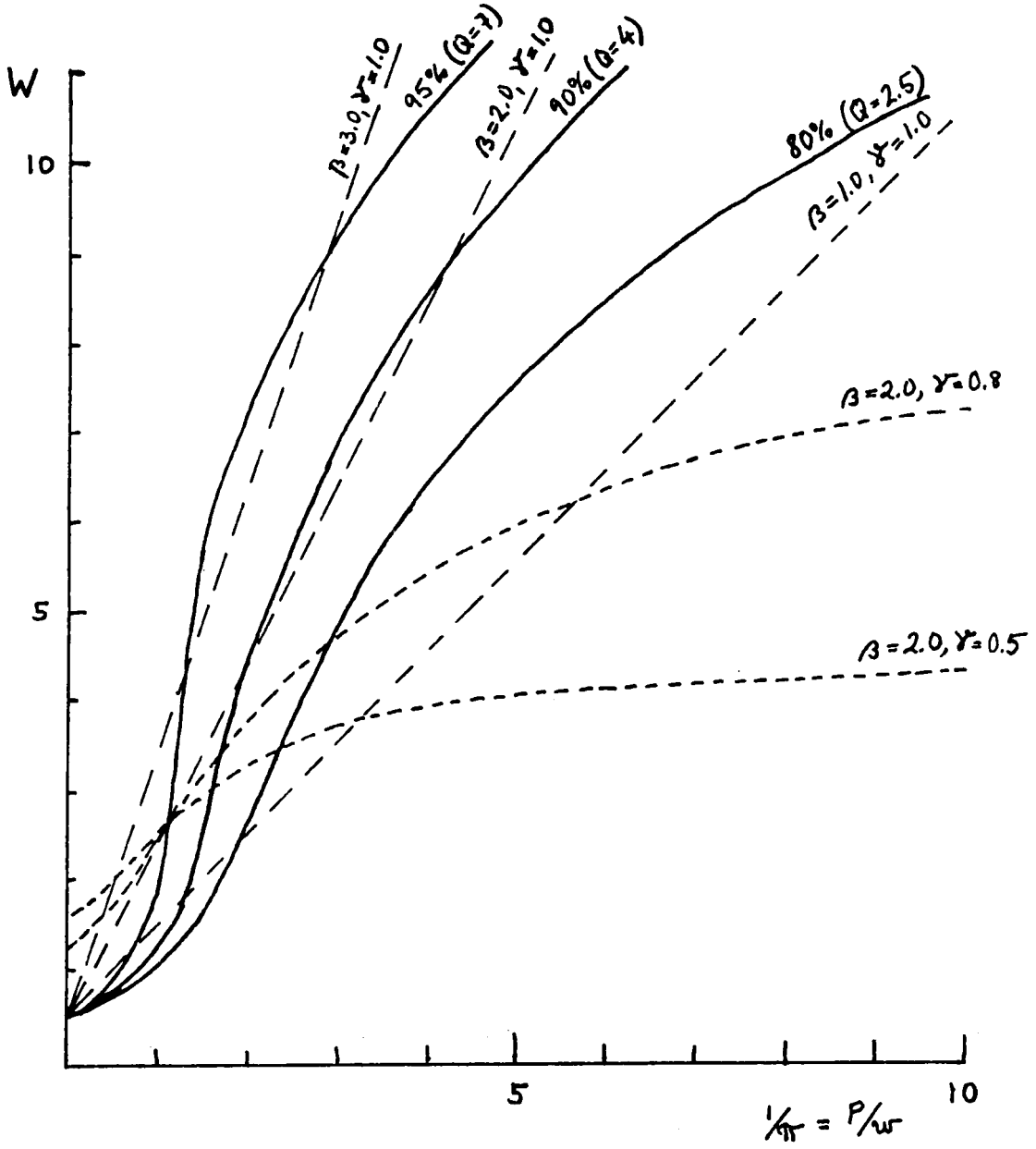
5.2.3 Waiting Time Estimation in a Dynamic Job Shop

The graphs in figure 5-2 above suggest the estimation of the expected waiting times with some function of the job parameters. But can we use the same waiting time estimates for both CoverT and AU? CoverT uses the maximum standard stay in the queues as a reference for the relative tardiness of a job. Apparent Urgency uses realistic operation due dates in the priority

---

[6] Obviously this is not always true since WSPT rule can leave some long jobs sitting in the queues. Conway et. al. [17] provide simulation results that qualify this approximation over a distinct busy period.

Figure 5-2: Three sets of a job's expected waiting time graphs as functions of the priority index π : 1) The approximation of the expected waiting time [7] (·····), 2) The priority-based estimation function $W = \beta\bar{w}/\pi(\gamma) + 0.5$, $\gamma = 1.0$, (– – –). 3) The priority based waiting time estimates for slack jobs, $\gamma < 1.0$, (·····).



[7] Job attributes p and w have are taken from uniform distributions p~U(0.1, 1.9) and w~U(0.1, 1.9) for these waiting time functions.

index valuation. Hence the waiting time estimates should be close to the expected waiting times forced by the sharing of resources. To recapitulate, *the coordination of the dispatching decisions via lead time estimates requires unbiased estimates of the expected waiting times for the AU rule but the worst case estimates for the CoverT rule.* The expected average and worst case waiting times are related to each other in a nontrivial manner. For the sake of simplicity, we approximate the normative standard lead times by multiplying the expected waiting times by some tolerance factor to count for the worst case. This tolerance is introduced by parameter k_c, $1.0<k_c$, discussed above:

(maximum normal waiting time) = k_c*(expected waiting time).

Hence we propose the same waiting time estimation methods for AU and CoverT rules to be tested against the traditional multiples of the processing times:

1. *Static lead time estimation* : Expected waiting times that depend on the attributes of the job and some aggregate measure of load such as the average utilization or the average queue length. We introduce two slightly different waiting time estimation functions to study alternative coordination principles:

   - The expected waiting time given in equation (5.4), approximated by a inverse function of the WSPT priority index value, and

   - A composite function in w_i and p_{ij}, that allows us to experiment with different weighting of these attributes.

2. *Dynamic lead time iteration* : Iterative adjustment of the waiting time estimates according to the performance feedback using dynamic adjustment of the estimates with rolling forecasting and planning horizons.

We use the following parameters for adjusting the waiting time estimation functions for operation j of job i ($\bar{p}$ is the average processing time, $\bar{\pi}$ the average priority index value, and $\bar{w}$ is the average weight of the jobs in the shop) :

α = smoothing parameter for the processing time effect, $\tilde{p}_{ij}(\alpha) = \alpha\, p_{ij} + (1 - \alpha)\,\bar{p}$.

β = scale parameter for the load effect, i. e. the waiting time of an "average" job is proportional to some measure of load.

γ = smoothing parameter for the priority index (or the weight) of the job: $\tilde{\pi}_{ij}(\gamma) = \gamma\pi_{ij} + (1 - \gamma)\bar{\pi}$, or $\tilde{w}_i(\gamma) = \gamma w_i + (1 - \gamma)\bar{w}$.

The functional form of *the priority based waiting time estimation* is an inverse of the priority index value π_{ij} plus the expected processing time of the job currently under processing:[8]

$$W_{ij} = \beta\, \bar{w}/\tilde{\pi}_{ij}(\gamma) + 0.5\,\bar{p}\ . \qquad (5.5)$$

The estimates can be bounded from above, for example by $\beta\bar{n}\,\bar{p}$, to avoid excessive waiting time allowances. Here $\bar{n}$ is the average number of operations of the jobs.

[8] Another modification of the priority based waiting time estimation function was tested: $W_{ij} = \beta\, \tilde{p}(\alpha)/\tilde{\pi}_{ij}(\gamma) + 0.5\,\bar{p}$ with good results.

In *the composite waiting time estimation function*, both the weight and processing times can be smoothed:

$$W_{ij} = \beta \bar{w} \tilde{p}_{ij}(\alpha) / \tilde{w}_j(\gamma) . \quad (5.6)$$

This estimate can be bounded from below and from above. The composite form allows the experimentation with different smoothing of w_i and p_{ij} separately. For example, we get the conventional multiple of crash time waiting time estimates when $\alpha = 1.0$ and $\gamma = 0.0$. By setting $\alpha = 0.0$ and $\gamma = 1.0$ waiting times can be estimated solely on the basis of the value of the job.

Lead time iteration uses the feedback from the global performance of the rule to adjust the current lead time estimates based on the realized waiting times. This procedure can be adopted in the dynamic case through a rolling horizon procedure. The rule is run from a starting date, t_0, to a forecasting horizon, t_f, and the performance in terms of weighted tardiness is recorded. Let W_{ik} be the estimated waiting time in the queue at machine k and p_{ik} the corresponding processing time. The iterative procedure for waiting time estimation consists of two stages. First some initial values are provided for the waiting time estimates; then the estimates are updated based on the realized waiting times in a simulation with the rule in question. The *a priori* waiting time estimates, W^0_{ik}, $k = 1,...,m$, can be based on the estimation methods introduced above. For example, they can be multiples of the corresponding processing times:

$$W^0_{ik} = \beta \times p_{ik} , \quad (5.7)$$

Letting $\beta = 2.0$ provides relatively robust estimates. The updated lead time estimates W^{n+1}_{ik} are then be obtained using the realized waiting times q^n_{ik} recorded during the n^{th} simulation as follows:

$$W^{n+1}_{ik} = W^n_{ik} + \alpha (q^n_{ik} - W^n_{ik}), \quad (5.8)$$

The parameter α, $0.0 < \alpha \leq 1.0$, can be used for smoothing the waiting time changes. In the main experiment $\alpha = 0.5$.

The operation due date for operation k of job i, d_{ik}, can now be determined based on the job due date d_i and the lead time estimates as follows:

$$d_{ik} = d_i - \Sigma^m_{q=k+1} (W_{iq} + p_{iq}) , \quad k = 1,...,m,. \quad (5.9)$$

This operation due date can then be used in equation (5.2) for the next iteration with the Apparent Urgency rule. For the iterative CoverT rule, the estimates of the standard waiting times in equation (5.1) can be derived, as discussed above, after smoothing the realized waiting times in the previous simulation. Since the convergence in schedule cost or lead time estimates is not uniform in general, we employ a stopping rule of a maximum of 10 iterations. Whenever the weighted tardiness measure improves, we record the waiting time estimates. After 10 iterations the best waiting time estimates thus far are restored, and the system is run to a planning horizon $t_p < t_f$. The performance of this iterative method was found to be rather insensitive to the choice of the lengths of the forecasting and planning horizons, subject to the following considerations:

1. Improvements and convergence are often easier to obtain with a relatively short period, such that $t_f - t_0 < 200 * \bar{p}$. Otherwise, local improvements in performance measures might be offset by a worsening somewhere else.

2. The consistency of the decisions presupposes a relatively long forecasting horizon, that is long enough so that lengthy jobs can be completed during the period.

Thus $t_f - t_0 > 20\,\bar{p}$, say.

3. The planning horizon should be sufficiently shorter than the forecasting horizon to prevent overly opportunistic myopic decisions at the end of the period.

4. Computational efficiency requires that the planning horizon is not exremely short compared to the forecasting horizon.

Robust performance was found with the following values:

$$30*\bar{p} < (t_f - t_0) < 150*\bar{p}, \quad (t_p - t_0) = (0.5...1.0)*(t_f - t_0).$$

The actual values used in the main experiment were $(t_f - t_0) = 133*\bar{p}$ and $(t_p - t_0) = 0.5*(t_f - t_0)$. The initial values for the waiting time iteration were $W_{ij} = 2.0*p_{ij}$. The smoothing parameter in equation (5.8), $\alpha = 0.5$, and the waiting time estimates were bounded from below and from above by $0.5*\bar{p} < W_{ij} < 10*\bar{p}$.

5.3 Experimental Design for Dynamic Job Shops

5.3.1 The Job Shop Problems

The dynamic job shops studied had 10 machines. Jobs arrive continuously according to a truncated Poisson process and are simulated over long time periods. We tested the weighted tardiness performance of the basic and iterative CoverT and Apparent Urgency rules in large job shop problems having 2,000 jobs, each of which had 1 to 10 operations assigned randomly. All 2000 jobs were included in the recording of the results of any simulation run, since we are interested in the performance of the rules in varying shop conditions.[9] The processing times were generated either from a uniform distribution $p \sim U(1,30)$ for a general load, or with proportionality in the following way: the job was assigned a size from a uniform distribution $s \sim U(5,25)$, and the operation processing times were generated from the uniform distribution $p \sim U(.33*s,\ 1.67*s)$. In addition to these general and proportionate loads, a third kind of load was generated with uneven distribution of the processing speeds of the machines. The relative processing times for the ten machines were assigned as follows: (0.7, 0.8, 0.9, 1.0, 1.0, 1.0, 1.0, 1.1, 1.2, 1.2.) Thus there were three machines which had less-than-average load (as much as 30% below the average) and there were three botleneck machines with up to 20% more work than average. The weight of a job was generated from a uniform distribution $w_i \sim U(1, 2*s_i)$, where $s_i = 15$ for the general load.

The average level of load was determined by varying the arrival rate to yield five different load conditions: 80%, 85%, 90%, 95% and 97% of the bottleneck capacity. Two different due date settings were used: in a slack shop, the due date allowance varied uniformly from 0 to 12 times the total processing time of an average job, $5.5*\bar{p}$. Thus the jobs had an average due date allowance of six times the average total processing time.[10] The other shops had a tighter average

---

[9] Most previous studies have recorded only the performance of the rule over a more stationary time period, by loading the shop without recording the performance of the rules. and also stopping the recording before the shop is empty [1, 17].

[10] Conway et.al. [17] found that the random due date setting was the most difficult for most rules to handle. See also [3].

due date allowance, set to three times the average processing time. This is less than in previous studies which used at least four times the average processing time as the average due date allowance.

5.3.2 Measures of Performance

The weighted tardiness criterion associates the cost, $C_i(t_i) = w_i \max\{0, (t_i - d_i)\}$, with the completion of job i at time t_i, when the due date of the job is d_i and the penalty for tardiness of one time unit is w_i. The major objective is to minimize the total tardiness cost, $\Sigma_i C_i(t_i)$. We make it easier to compare the weighted tardiness performance of the rules in different job shops, by normalizing the weighted tardiness cost by the average processing time, the average weight and the total number of jobs completed as was done in chapter 4 above, or in [66]. The number of tardy jobs is an important secondary criterion that is normalized in the reporting to the portion of tardy jobs of the total number of jobs completed. The measure of maximum weighted tardiness was found to be correlated with the average weighted tardiness performance of the rules in a previous study. It will not be reported here.

A good dispatching rule should also provide resonable inventory holding cost performance. We used the flow time from the start to the completion, weighted by the size of the job as the primary measure of the work-in-process inventory (WIP), normalized by the average length, average size and total number of the jobs completed. Because early completion sometimes forces the firm to hold the product until the requested delivery date, we recorded a secondary inventory measure called work-in-shop (WIS), in chapter 4. The normalized WIS is computed like the WIP statistic except that the flow time is replaced by the time from the start date until the due date or the completion of the job, whichever is later. A large difference between WIP and WIS in a problem with a relatively high capacity utilization indicates that the rule is not successful in coordinating the completion dates and due dates.

5.3.3 Rules Tested

The main interest of the experiment was to find out if the new lead time estimation methods improve the performance of the AU and CoverT rules over the basic versions. However, since no systematic comparison of any rules in the weighted tardiness problems has been reported before, we wanted to test some common rules for benchmarks. These were the same as in chapter 4:

1. The FCFS rule that served as a "random" benchmark,

2. The WSPT rule, using the processing time of the imminent operation,

3. The EDD rule with global due dates,

4. The S/RPT rule, or slack per total remaining processing time.

We made the CoverT rule "weighted" by multiplying the original "unweighted" priority index by the delay penalty job j, w_j, see equation (5.1) above. Four different versions of CoverT will be reported. One used constant multiples of crash-times as waiting time estimates ($CoverT_{con}$) and another used lead-time iteration ($CoverT_{iter}$). The values for the look-ahead parameter k_c and for

Table 5-1: The assignment of the look-ahead parameter κ and the lead time estimation parameter β for the AU and CoverT rules corresponding to the load in the shop in terms of the expected capacity utilization.

		Lead Time Estimation Method:			
		Iterative (initial)		Priority Based	
	Capacity Util.	κ	β	κ	β
CoverT:	< 90%	1.5	2.0	2.0	1.5
	90% ... 95%	1.5	2.0	2.0	2.0
	95% <	2.0	2.0	2.0	3.0
Apparent Urgency:	< 85%	2.0	2.0	2.0	1.0
	85% ... 90%	3.0	2.0	3.0	1.0
	90% <	4.0	2.0	4.0	2.0

the initial lead time estimation parameter β are given in Table 5-1 for different levels of anticipated average load of the shop. The priority-based waiting time estimation method ($CoverT_{prio}$) was applied with the parameter value $\gamma = 1.0$ in equation (5.5). The composite waiting time estimation method ($CoverT_{comp}$), was used with the parameters $\alpha = \gamma = 0.5$.

The Apparent Urgency rule in equation (5.2) above, was tested with initial constant lead time estimates (AU_{con}) and with iterated lead time estimates (AU_{iter}). The values of constant look-ahead parameter κ and lead time estimation parameter β in equation (5.7) are given in table 5-1. AU_{iter} is the same rule with lead time adaptation, updating the waiting time estimates according to equation (5.8). The same priority-based, AU_{prio}, and composite, AU_{comp}, waiting time estimation functions were used as for CoverT.[11]

5.4 The Results of the Experiment

5.4.1 Weighted Tardiness and Portion of Tardy Jobs

The results of the experiment are reported according to the expected load and the average due date allowance. Since there was no significant difference in the performance of the rules and waiting time estimation methods for the different job shop types tested (general, proportionate and bottleneck job shops), the results for the three have been averaged to obtain larger samples.

The normalized weighted tardiness for the different rules are shown in Table 5-2 as a function of the expected load of the shop. These problems had rather loose random due dates with the average due date allowance of six times the processing time of an average job. The corresponding results for the tight due date setting having the average allowance of three times

[11] In the pilot studies, the following values of the rule parameters were tested: $\kappa = 1.5$, 2.0, 3.0 and 4.0, $\beta = 1.0$, 2.0 and 3.0, $\gamma = 0.0$, 0.3, 0.5, 0.7 and 1.0, and $\alpha = 0.0$, 0.3, 0.5, 0.7 and 1.0. Most of the combinations of the parameters worked according to the intuition given in the text. The results for the reported parameter combinations were most consistent, although some others performed reasonably well, too, e.g. the composite model with $\alpha = 0.0$ and $\gamma = 1.0$, or lead time estimates coordinated by jobs' weights.

the average processing time are shown in table 5-3. The performance of the rules in terms of the portion of tardy jobs is shown in tables 5-2 and 5-3 in parenthesis. These results are shown in a graphical form as well. The weighted tardiness graphs for both slack and tight due date settings are depicted in figure 5-3. The average portion of tardy jobs is graphed in figure 5-4.

Table 5-2: The normalized weighted tardiness averaged for the three shop types. Slack due dates. The average portion of tardy jobs is shown in parenthesis.

	Normalized Weighted Tardiness (Portion of Tardy Jobs):				
	Load estimates:				
Rule:	80%	85%	90%	95%	97%
FCFS	.278 (16.5)	.546 (23.0)	1.173 (34.3)	2.692 (51.2)	3.390 (52.3)
EDD	.022 (5.1)	.073 (8.7)	.197 (20.4)	1.222 (45.3)	1.899 (51.0)
S/RPT	.018 (5.3)	.034 (9.6)	.078 (19.7)	.919 (48.6)	1.503 (54.2)
WSPT	.110 (12.7)	.208 (16.1)	.348 (20.1)	.617 (24.3)	.710 (24.1)
$CoverT_{con}$	.019 (6.7)	.031 (10.2)	.055 (15.2)	.199 (23.1)	.294 (25.7)
$CoverT_{prio}$	.015 (4.8)	.025 (6.6)	.041 (8.7)	.198 (16.8)	.268 (17.5)
$CoverT_{comp}$	.016 (4.1)	.024 (6.7)	.043 (9.2)	.193 (17.4)	.268 (17.8)
$CoverT_{iter}$	.012 (5.0)	.023 (8.7)	.049 (12.9)	.204 (22.5)	.294 (24.8)
AU_{con}	.016 (4.3)	.029 (6.8)	.046 (8.6)	.199 (16.8)	.291 (18.8)
AU_{prio}	.015 (4.2)	.025 (5.7)	.046 (7.6)	.189 (15.2)	.281 (17.4)
AU_{comp}	.015 (4.6)	.024 (6.1)	.038 (8.0)	.190 (15.7)	.277 (18.2)
AU_{iter}	.010 (3.7)	.021 (6.2)	.037 (8.0)	.182 (15.9)	.267 (17.8)

The conventional rules did not perform satisfactorily in the test problems. The due date oriented rules, EDD and S/RPT, are reasonable in light load and loose due date situations, but their performance deteriorates quickly when the load increases beyond 85% with tight due dates and beyond 90% with slack due dates. WSPT while never extremely bad is far from the top in most cases.

The modified CoverT and AU rules were superior to the other rules for all shop-load conditions. The basic AU rule was up to 10% better than CoverT in the weighted tardiness measure, and 10-25% better in the portion of tardy jobs. The priority-based lead time estimates improved the weighted tardiness performance of the modified CoverT by 5-20% on average. Moreover, the portion of tardy job was improved up to 30%. The AU rule gained less from the priority dependent waiting time estimation while the AU rule with lead time iteration was the best overall rule in both average weighted tardiness and average total cost. CoverT's improvement due to the lead time iteration was usually less than with the static priority based lead time estimates. Its improvement relative to AU in heavily loaded shops is due to the more global nature of its priority index. CoverT minimizes the average contribution to the weighted tardiness that is the major concern when several jobs are interacting intensively in contention over the machines. The AU rule is designed to minimize any local contribution to the tardiness. Hence it is at its best in more decomposable problems in lightly loaded shops. AU's global performance can be

Table 5-3: The normalized weighted tardiness averaged for the three shop types. Tight due dates. The average portion of tardy jobs is shown in parenthesis.

	Normalized Weighted Tardiness (Portion of Tardy Jobs):				
	Load estimate:				
Rule:	80%	85%	90%	95%	97%
FCFS	.753 (36.8)	.922 (42.1)	2.329 (60.9)	3.033 (64.5)	5.984 (76.4)
EDD	.354 (31.4)	.444 (38.2)	1.662 (67.2)	2.360 (69.7)	4.465 (80.5)
S/RPT	.269 (36.1)	.338 (43.0)	1.586 (74.5)	2.062 (74.9)	4.063 (85.1)
WSPT	.247 (24.3)	.296 (26.3)	.556 (32.1)	.666 (34.1)	1.079 (36.8)
$CoverT_{con}$	.101 (22.4)	.121 (26.4)	.342 (40.0)	.432 (38.7)	.777 (44.3)
$CoverT_{prio}$	.088 (17.8)	.102 (20.6)	.325 (29.1)	.416 (30.2)	.741 (34.5)
$CoverT_{comp}$	.087 (17.4)	.099 (20.7)	.322 (31.4)	.412 (30.8)	.735 (35.4)
$CoverT_{iter}$	.092 (22.4)	.100 (22.3)	.311 (35.4)	.401 (35.1)	.731 (37.8)
AU_{con}	.101 (18.4)	.112 (20.7)	.332 (30.2)	.422 (33.1)	.754 (35.4)
AU_{prio}	.096 (17.0)	.109 (19.0)	.327 (29.0)	.404 (30.1)	.752 (34.6)
AU_{comp}	.092 (17.3)	.107 (21.1)	.331 (31.3)	.416 (32.0)	.773 (35.2)
AU_{iter}	.080 (17.2)	.096 (19.8)	.292 (29.0)	.380 (31.2)	.734 (35.1)

improved by "costing" the congested machines that a job needs later on, thus reducing the immediate priority of a slack job in CoverT-like manner (see chapter 6 for details).

5.4.2 Other Measures of Performance

The robustness of the AU and modified CoverT rules indicates that they must be reasonably good in utilizing the machine resources. The statistics for work-in-process inventories and the work-in-shop measures, shown in tables 5-4 and 5-5 for all rules and different load conditions, confirm that there was no deterioration of the performance of AU and modified CoverT for these objectives. Quite the contrary, the AU rule ranks among the best two or three for both WIP and WIS measures in most problems.

The maximum weighted tardiness measures were not recorded systematically in the dynamic job shops. Several observations indicated that the AU and CoverT rules are also superior to the other rules in this measure of performance, as was the case in the static flow shops reported in chapter 4.

Recognizing the multiple objectives of scheduling, the rules are also compared in terms of combined costs of inventory holding, fixed charges for tardy jobs, and the variable tardiness penalty. The average tardiness penalty is set to twenty times the inventory holding cost, and the relative fixed cost for late shipment equals half of the tardiness penalty of an average job delayed by its processing time. Thus the total costs are:

$$\text{Total Cost} = 2.0*(\text{Norm. Weighted Tardiness}) + (\%\ \text{Tardy Jobs}) + 0.1*(\text{Norm.WIP}) \tag{5.10}$$

Figure 5-3: The weighted tardiness for the different levels of load averaged over the three shop types. Slack (- - -) and tight due date setting (——).

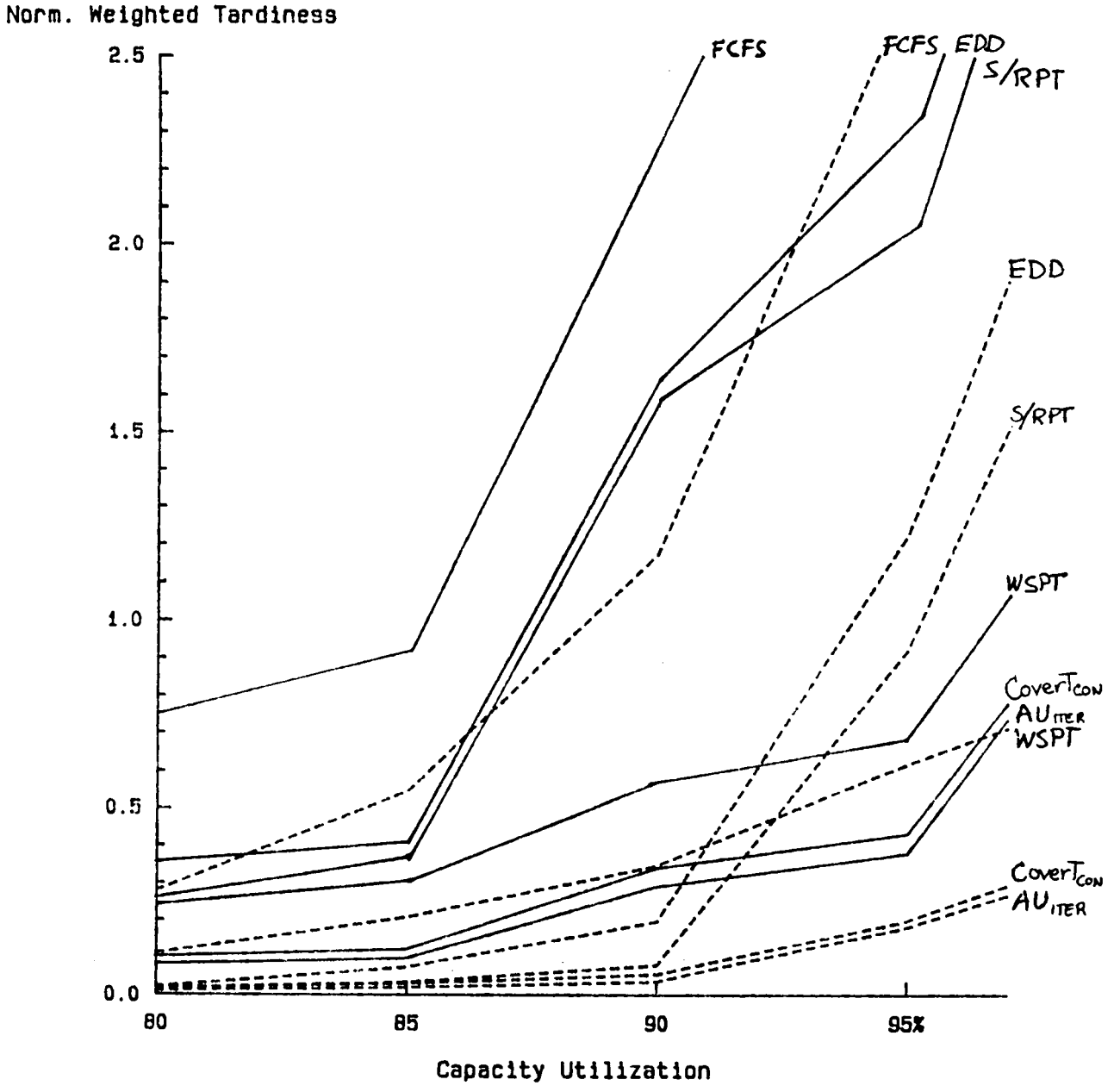


Figure 5-4: The percentage of tardy jobs for the different levels of load averaged over the three shop types. Slack (- - -) and tight due date setting (——).

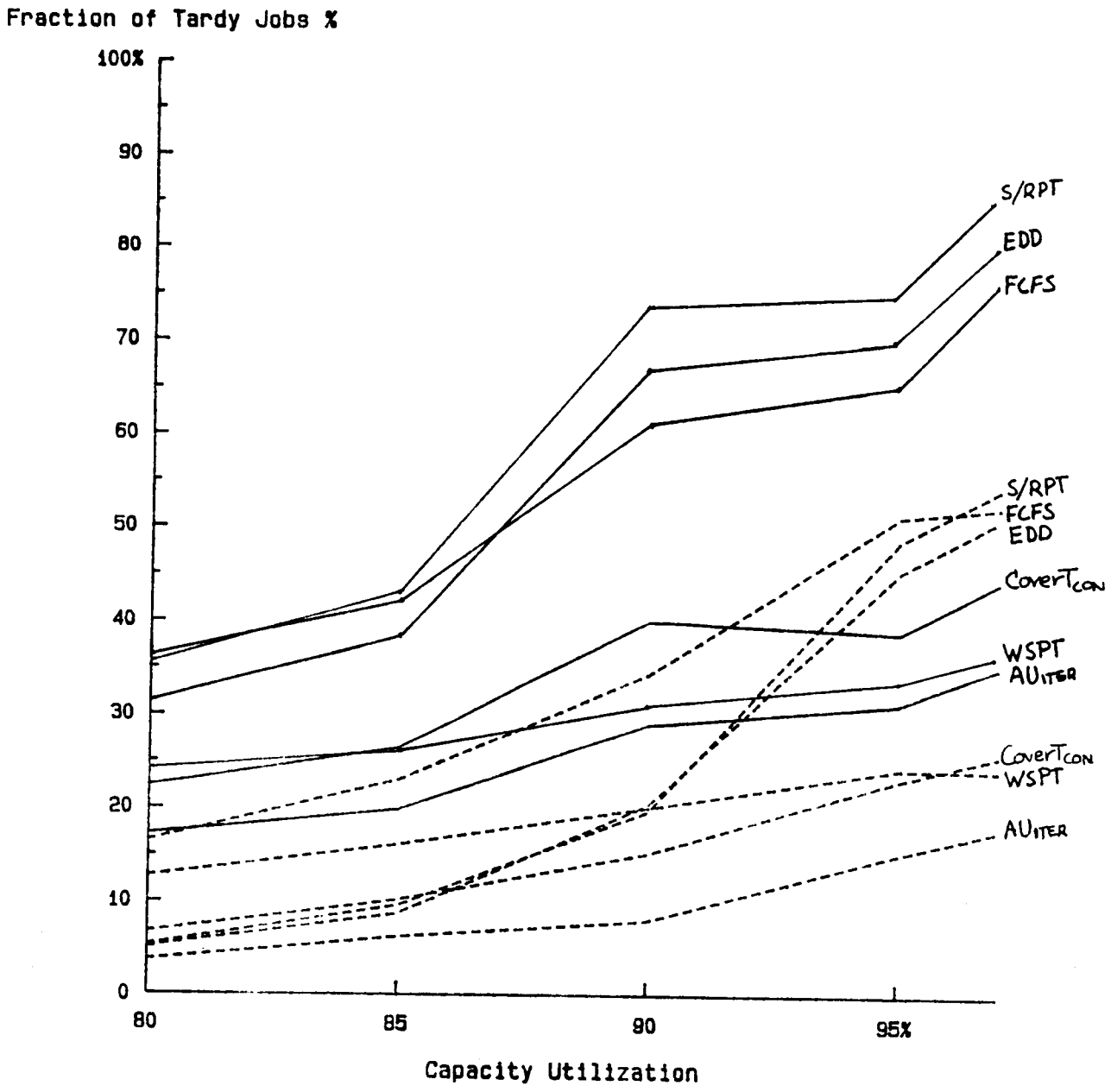


Table 5-4: The normalized WorkIn-Process (WIP) inventory averaged for the three shop types, slack due dates. The Work-In-Shop (WIS) is shown as well.

	Normalized WIP (WIS):				
	Load estimate:				
	80%	85%	90%	95%	97%
Rule:					
FCFS	2.55 (6.85)	3.20 (6.88)	4.46 (7.38)	6.32 (8.35)	6.80 (8.91)
EDD	2.30 (6.46)	2.78 (6.05)	3.47 (5.91)	4.37 (5.85)	4.69 (6.26)
S/RPT	2.62 (6.58)	3.12 (6.30)	3.86 (6.08)	4.72 (5.80)	4.96 (6.24)
WSPT	2.23 (6.98)	2.74 (7.14)	3.59 (7.83)	4.86 (8.78)	5.35 (9.37)
$CoverT_{con}$	2.45 (6.56)	2.97 (6.29)	3.81 (6.15)	4.74 (6.18)	5.11 (6.61)
$CoverT_{prio}$	2.50 (6.59)	3.05 (6.34)	3.92 (6.29)	4.83 (6.17)	5.34 (6.93)
$CoverT_{comp}$	2.45 (6.58)	2.98 (6.30)	3.90 (6.27)	4.77 (6.17)	5.27 (6.84)
$CoverT_{iter}$	2.43 (6.53)	2.97 (6.29)	3.86 (6.11)	4.77 (6.16)	5.12 (6.64)
AU_{con}	2.33 (6.54)	2.81 (6.21)	3.49 (6.14)	4.33 (6.22)	4.77 (6.77)
AU_{prio}	2.34 (6.55)	2.81 (6.23)	3.55 (6.18)	4.41 (6.29)	4.87 (6.75)
AU_{comp}	2.29 (6.55)	2.78 (6.16)	3.44 (6.05)	4.31 (6.17)	4.70 (6.56)
AU_{iter}	2.38 (6.51)	2.86 (6.21)	3.59 (6.14)	4.41 (6.30)	4.98 (6.86)

Table 5-5: The normalized Work-In-Process (WIP) inventory averaged for the three shop types, tight due dates. The Work-in-Shop (WIS) is shown as well.

	Normalized WIP (WIS):				
	Load estimate:				
	80%	85%	90%	95%	97%
Rule:					
FCFS	2.85 (4.29)	3.08 (4.36)	4.64 (5.45)	5.20 (5.97)	7.40 (7.90)
EDD	2.43 (3.64)	2.54 (3.59)	3.34 (3.97)	3.75 (4.25)	4.74 (5.08)
S/RPT	2.59 (3.62)	2.70 (3.61)	3.58 (3.96)	3.77 (4.15)	4.63 (4.88)
WSPT	2.43 (4.31)	2.63 (4.47)	3.78 (5.46)	3.97 (5.62)	5.57 (7.10)
$CoverT_{con}$	2.57 (3.77)	2.71 (3.77)	3.66 (4.25)	4.06 (4.69)	5.34 (5.78)
$CoverT_{prio}$	2.56 (3.76)	2.66 (3.73)	3.83 (4.51)	4.11 (4.83)	5.38 (5.91)
$CoverT_{comp}$	2.55 (3.74)	2.67 (3.72)	3.74 (4.35)	4.10 (4.77)	5.34 (5.84)
$CoverT_{iter}$	2.52 (3.75)	2.62 (3.75)	3.71 (4.36)	3.96 (4.67)	5.25 (5.78)
AU_{con}	2.46 (3.77)	2.60 (3.78)	3.46 (4.54)	3.83 (4.63)	5.21 (5.99)
AU_{prio}	2.46 (3.78)	2.61 (3.77)	3.66 (4.51)	3.88 (4.70)	5.25 (5.98)
AU_{comp}	2.44 (3.72)	2.54 (3.66)	3.57 (4.36)	3.88 (4.63)	5.27 (5.94)
AU_{iter}	2.50 (3.74)	2.66 (3.77)	3.67 (4.49)	3.95 (4.66)	5.32 (6.06)

These costs, averaged over the different shop types, are shown in Figure 5-5 for slack and tight due dates separately. AU and CoverT are again far superior to the other rules tested. The standard AU is approximately as good as the best version of CoverT, using enhanced lead time

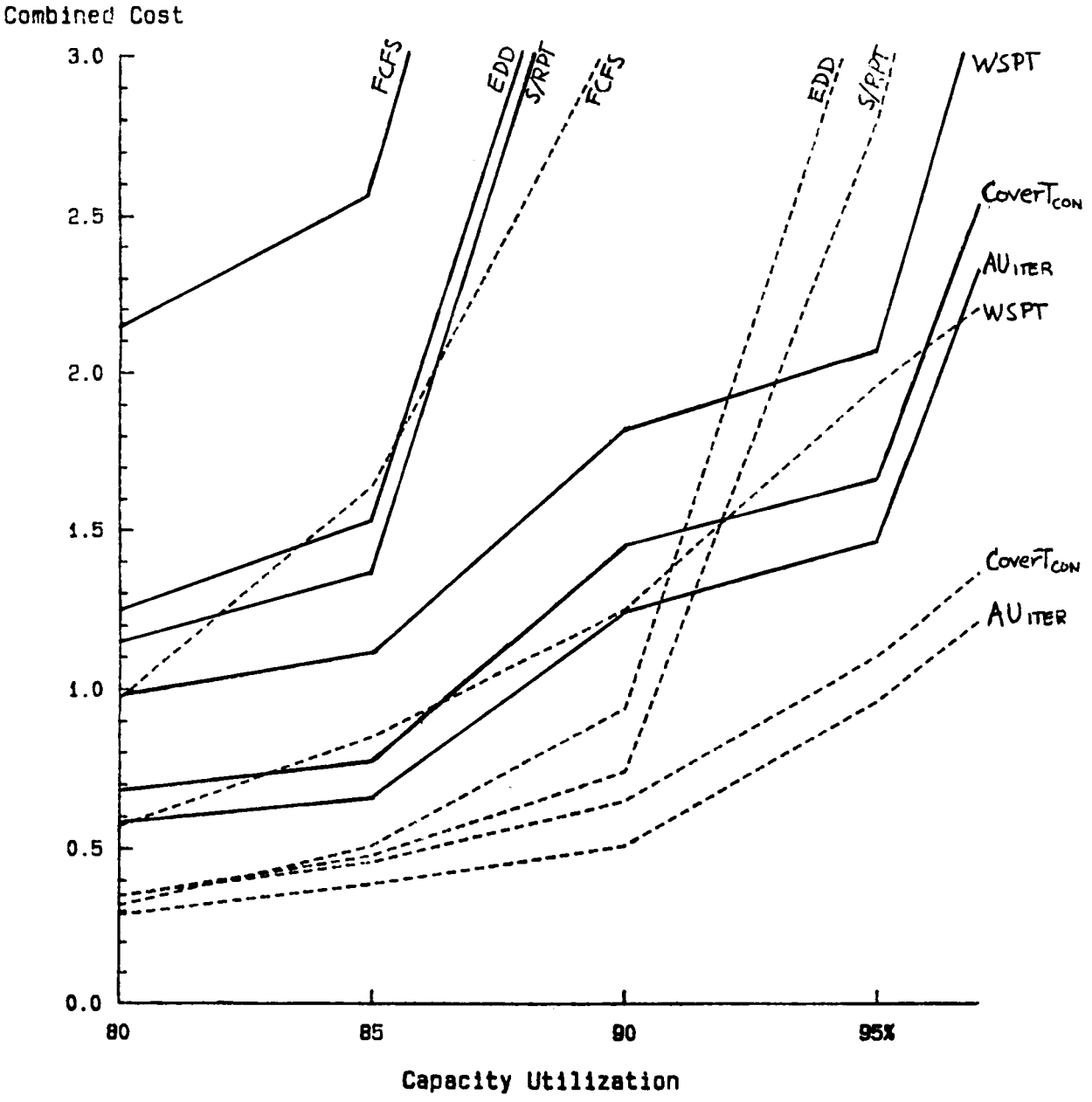
estimation in the problems with long average lead time allowance. But the enhanced CoverT is better with shorter lead time allowances. However, the AU rule with lead time iteration is the best by a consistent margin over all the problems studied.

5.5 Conclusions

In this chapter, we have discussed the evaluation of priorities for slack jobs under the weighted tardiness criterion in dynamic job shops. The new state dependent rules have surpassed the previous standards, represented by the CoverT rule, in several ways. First, we have improved the original CoverT rule by introducing new static waiting time estimation methods which yield reductions of 5-30% in the weighted tardiness and portion of tardy jobs, depending on the characteristics of the problem. More specifically, the new priority-based lead time estimates sometimes suffice for obtaining the best performance of all rules tested. Second, the basic version of our Apparent Urgency rule is better than CoverT in terms of weighted tardiness and number of tardy jobs in most moderately loaded shops with a capacity utilization of less than 95%. CoverT is designed to minimize global average weighted tardiness, hence it performs well in tightly coupled problems in congested job shops. The AU rule was designed to be locally optimal. It performs best in decomposable problems in lightly loaded shops but needs additional coordination in heavily loaded shops. Third, the lead time iteration makes the AU rule consistently better than CoverT in weighted tardiness costs, in inventory-holding costs, and in rush-shipping costs.

The new rules use the information of the load and status of the shop indirectly, either in setting some static lead time estimates or in adjusting dynamically the lead time estimates according to the simulated performance of the rule. The look-ahead and waiting time estimation parameters have been fixed, based on aggregate load estimates, over an extended busy period. The lead time iteration generates waiting time estimates which reflect the specific load conditions but are not derived from any direct measure of the load. An obvious improvement to the lead time iteration would be to initialize it with the new priority-based waiting time estimates. The testing of possible gains is left for future studies. In practical scheduling situations, the scheduler would probably be tempted to adjust the the parameters of the dispatching rule in anticipation of certain kind of shop load conditions in the near future. In next chapter, we will analyze the consequences of dynamic adjustment of the slack evaluation parameters based on the anticipated load on the machines. If this kind of locally opportunistic adjustment can be coordinated automatically to ensure global performance, the resulting dispatching system would eliminate the initialization problems of the parametrized rules.

Figure 5-5: An example of combined inventory, tardiness and rush order costs for different load levels averaged over the three shop types. Slack (– – –) and tight (——) due date settings.



6. Coordination of Job Priorities via Direct Load Information

Summary

The purpose of this chapter is to study the coordination of job priorities with the use of detailed state information in a "self-adjusting" priority rule. The efficiency of priority assignment depends, intuitively speaking, on the opportunistic use of resources according to the local load in the machine centers and on the coordination of the dispatching decisions across machines according to more global characteristics of load in the shop. Hence we make the look-ahead and waiting time estimation parameters of the myopic Apparent Urgency rule dependent on the estimates of *direct* load information, such as the anticipated lengths the machine queues and the average urgency of the jobs. The analytical results and experiments with the constant AU rule in the previous chapters have suggested that the average number of critical jobs on a machine might determine the most appropriate look-ahead factor and waiting time estimates. Despite extensive testing of different adjustment procedures, however, no consistent improvement over the best constant values of the parameters has been obtained.

The priority of a job on the current machine can also be adjusted according to its relative priority on the next machine to avoid congestion of the shop while allowing the most urgent jobs to be rushed through. But again several attempts to implement this "probing" of a job's relative urgency on the next machine did not improve consistently the weighted tardiness performance of the AU rule. These negative results parallel, by and large, with the earlier findings in the case of average (non-weighted) tardiness scheduling.

6.1 Introduction

The performance of most dispatching rules varies greatly when applied in different shop conditions or when measured with some modified objective function. Even the Apparent Urgency rule that has been robust in our tests requires some adjustment of its parameters for extreme load conditions. Hence we risk some opportunity costs when using fixed parameter values.[1] Could the parameter of the priority index be adjusted to the temporary variations of the load in the shop, or even at the level of each machine as well ? If the priority index parameters were adjusted continuously, the consistency of the priority assignments might be lost. The indications of priority coordination problems are obvious. The look-ahead adaptation in the single machine case worked better in small problems than for a large number of jobs. In the dynamic job shops, the lead time iteration works better on several loosly coupled problems, i.e., separate busy periods in a lightly loaded shop. Furtermore, the combination of the look-ahead adaptation with the lead time iteration was not worthwhile. Even though some immediate savings seem attainable through more opportunistic use of resources, the gains are often more than offset by the less advantageous consequences of these decisions on the machine loads later on.

[1] For example, the AU rule with look-ahead parameter $\kappa = 2.0$ yields systematically 10% lower cost than $\kappa = 4.0$ over a period with average capacity utilization approximately 80%, but $\kappa = 4.0$ is almost equally better when the load approaches 95%.

Consider a dispatching decision when the main objective is to minimize weighted tardiness and you have a reliable forecast of the future load. If the machine is a bottleneck with a a long queue it should be loaded efficiently according to the WSPT rule: short and expensive jobs should be given higher priority despite their slack. Any idle time on that machine should be avoided as well. On the other hand, if a machine will have rather light load in the future, jobs can be sequenced according to the shortest slack, allowing idle time if necessary for expediting a critical job. Naturally, the expected waiting times of jobs on these two machines will be very different since a long job would probably stay for a long time in the bottleneck queue but could be processed quickly on the slack machine. These heuristic rules can be implemented by proper parameter setting of the AU rule: a long look-ahead (large **k**) and long lead time estimates (large β) are needed for jobs going through a bottleneck machine, and smaller **k** and β on the slack machines. Moreover, the priority index of a job that is going next into a long tardy queue should be lowered on the current machine.

In the previous studies in chapters 4 and 5, the lead time estimation and look-ahead parameters were set on a constant level that was appropriate for the average utilization of the machines. However, the lead time estimation with a constant β did not work as well in a shop with bottleneck machines as in an evenly loaded shop, as indicated by improvement obtained with the lead time iteration. We have already found in chapter 3 that the bottleneck machines determine the structure of the optimal schedule in a proportionate flow shop with unit jobs. The experience with some commercial job shop scheduling systems, such as OPT [31], supports also the view that an efficient loading of the bottleneck machines has a primary role in job shop scheduling. Hence we might expect some gains from increasing the κ and β parameters on a bottleneck machine to adapt the rule to the load conditions. But by how much these parameters can be adjusted before losing the consistency of the dispatching decisions over time and across the machines. To address the coordination problem, we need to measure the trade-off between the cost of delaying the processing of a job (time cost of a job) and the cost of using the machine (opportunity cost of the machine). This ratio would allow comparisons across the opportunity costs of different machines, and the costs of the jobs across different time periods as well. Previous research has concentrated on the study of the behavior of simple rules which do not encompass the cost information required for proper coordination [3, 17, 33, 58]. A common state indicator used before has been the length of a queue, measured in terms of the number of jobs, work contents, the utilization of the machine, or the expected waiting time of a job. As Maxwell [49] has shown these measures are closely related and hence equally suitable surrogates for estimating the opportunity costs of the machines. Bertrand [10] proposes a direct loading procedure to estimate the job lead times with some coordination of job priorities. His adjustable dispatching rule uses aggregate load information in the form of the average due date of the jobs in the shop and the lengths of the queues. Examples of state coordinated rules that use the queue length as state indicator are the (anticipated) work in next queue, (A)WINQ, rule and the composite rules that include it [3, 17, 58]. This "machine look-ahead", or probing component lowers the priority of a job on the current machine if it would exit into a long queue at the next machine. In order to avoid overreaction, anticipated queue length can be used instead the currently observed ones. A component similar to the WINQ but normalized by the total work in the shop is included in the Dynamic Composite Rule (DCR) in [17]. DCR is the only priority rule that adjusts the priority index directly to the load on the current machine as well (see section 2 below).

Emery [20] designed a multi-stage dispatching procedure which uses several state indicators to screen jobs from the final dispatching. The components of this optimum seeking search procedure will also be discussed below.

Some of the previous results in this thesis have indicated how to choose the most efficient parameter values for the Apparent Urgency rule. The purpose of this chapter is to study systematically the possibilities of periodic adjustment of the priority indexes and the coordination of dispatching decisions using anticipated state information. In section 2, we specify adaptation rules for the AU priority index. The efficiency of the new "automatically" state dependent rule is tested against the constant rules in the problems reported the previous chapter. The results, which do not indicate any consistent improvement in the weighted tardiness performance of the AU rule, are analyzed in section 3. Section 4 concludes the experiment.

6.2 Load Estimates as Priority Coordination Parameters

6.2.1 Different Mechanisms of Priority Coordination

The dispatching process is reminicient of any coordination problem. We have to make an opportunistic local decision, taking into account the interactions of several local decisions to guarantee a good global performance. Hence we have to anticipate the future status of the shop *before* knowing the final dispatching decisions. In the myopic dispatching approach, the complexity of local decisions is reduced by the greedy search forward in time. But this computational efficiency comes at the cost of not using any information of future status of the machines. The uncertainty of the future is endogenous since it could be removed if we could solve the problem using forward dynamic programming or some other exact method.[2] However, we can get an *estimate* of the future load by an aggregate analysis of the problem data or through simulation with a myopic dispatching rule. Different methods and time horizons of state estimation are classified in our framework in chapter 2. Some of this kind of coordination was included in the Dynamic Composite Rule (DCR), a 3-parameter state dependent rule for average tardiness scheduling [17]. The priority index at time t for jobi, $\pi^{ij}_{DCR}(t)$, is determined as follows (o_{ij} denotes the due date of operation j, set according to the standard lead times of the job class in question, p_{ij} the processing time):

$$\pi^{ij}_{DCR}(t) = o_{ij} - p_{ij} + b\,[\Sigma_k p_{kj}]^r\, p_{ij} + h\, W_{nq}\,, \qquad (6.1)$$

where b, r and h are parameters constant during a particular simulation run, $\Sigma_k p_{kj}$ is the sum of the processing times in the queue from which the selection is to be made, and W_{nq} is the ratio of the sum of the processing times in the next queue to the sum of the processing times of the jobs in all the other queues in the shop. DCR is actually a minimum slack rule (the job with the lowest index value has the priority) that is adjusted for the congestion of the current queue and of the next queue. With proper parameter selection, for example b = 15, r = 1, h = 0, or b = 0.3, r = 1, h = 160, this combination rule performed better than its component rules in terms of average tardiness and conditional tardiness. The DCR index in equation (6.1) represents the time dimension (slack and processing time) of priority. The AU priority index is expressed in terms of the marginal tardiness cost per time unit on the machine (high value gets the priority):

[2] Here we assume that the problem data is known with certainty.

(6.2)

$$\pi^{ij}_{AU}(t) = (w_i/p_{ij})\, e^{-[d_i - t - p_{ij} - r_{ij+1} - W_i^{j+1}]^+ / \kappa \bar{p}}.$$

Here W_i^{j+1} is the expected lead time on the subsequent machines and $\bar{p}$ is the average processing time of the unscheduled jobs. Parameter κ determines the scale of the measure of the slack of the job. This AU index can be represented in the slack time domain as well. Any monotonic transformation of the priority index will maintain the ranking of priority assignment unaltered. Thus we can take the negative of the logarithm of the AU index in equation (6.2), following [54]:

(6.3)

$$r_{ij}(t) = [d_i - t - p_{ij} - r_{ij+1} - W_i^{j+1}]^+ + \kappa \bar{p}\, \ln(p_{ij}/w_i).$$

Comparing this to the DCR index above, we notice that both have a similar slack component. The second component is a function of the queue length multiplied by a term containing the weight and the processing time, or a "utilization factor" [54]. The main difference is that the DCR index does not include a job specific weight, w_i. The third term of DCR, the relative length of the next queue, in effect adds a "time penalty" to the local slack due to the heavy load on the next machine. In the original study [17], the second, "SPT-like" term, improved the performance of the plain slack term considerably, whereas the third, "probing" term, did not improve all tardiness related measures. In the following section, we experiment with a probing term in the opportunity cost domain.

Emery [20] designed an optimum seeking procedure for minimizing average inventory holding, earliness and tardiness costs in a static job shop scheduling problem. His priority function is a composite rule. The priorities of schedulable jobs are determined in two stages. First, six screening criteria are applied in a sequence to eliminate the jobs which are not "critical" for the final priority evaluation. The six threshold criteria are the job's 1) external priority class, 2) CoverT index value, 3) time spent in the present queue, 4) remaining processing and queue time over current processing time, 5) processing time of current operation, and 6) size of next queue. The second stage is an ordinary dispatching decision among the jobs which survived the screening stage. The priority index function is a weighted sum of the terms used in each of the screening stages (except the external priority). Thus there are 5 parameters to be adjusted over some region using simple heuristic grid search. In the example reported in [20], the optimum seeking process resulted in total inventory holding and tardiness cost that was 5% lower than the starting solution and 10% lower than the cost with the plain CoverT rule. However, the effects of each of the coordination terms upon the performance of the rule are not explained in [20].

6.2.2 The "Self-Adjusting" AU Priority Index

We propose a new adjustable form of the AU rule to test the need for coordination in variable load situations. The AU priority index function can be adjusted "automatically" to the variations of the load on the different machines and in the shop over time following the intuition given in Introduction:

- long look-ahead (large k) and long lead time estimates (large β) should be used for jobs going through a heavily loaded (bottleneck) machine with long queues, and smaller k and β on the slack machines having short queues.

- the priority of a job on the current machine should be reduced if it is going next onto a machine which will have more urgent jobs to process (and possibly a longer queue). We could also expedite jobs going to an idle machine that will later have enough load and delay jobs going to heavily loaded machine toward the end of its busy period.

The priority index of job i at machine j is the basic AU priority index with the following modifications:

1. The waiting times for each of the remaining operations after the current one, W_{ik}, are estimated based on the use of the average anticipated queue lengths on the machines, $\mathbf{q} = (q_1, q_2, \ldots, q_m)$. $\mathbf{q}$ consists of weighted averages of the current, anticipated and total average queue lengths for the look-ahead estimation (see next section). The waiting time estimation parameter, β, of the priority-based model[3] can now be determined on the basis of the anticipated queues:

$$W_{ik}(\mathbf{q}) = \beta(\mathbf{q})\,\bar{w}/\pi_{ik} + 0.5\,\bar{p}\,, \tag{6.4}$$

$$W_i^j(\mathbf{q}) = \Sigma_{k=j}^{m} W_{ik}(\mathbf{q}),$$

where π_{ij} is the WSPT priority index value of job i on machine j, $\bar{w}$ is the average weight and $\bar{p}$ is the average processing time of the jobs in shop. β is some non-decreasing function of $\mathbf{q}$ to be determined later.

2. The length of the look-ahead is also determined by the anticipated queue lengths, $\mathbf{q}$. That is, $\kappa = \kappa(\mathbf{q})$ in the state dependent AU priority index:

$$\pi_{AU}^{ij}(t, \mathbf{q}) = (w_i/p_i)\, e^{-[d_i - t - p_{ij} - r_{ij+1} - W_i^{j+1}(\mathbf{q})]^+/\kappa(\mathbf{q})\,\bar{p}}\,. \tag{6.5}$$

We use the average anticipated queue length because the jobs arriving to the queue in the near future will compete for the machine capacity with the jobs which might not yet have arrived.

3. Add probing or a further adjustment of the priority index for the load on the next machine. The correction term in the opportunity cost space considered here is the difference between the anticipated AU priority index value of job i on next machine n after time T_1, $\pi_{AU}^{in}(t + T_1)$, and the anticipated average priority index value of the jobs being processed on the next machine at the same time, $\pi^n(t + T_1)$, or:[4]

$$\Delta\pi_{AU}^{in}(\pi^n, t, T_1) = \pi^{in}{}_{AU}(t + T_1) \cdot \pi^n(t + T_1)\,. \tag{6.6}$$

The implementation of this probing term depends on the load information available in the scheduling system discussed in the next section.

4. The "self-adjusting" AU priority index is the sum of the urgency terms on the current and the next machine:

$$\pi_{Adj}^{ij}(t, \mathbf{q}, \pi^n) = \pi_{AU}^{ij}(t, \mathbf{q}) + c\,\Delta\pi_{AU}^{in}(\pi^n, t, T_1)\,. \tag{6.7}$$

The probing of the anticipated urgency differential on the next machine is controlled by setting the parameter c>0 above.

---

[3] See chapter 5 for the derivation of the priority-based waiting time model.

[4] The priority could be decreased if the priority index differential increases later after time period $T_2 > T_1$.

In the earlier studies we have found that the AU priority index could be adjusted to certain load conditions if the appropriate state information were available. We test three further refinements:

1. The waiting time estimates W_{ij} can be adjusted according to the average tardiness of the jobs in the anticipated machine queues as discussed in chapter 5. If the jobs have long slack on average, the waiting time estimation function is flattened using $\gamma < 1.0$ in equation (5.5).

2. The look-ahead parameter κ can be adjusted according to the value of the job, if there are only few jobs anticipated in the queue: more expensive jobs should have shorter look-ahead or a lower value of κ as discussed in chapter 2.

3. The priority index of a job going later to heavily loaded machines can be reduced by allocating the expected marginal tardiness cost over the "effective" remaining processing time. The effective processing time is the processing time on the current machine plus a portion of remaining processing time that depends on the load on the subsequent machines. This modification gives the AU rule "CoverT-like" global coordination in heavily loaded shops: the priority index of a job with several operations remaining reaches the full WSPT value just when job is within the crash time from its due date.

Before testing the new coordinated AU rule, we specify the load estimates needed for the priority index functions given above.

6.2.3 Estimation of State Indicators

The estimation of the two state indicators proposed above, the average queue length and the average priority index values of the schedulable jobs, requires a simulation of the problem with some dispatching rule, for example the basic AU rule. The horizon of this forward simulation can extend from the extrapolation of the current status by a few tentatively simulated decisions (as with the AWINQ rule) to the use of long forecasting and planning horizons. Since we are mainly interested in the waiting time estimates in the near future, a rolling horizon procedure can be implemented.

To estimate the queue length, we should count only the "critical" jobs in the queue. The critical jobs are those jobs i for which the urgency function for operation j, $e^{-[d_i-t-p_i-r_{ij+1}-W_i]^+}$, exceeds some treshold value. Alternatively, the critical queue length can be estimated as a sum of the urgency function values of the jobs in the queue. Both of these critical queue length estimates characterize the anticipated dispatching situation better than the queue length including all jobs, even those with considerable slack. The critical queue lengths could be weighted averages of three components: the current queue length at a machine, the average anticipated queue length of that machine, and the average queue lengths averaged over all machines of the shop. In the pilot studies, the current queue length was found too volatile to be used as a rule parameter in the simulation. More stable queue statistics were obtained through

moving averages[5] of the average machine queue lengths and the global average queue length of the shop. The forecasting horizon was set to $100*\bar{p}$ and the planning horizon to $60*\bar{p}$. Second, the anticipated average priority index value on the next machine reflects the urgency of the load on that machine. This index can be obtained as a sum of the projected[6] priority indexes of the schedulable jobs and the those of the jobs scheduled to arrive on that machine, divided by the average critical queue length. If on a bottleneck machine the load is temporarily higher than average, the correction term works like the AWINQ rule, preventing further congestion of the machine. Otherwise, a job with relatively high projected Apparent Urgency index can be rushed to that machine against the prescription of the AWINQ rule.

6.3 Experiments in Dynamic Job Shops

6.3.1 Job Shop Problem and Rules Tested

Different versions of the new coordinated AU rules were tested in the problems used in chapter 5. The performance of the rules was recorded over a complete busy period of the shop to count for varying shop conditions. The rules were:

Apparent Urgency

Constant — The AU rule using the priority based waiting time estimates with $\beta = 1.0$, and look-ahead factor $\kappa = 3.0$.

Indirect — The AU rule with selected constant priority based lead time estimation and look-ahead parameters used in the previous experiment in chapter 5.

Lead Time Iter. — The AU rule with lead time iteration as benchmark from the previous experiment.

Direct Load Information, Coordinated with λ_β and λ_k

The AU rule with a dynamic priority based waiting time and look-ahead estimation. Coordination parameter λ_β represents the weight of the global average queue length, $\bar{q}(t)$, in the queue length estimate, $q_m(t)$, of machine m, for the waiting time estimation parameter, $\beta_m(t)$, at time t:

$$q_m(t) = \lambda_\beta \bar{q}(t) + (1-\lambda_\beta)\bar{q}_m(t), \tag{6.8}$$

$$\beta_m(t) = (q_m(t))^{0.5},$$

$$W_{im} = \beta_m \bar{w}/\pi_{im} + 0.75\,\bar{p}.$$

The functional form $\beta = q^{0.5}$ was robust in a pilot study. Coordination parameter λ_k represents the weight of the global queue length in a machine's queue length estimate for the look-ahead parameter $\kappa(t)$ at time t:

---

[5] The queue lengths were averaged continuously over time buckets of the length $10*\bar{p}$. The forward moving average of four of these time buckets was used as the estmate of the anticipated queue length.

[6] The slack estimate of a job was reduced by $2*\bar{p}$.

$$q_m(t) = \lambda_\kappa \bar{q}(t) + (1 - \lambda_\kappa)\bar{q}_m(t), \tag{6.9}$$

$$\kappa_m(t) = (q_m(t))^{0.75}$$

The form $\kappa = q^{0.75}$ was robust in the pilot study.

The parameters $\lambda = (\lambda_\beta, \lambda_k)$ determine to what extent the priority assignment is coordinated by global load in the shop. The weight $1.0 - \lambda$ is given to the specific machine queue statistic. For example, the values $\lambda = (1.0, 1.0)$ assign the same waiting time look-ahead estimates for all machines, adjusted periodically by the changing load in the shop. $\lambda = (0.5, 0.5)$ allow the load of a machine to affect its look-ahead and waiting time estimation. The "effective" processing time approach has been used with the AU rules to improve the clobal coordination in on congested machines.[7]

Next Machine Probing

The AU rule with the probing of the next machine urgency. The anticipated average priority index value on machine n is computed as

$$\bar{\pi}^n(t + T_1) = \Sigma_i \pi^{in}_{AU}(t + T_1)/0.5(\bar{q}_n(t) + \bar{q}(t)), \tag{6.10}$$

where π^{in}_{AU} is job i's projected priority index value on the next machine n. Hence the machines with less-than-average queue length should be easier to load. The probing parameter in equation (6.7) is assigned the value $c = q_n(t)/20.0$ if the correction $\Delta\pi^{in}_{AU} \leq 0$, and $c = 0.0$ otherwise. Hence a machine with a long queue can reject a job with low projected urgency but no machine can "pull" a job even with high projected urgency.

Look-ahead Adjustment

The same AU rule except that the look-ahead parameter κ is reduced for jobs with value w_i above average, especially in a slack shop.

$$\kappa'_m = \kappa_m(1 - (1 - w_i/\bar{w})/\bar{q}). \tag{6.11}$$

Probing and Look-ahead adjustment

Both next machine probing and look-ahead adjustments implemented.

We studied the periodic adjustment of waiting time estimates, or parameters β_m, for CoverT as well. The probing of next machine urgencies was not implemented.

CoverT:

Constant — The CoverT rule with constant priority based waiting time estimates, with $\beta = 2.0$ and $k_c = 2.0$.

Indirect — The CoverT rule with the selected constant priority based lead time estimation parameters used in the previous experiment in chapter 5.

Lead Time Iter. — The CoverT rule with lead time iteration as benchmark from the previous experiment.

---

[7] The effective processing time is $\tilde{p}_{ij} = p_{ij} + \Sigma^n_{k=j+1} \rho_k p_{ik}$, where the machine congestion factor $\rho_k = (\bar{q}_k(t) - 1)/20$.

Direct Load Information

The modified CoverT with priority based waiting time estimation method, using the coordination parameter λ_β = 0.5 or 1.0, with a constant look-ahead tolerance k_c = 2.0.

The parameters of the final experiment were selected from a pilot study. The exponential forms of β- and κ-functions were tested with the following values of the exponent: 0.25, 0.4, 0.5, 0.6, 0.75, 0.8, 0.9, and 1.0. The values of the coordination variables λ tested were 0.0, 0.25, 0.5, 0.75 and 1.0.

6.3.2 The Results of the Experiment

The weighted tardiness results of the experiment are shown in Tables 6-1 and 6-2, for the different load categories and averaged over the three shop types, for slack and tight due dates, respectively.[8] It is obvious that the periodic adjustment of the priority index functions using direct state information fails to improve the average performance of the rules consistently in terms of weighted tardiness or percentage of tardy jobs. Also the inventory holding measures were approximately the same as with the constant rules. The new state dependent AU rules had some promising cases with better performance than the selected constant parameter rules. CoverT, in the contrary, has very stable performance that is worse than the "constant" CoverT in most cases. The percentage of tardy jobs was, however, very low compared to the other rules.

The coordination achieved via the global queue length seems to provide some robustness. The additional state information from the machine queue lengths improves the weighted tardiness performance occasionally, but the gains are, by and large, rather disappointing. CoverT holds some advantage in heavily loaded shops but fails in the case of light load. The amount of coordination does not appear relevant for CoverT. This can be understood by the fact that the CoverT index aggregates the waiting time estimates thus reducing the variance of any estimation errors. Probing the priority on next machine does not improve the average performance of the AU rule although in individual problems the improvement can match the corresponding results in the earlier study with the DCR [17]. The adjustment of the look-ahead parameter κ according to the relative weight of the job had mostly negative consequence.

[8] See previous chapter 5 for description of the job shop problems.

Table 6-1: The performance of the self-adjusting rules in terms of normalized weighted tardiness with slack due dates, averaged for the three shop types. The average percentage of tardy jobs is shown in parenthesis.

	Normalized Weighted Tardiness (Portion of Tardy Jobs):				
	Load estimates:				
Rule:	80%	85%	90%	95%	97%
Apparent Urgency:					
Constant	.016 (4.7)	.030 (6.4)	.048 (7.8)	.195 (16.4)	.287 (19.7)
Indirect	.015 (4.2)	.025 (5.7)	.046 (7.6)	.189 (15.2)	.281 (17.4)
Lead Time Iter.	.010 (3.7)	.021 (6.2)	.037 (8.0)	.182 (15.9)	.267 (17.8)
λ_β, λ_κ	Coordinated Direct Load Information:				
1.0, 1.0	.015 (3.7)	.029 (5.9)	.043 (7.9)	.191 (17.1)	.287 (18.1)
0.5, 1.0	.015 (3.6)	.028 (6.0)	.049 (8.0)	.192 (16.4)	.283 (18.6)
0.5, 0.5	.015 (3.7)	.025 (5.7)	.048 (7.3)	.192 (16.4)	.299 (17.6)
	Coordinated Next Machine Probing:				
1.0, 1.0	.015 (4.3)	.026 (5.9)	.046 (8.0)	.197 (17.3)	.271 (18.3)
0.5, 1.0	.015 (4.2)	.030 (6.6)	.046 (8.5)	.197 (17.2)	.277 (18.5)
0.5, 0.5	.015 (4.3)	.028 (5.9)	.048 (7.9)	.197 (16.3)	.272 (17.5)
	Coordinated Look-ahead Adjustment:				
1.0, 1.0	.015 (4.0)	.026 (6.1)	.044 (8.3)	.191 (17.3)	.299 (19.3)
	Probing and Look-ahead Adjustment:				
1.0, 1.0	.015 (3.7)	.025 (6.0)	.040 (8.3)	.193 (17.2)	.291 (19.6)
CoverT:					
Constant	.017 (4.8)	.032 (6.8)	.047 (8.8)	.195 (16.5)	.271 (18.2)
Indirect	.015 (4.8)	.025 (6.6)	.041 (8.7)	.198 (16.8)	.268 (17.5)
Lead Time Iter.	.012 (5.0)	.023 (8.7)	.049 (12.9)	.204 (22.5)	.294 (24.8)
λ_β, κ	Coordinated Direct Load Information				
1.0, 2.0	.016 (4.2)	.028 (6.4)	.052 (8.7)	.205 (16.3)	.283 (17.0)
0.5, 2.0	.018 (4.3)	.030 (6.3)	.052 (8.6)	.211 (16.1)	.284 (16.8)

Table 6-2: The performance of the self-adjusting rules in terms of normalized weighted tardiness with tight due dates, averaged for the three shop types. The average percentage of tardy jobs is shown in parenthesis.

	Normalized Weighted Tardiness (Portion of Tardy Jobs):				
	Load estimates:				
Rule:	**80%**	**85%**	**90%**	**95%**	**97%**
Apparent Urgency:					
Constant	.096 (16.6)	.115 (18.8)	.329 (29.2)	.406 (31.1)	.766 (36.7)
Indirect	.096 (17.0)	.109 (19.0)	.327 (29.0)	.404 (30.1)	.752 (34.6)
Lead Time Iter.	.080 (17.2)	.096 (19.8)	.292 (29.0)	.380 (31.2)	.734 (35.1)
λ_β, λ_κ	Coordinated Direct Load Information:				
1.0, 1.0	.089 (16.1)	.105 (19.4)	.328 (30.4)	.416 (31.6)	.762 (35.6)
0.5, 1.0	.080 (16.0)	.110 (19.0)	.333 (29.6)	.416 (31.9)	.756 (35.0)
0.5, 0.5	.089 (15.3)	.107 (18.3)	.328 (28.7)	.413 (30.7)	.749 (34.6)
	Coordinated Next Machine Probing:				
1.0, 1.0	.090 (17.0)	.105 (18.9)	.326 (29.5)	.403 (30.4)	.770 (35.9)
0.5, 1.0	.091 (16.9)	.110 (19.6)	.328 (29.9)	.403 (30.2)	.748 (35.5)
0.5, 0.5	.088 (16.2)	.109 (18.6)	.324 (28.5)	.409 (30.3)	.773 (35.3)
	Coordinated Look-ahead Adjustment:				
1.0, 1.0	.092 (16.6)	.112 (19.9)	.341 (30.6)	.425 (32.2)	.760 (35.7)
	Probing and Look-ahead Adjustment:				
1.0, 1.0	.089 (16.2)	.108 (19.6)	.325 (29.7)	.406 (32.0)	.746 (35.6)
CoverT:					
Constant	.097 (16.6)	.113 (19.0)	.330 (29.2)	.417 (30.9)	.750 (37.3)
Indirect	.088 (17.8)	.102 (20.6)	.325 (29.1)	.416 (30.2)	.741 (34.5)
Lead Time Iter.	.092 (22.4)	.100 (22.3)	.311 (35.4)	.401 (35.1)	.731 (37.8)
λ_β, κ	Coordinated Direct Load Information				
1.0, 2.0	.093 (16.4)	.109 (18.4)	.327 (28.0)	.405 (29.4)	.779 (33.8)
0.5, 2,0	.096 (16.3)	.116 (19.1)	.329 (27.1)	.404 (29.1)	.781 (34.0)

6.4 Conclusions

Direct load information has been applied in the dispatching approach without any significant loss of weighted tardiness performance. The coordination is achieved through smoothed queue length information. Our failure to improve the AU and CoverT rules with direct load information should not deter further experiments with self-adjusting rules. First, despite careful pilot studies, the state indicators and parameter estimation functions specified here might not have been the most appropriate ones. Second, any additional constraints (multiple resources, alternative routings) will add to the potential need for more direct dual price information. Third, a conclusion of our results is that indirect load information suffices for good performance in weighted tardiness problems, especially with performance feedback. The self-adjusting rules can be used, however, to find the best *constant* parameter values for the indirect rules.[9]

The implementation of the state dependent rules depends on frequency of the parameter adjustment and the updating of the queue estimates. Actually this direct information should be replaced with more accurate shadow price information (resource rents). The insensitivity of our results to the length of the estimation period allows quite infrequent use of any centralized information. However, the utilization of the urgency probing of the next machine would require more advanced scheduling information system. Other issues, such as the goodness of load forecasts, errors in data, etc. should be studied before practical application.

[9] In the previous studies the parameter selection problem has not been discussed in detail [3, 17, 58]. It has been viewed as an empirical question without any systematic analysis of the superposition of separately estimated parameters.

7. Conclusions and Future Research

7.1 Conclusions

7.1.1 Introduction

The scheduling problems addressed in this thesis are important in practice but intractable for an exact solution. Minimizing the tardiness penalties without risking the costs associated with other important criteria, such as number of tardy jobs, work-in-process inventory and extremely late deliveries, can be viewed as the main goal of production control [15, 33]. Although experienced schedulers perform this task daily in many firms, the exact information requirements and cost trade-offs of dispatching decisions have not been subject to analytical study before. A multitude of *ad hoc* dispatching rules has been suggested [1, 3, 16, 17, 40, 58], but none of these rules has been acceptable for weighted tardiness problems in a natural variety of job shop problems. The family of rules developed and tested in this thesis, the Apparent Urgency rules, have well-defined informational requirements for dominating the previous dispatching rules in weighted tardiness problems. The priority indexes of the new rules are derived from the necessary conditions for local optimality of a dispatching decision. A comprehensive framework is used to categorize the state information that improves the global coordination of the local decisions, including the job attributes, aggregate shop status indicators, and local and global performance feedback. Finally, computationally efficient procedures have been revised and tested to encompass the relevant state information for the parametrized rules. The implementation of the new state dependent rules requires often diagnostic or iterative analysis of the anticipated load or rule performance, but the final scheduling decisions can be supported in a dispatching mode.

It has been shown previously that rather simple linear rules suffice to capture most of the scheduling expertise used in practical as well as in laboratory settings [13, 25], but the attempts to derive the priority rules from the problem data have been few and tentative [42]. Our state dependent rules formalize, in fact, the relevant economic constraints and opportunities facing a rational scheduler. In the next sections, we conclude with the framework and guidelines for the choice of a priority rule in a given scheduling environment. Possible extensions of this research are discussed as pertaining to the promising directions for future research.

7.1.2 The Framework for Information in Priority Rules

Two principal ways to add more information to a dynamic priority dispatching rule are shown in the framework in figure 7-1 below: the extension of the forecasting horizon and type of information feedback, and the extension of the scope and detail of the information. First, in addition to the *observable status* of the job shop, the priority index formula and its parameters can be based on the *anticipated status* within some forecasting horizon. The parameters of the rule can also be adjusted according to the anticipated performance of the rule over a forecasting horizon. An appropriate use of this *performance feedback* ensures the consistency of the dispatching decisions over time.

SCOPE AND DETAIL OF STATUS INFORMATION:	HORIZON OF INFORMATION FEEDBACK: 1. Observable status	2. Anticipated status	3. Performance feedback
A. Local	Slack/RPT Basic AU,CoverT Ch. 2.	Look-ahead estim. (single machine) (Ch. 2)	Look-ahead adapt. (single machine) Ch. 2.
B. Indirect Global	Lead time estimation AU, CoverT Chs. 4 and 5.	Dynamic lead time estimation Ch. 6.	Lead time iteration Chs. 4 and 5.
C. Direct Global	DCR WINQ Ch. 6.	Probing of next machine load AWINQ Ch. 6.	Rule adaptation Emery's rule (Ch. 6.)

Figure 7-1: The classification of the state dependent dispatching rules according to the information used in the priority index. The enhancements of the basic look-ahead rules AU and CoverT are shown along with some previous rules and the pertinent chapters of the thesis.

Second, the spatial coordination of the decentralized dispatching decisions requires the extension of the scope and detail of the machine and load information used by the priority rule. In addition to the *local information* concerning the current machine and the jobs to be scheduled on it, a rule can use some observable and/or anticipated information concerning the load on some other machines as well. This *global information* is indirect if it does not refer explicitly to the attributes of the other jobs or machines. Examples of *indirect* global load information are the distribution of the weights of the jobs and some appropriate estimates of their lead times. The explicit measures of the load on the other machines constitute *direct* global information, such the workload of the next machine, or the opportunity cost of that machine.

The framework illustrated in figure 7-1 shows the principal categories of information required for the implementation of different state dependent rules. The concentration of each of the chapters of the thesis in terms of the additional state information embedded in the new rules is indicated along with a characteristic implementation technique.[1] The value of the additional information depends also on the properties of the specific rule selected for the scheduling system

[1] If a procedure included in the framework is not tested, the corresponding chapter is shown in parenthesis.

and on the quality of the available data in terms of completeness, correctness, accuracy and timeliness.

Two priority rules were superior for weighted tardiness scheduling. Carroll [16] had revised a family of parametrized CoverT rules for average (nonweighted) tardiness scheduling. His original idea was to minimize the total expected tardiness by prioritizing the jobs according to the probability of a job being tardy allocated over its processing time. We have modified CoverT to apply when the jobs can have different delay penalties,or weights.[2] Carroll used some "standard" waiting times of the remaining operations to evaluate the global slack of a job in terms of the probability that a job will be tardy. Our analysis confirms that the conventional lead time estimates, the multiples of the "crash time" or the remaining processing time, are appropriate for the average (nonweighted) tardiness problems. Furthermore, this method is shown to be a special case of a general "priority-based" lead time estimation method developed in the thesis.[3] Another family of (parametrized) priority rules, originally derived for the single machine case by Morton and Rachamadugu [61, 62], uses the opportunity cost of the job not being started next as the priority index. This rule, the Apparent Urgency or AU rule, uses the expected waiting times on the subsequent machines in setting the local operation due dates. Three kinds of forward enhancements of the basic AU rule have been studied: adjusting a local look-ahead, more accurate lead time estimation on the subsequent machines, and the probing of the relative priority on the next machine. First, the AU priority index has an exponential look-ahead, i.e., priority discounting with increasing slack, scaled by the average processing time. A constant length of the look-ahead can be determined for all problems, but better performance can be achieved by adjusting the look-ahead according to the anticipated tardiness of the load. Performance feedback can be improved by adjusting the look-ahead period for each job according to its lateness in a schedule. An iterative procedure is revised for the look-ahead adaptation and tested in the single machine case. Second, the estimated waiting times on the subsequent machines, or the lead time of the job, are used to determine the slack and normative operation due dates for a job. Two kinds of methods using indirect load information have been revised:

1. Lead time estimation: The approximation of the expected waiting time of a job based on its attributes and the average level of load in the shop, or some other indirect information.

2. Lead time iteration: The smoothing of realized waiting time of a job in successive simulations with a dispathcing rule. In an iterative process, the waiting times yielding the lowest cost of a partial schedule are used in the final dispatching.

The performance feedback can be obtained through iteration of longer periods to find the parameter values or decision variable values which yield the best objective function values.[4] The

---

[2] See chapters 4 and 5 for the details of modifying the CoverT rules for the weighted tardiness problem.

[3] See chapter 5. for the discussion of lead time estimation in a dynamic job shop.

[4] The performance feedback can be on the level of an individual job, as in the look-ahead adaptation in chapter 2, or on the level of the full load as in the lead time iteration in chapters 4 and 5. The adjustment of a job's parameters based on the performance of the job in a schedule appears to be as efficient as a pairwise interchange routine. In larger problems, the simultaneous local adjustments are not as likely to converge in global performance.

efficient length of the look-ahead and the lead time estimates constitute indirect load information used in the priority index. Direct state information, such as the number of jobs in a queue, can be used to adjust periodically the priority setting parameters among different machines. This adjustment of the look-ahead and waiting time estimation parameters does not improve the performance compared to the best constant values of these parameters, but confirms the validity of the parameter setting rules for the constant rules in a given load conditions. Third kind of direct state information, such as the average urgency of the jobs, can be obtained through probing of the anticipated load on the machines using projected AU indexes. No significant improvement of the weighted tardiness criterion could be achieved, however, confirming the earlier findings with the Dynamic Composite Rule in averge tardiness scheduling [17].

7.1.3 Application of State Dependent Priority Rules

The results of the large scale experiments in static flow shops (chapter 4) and in dynamic job shops (chapters 5 and 6) have shown that the AU rule and the modified CoverT rule are very effective in minimizing weighted tardiness penalties and the number of tardy jobs. Both rules maintain also low levels of inventory. Since the AU rule is based on the local minimization of the marginal tardiness cost, it works best for decomposable problems in lightly loaded shops. CoverT is designed to minimize the average contribution to weighted tardiness costs, having an advantage in extremely congested shops. However, the performance of both rules is sensitive to the information used in the priority index. Four levels of information requirements can be distinguished, indicating the relevant trade-offs for scheduling system design. These relevant levels of state dependence of the AU and CoverT rules are:

Constant rules.

A simple system would can have a constant look-ahead parameter κ and waiting time estimation parameter β, without any global state information. The setting of appropriate parameter values for extended periods is rather easy on the basis of an estimate of the shop utilization. The performance of constant rules can, however, vary 10% and more depending on the overall load conditions. The rules are easy to implement in a decentralized system. The new priority based waiting time estimates, which use the distribution of the weights and processing times of the jobs, are robust enough to be applied even without frequent updating of the indirect load information.

Lead time iteration.

The parameters κ and β[5] are fixed but the waiting time estimates are improved through centralized iteration. This method was the best in the average performance for the simulation problems, moderating also the adverse effects of an inappropriate parameter selection. The new priority based waiting time estimates can be used to initialize the lead time iteration to guarantee high performance. The implementation of the lead time iteration is computationally manageable since the centralized simulation is straightforward and the performance feedback is not needed very frequently. The robustness against changes and errors in problem data has not been tested.

---

[5] β is needed for initial lead time estimation.

Load anticipation.

The values of κ and β parameters are directly dependent of some state indicator, for example the average critical queue length in the shop. The look-ahead and waiting time estimation functions are intuitive and the state estimates are based on simulated load conditions. The system can work in dispatching mode most of the time since the smoothed queue length estimates have to be updated only periodically eliminating the need for on-line state estimation. However, these rules provide no significant improvement in the weighted tardiness performance. The estimation of the queue lengths, once installed, allows the use of several refinements of the AU priority index function which require the identification of bottleneck machines and the machine-specific adjustment of the parameter values.

Urgency anticipation.

The coordination of the priority assignment according to the anticipated relative priority of the job on the subsequent machine requires rather frequent updating of the projected average urgency values or the queue lengths of the machines. The improvement of rule performance is not consistent, and the sensitivity of the probing procedure to the updating period and changes in problem data should be tested before implementation would be warranted.

The implementation of several state dependent enhancements was tested with the "self-adjusting" rule with marginal results. However, combining the state estimated look-ahead and priority probing with the lead time iteration procedure might provide enough improvement to warrant the testing. Another area of further testing is the setting of forecasting and planning horizons for performance feedback. The most appropriate lengths of these horizons depend on the congestion of the shop (longer horizons are needed in more congested shops) and on the possible forecasting errors in scheduling data.

These methods incorporate load indicators and managerial expertise into a scheduling support system. They can be compared to the previous methods reviewed in Introduction (heuristic search, estimation of decision rules, beam search or CPM dispatching with constraint driven state evaluation). Most centralized systems are inefficient in rapidly changing environments. Some are more decentralized but lack sufficient coordination of the local rules through resource pricing or global allocation rules. Managerial support is solicited in one of two ways: either the manager is given a fully designed, parametrized system and he has to learn to use it, or the manager serves as the source of different situational rules, state evaluations and constraints to be implemented in the system.[6] In both cases, resulting systems are often application dependent and difficult to transfer to different shops. Too often the amount data generation and mental adjustment needed to initialize and learn the system has had counterproductive impact on the user organization. We solve the information extraction problem by letting the managers first define the scheduling task in terms of costs, technical constraints and local decision premises. We provide tangible rules that can accommodate the relevant information, starting with simple and general rules and then refining the approach for additional

[6]The "Take (as is) and learn to use" is the conventional approach in operations research, whereas the "Extract, use and modify" is the approach adopted in the A.I. work on knowledge based systems.

constraints and performance. The marginal benefits and costs of the dispatching decisions, used as the priority index, depend on the future status of the shop which has been the weak part of the previous approaches. We let the computer analyze the anticipated future status of the shop through iterative application of the dispatching rule. The load indicators realized in one iteration are then used as the estimates in the next one. The effectiveness of the priority rules, the state indicators and the coordination mechanisms is tested through large comparative simulation studies in statistically specified load conditions.

7.2 Directions for Future Research

The results obtained in developing coordinated state dependent priority rules suggest several extensions and new applications of our approach. First, the Apparent Urgency rules can be generalized to consider explicitly several cost components, such as tardiness cost, inventory holding cost, earliness cost, and rush shipping costs of late orders. This leads to a composite priority index approach that can also deal with several resource constraints. Second, in dynamic inventory problems with finite production rates and a machine constraint, the stockout costs can be interpreted as quadratic tardiness cost function. Item availabilities can be used as slack estimates to derive efficient dispatching policy for the separately determined production lots. Third, the use of priority rules in a service system can lead to incentive-problems. If the information needed for the priority assignment is not publicly available, the user of a service system might get a strategic advantage from overstating his (unobservable) delay penalty. This possibility should be eliminated by an incentive-compatible pricing mechanism. Finally, the approach of state dependent priority indexes can be generalized in search systems requiring more extensive state evaluation, for example in a knowledge-based scheduling "expert" system. In the following, we explore these potential research topics in more detail concerning the methods of analysis and the expected results.

7.2.1 Composite Priority Indexes

The priority indexes studied above consist of the ratio of the expected marginal benefit (cost avoided) to the marginal cost of of implementing the decision (the opportunity cost of the machine). The opportunity cost of processing a job is proportional to its processing time and the opportunity cost of the machine has been normalized to one since we have considered only one resource. This basic approach can be generalized by introducing a composite priority index of the form:

$$\text{Composite Priority Index} = \Sigma_i MC_{ji} / \Sigma_r a_{rj} \lambda_r p_j, \tag{7.1}$$

where MC_{ji} is the estimate of the marginal benefit of processing job j immediately, in terms of avoiding cost component i. The marginal cost of resource r is its opportunity cost λ_r for the fraction used by job j, a_{rj}, and the processing time of job j, p_j.

The benefit side in Equation (7.1) above has a term for each cost component of the problem.

Examples of typical cost components and their dynamic approximations are:

1. Tardiness penalty: the simple exponential look-ahead of the Apparent Urgency rule would be the first approximation.

2. Inventory-holding cost: before the job has been started, the holding cost can be avoided by delaying the start of the first operation. Thus, for an extremely slack job, the marginal benefit should be reduced by the holding-cost rate. When the job becomes critical, however, the holding cost could be increased linearly from this negative value to enforce a timely start of the job. After the job has been started, the the holding cost should be added to the potential marginal benefit.

3. Rush-shipping cost: there is a fixed penalty s_j for shipping job j late. An approximation of the marginal cost rate for avoiding tardiness of the last operation can be given as s_j/p_c, where p_c is the estimate of the processing time of a competing job. When a job is projected to be tardy, this benefit would be reduced to zero. In order to avoid too early reaction, the full rate s_j/p_c should be reduced by a fraction for each additional operation remaining and for the global slack of the job analogously to the CoverT rule.

4. Earliness penalty: an explicit penalty for early completion over and above the work-in-process inventory-holding cost, such as a special storage cost.[7] For more detailed analysis of efficient estimation of the marginal cost rates, see [59].

Morton [54] has discussed the estimation of the opportunity costs of the resources. We could solve an aggregate version of the machine loading problem with a method which yields the dual prices. The dual prices are, however, usually quite unstable and thus problematic in the detailed scheduling application. More practical approach in the simulation context would be to use average resource utilization rents, based either on the historical values or on the anticipated usage as in chapter 6.

The testing of different approximations of the composite priority indexes implied above follows the experimental design used in the thesis. Starting with static single machine problems, each one of the cost components would first be studied separately as in the case of tardiness costs. Then different combinations of the cost components can be tested as a composite index. Similarly, additional resources can be added for experimenting with different shadow price estimation methods. After detailed studies in the simple environments, the most promising composite priority index methods could be tested in general dynamic job shop scheduling. We expect to discover simple superposition principles for combining individually tested results from the pilot studies in the full scale experiment. It should be noted, however, that the marginal costs and the opportunity costs are related by duality. Hence the consistency of their estimates should be a guiding factor in the development of the iterative adaptation procedures. Except for small problems, we cannot use the optimal solutions as benchmarks for the performance of the heuristics. The comparisons among different heuristics have to be facilitated in the experimantal design.

---

[7] Actually the inventory-holding cost should be changed from stage to stage during the manufacturing process.

7.2.2 The Lot Scheduling Problem

The research above has concentrated on scheduling in machine constrained open job shops. Inventory management in closed shops provides another example of a dynamic penalties (for inventory holding and stock-outs) that are convex (piecewise linear or quadratic) in lateness. More specifically, we are considering a single machine, multi-product lot sizing problem with dynamic demand. This problem has been solved in the case of stationary demand, allowing no stock-outs, and with the restriction to cyclical policies. We formulate a heuristic solution procedure that relies to certain extent on the principles of priority dispatching. The projected inventory-holding cost and stock-out costs, which depend on the rate of demand, will be used as the components of a composite priority index. The cost and duration of machine setup determines, in part, the desirable size of a manufacturing lot. Some of the costs of a setup are independent of the sequence and scheduling situation, such as labor and fixture costs, whereas the opportunity cost of the machine while it is down for the setup can vary over time.[8] If there is slack capacity, the capacity cost will be low allowing more frequent setups, smaller lot sizes and lower holding costs. The lot size will be constrained from below by the availability requirement through the production cycle. Once the lot size has been fixed, the production rate of the product in question determines the length of the machine reservation.

The use of dispatching approach requires a specification of the lots in terms of their (expected) processing times and due dates. The problem decomposes into two decisions:

1. When the machine becomes idle, what is the product to be produced next.

2. Given a product to produce, what is the appropriate lot size.

These levels interact since the dispatching decision depends on the possible lot sizes. This hierarchical system could be solved using iteration, solving first the problem with fixed lot sizes (e. g. EOQ or approximation of master cycle with static demand) to find to what extent the finite production rates limit the availability of some products. Then the lot sizes can be adjusted for the externalities caused by the other products. Given the tentative schedule, the dispatching priorities can be reassessed, possibly adjusting them for the contingencies later in the schedule.

Consider a case with zero setup cost and time for all but one product. An optimal policy would have larger than one unit production lots for all products since the longer production run for one product would violate the availability of the other products. The priority index in the quadratic problem could be derived analogously to the linear case in [59] smoothing the earliness and tardiness components properly. The increasing marginal tardiness cost would require a careful analysis of the local look-ahead. The expected tardiness cost of a job can be derived by integrating the marginal cost function over the duration of the competing lots. The optimal processing time of a job (lot size), and its lead time (inventory availability) are determined simultaneously. The production plan for one product can, in fact, be viewed as a multi-stage job with flexible operation processing times and due dates. Cyclical production policies with fixed lot sizes serve as benchmarks for different dynamic demand scenarios.

---

[8] The major factor in reducing the setup costs in Japanese production has been the preparation of the setup off-line, before the machine is stopped for setup, to eliminate the capacity cost.

7.2.3 Incentive-Compatible Priority Pricing Mechanisms

The application of a priority rule can create incentive problems since some jobs, typically long ones, have in the optimal schedule a longer wait than expected under the "fair" First Come-First Served discipline. The poor global performance of the FCFS rule in terms of weighted tardiness and inventory-holding costs indicates the need for motivating the users of a service center to comply with a more efficient priority assignment. The people involved might also have private information, for example the product manager alone has an accurate estimate of the cost of missing a delivery date promised to a client. A Pareto efficient priority rule would implement the globally optimal schedule that satisfies the jobs (product managers) via an equitable charge for a service on a shared facility delaying some other jobs. Moreover, this kind of pricing mechanism would be incentive-compatible if no user can gain from mispresenting his true cost parameter. We review first some results concerning priority pricing under lateness costs before addressing the problems of Pareto efficient incentive mechanisms for scheduling with due dates. Kleinrock [44] studied how a customer could determine his queue position by paying a fee, or a "bribe", to the service station manager when customers are served in the order of decreasing bribes. In a one-server poisson queue, the optimal[9] bribe is an increasing function of the individual waiting cost. Naor [56] proposed a toll levied for joining the queue to regulate the queue length and the externalities caused by the new entrant. The customers get a constant benefit from the service and they have homogeneous waiting costs. Without a toll a customer would balk when the estimated waiting time cost exceeds the benefit of the service whereas a toll for joining the queue, imposed by a central planner, induces the customers to balk when a "socially allowable" queue length has been exceeded. Adiri and Yechiali [2] considered several queues with different priorities and different tolls in one service center. They study customers' optimal queue (priority) selection rules and the related price setting procedures by monopolistic and non-monopoly service stations. Beja and Sid [8] look at the selection of optimal[10] priority class assignments when the customers have heterogenous waiting time costs as well as different service time requirements. However, the customers do not consider the state of the queues and their incentives when making the priority class selection. Marchand [48] derived a pricing formula for serving customers who have different wealth levels, delay costs and expected service times when a customer cannot have priority over another one. The optimal pricing rule was linear in the mean and variance of the service rate. Thus customers with longer service times are penalized more than proportionately. Levhari and Sheshinski [46] extend the pricing rule to a priority queue by determining the necessary conditions for a price system that induces the choice of the appropriate priority class by the customer.

Balachandran [6] addressed the question of the incentives of the customers to implement an efficient decision rule in decentralized queueing system. In his model, following [44], he allows customers to purchase priorities based on certain information about the queueing situation. He defines a stable payment policy which is optimal to use for any individual if it is adopted by the others as well. The remaining problem is, however, that while stable payment policies often are

---

[9] Here optimality is defined in terms of minimum expected total cost (bribe plus waiting time cost) subject to an average bribe constraint.

[10] The optimality of the system is determined by the expected waiting costs in the queue per unit of operating time.

easy to determine, they quite seldom yield globally optimal performance.[11] Dolan [19] introduced an incentive compatible taxation mechanism for a centrally run service facility. In a dynamic case, the mechanism charges an entrant a priority price that is equal to the marginal delay cost imposed on others. This price induces the user to reveal his true delay cost when maximizing his individual welfare. Hence the customers pay priority premiums which depend on the expected later arrivals into the queue. Dolan discusses also the incentive problems due to the possible redistribution of the collected taxes back to the customers.

Most of the previous work has addressed the efficiency problem, whereas the individual rationality of the users is not always assumed. Only Dolan solves the self-selection problem which leads to a incentive-compatible pricing mechanism. We want to extend his results. First, his taxation scheme for the dynamic lateness problem accumulates premiums to the manager of the facility. The same charges could be redistributed through a transfer-pricing mechanism. Each job passing another one in the line would pay an amount no less than the additional waiting cost directly to the other job. The research question is, does this procedure work when the jobs know only their own delay penalties. The possible gains from lying depend on the distribution of the delay penalties, the status of the machine queue, and the expected arrivals to the queue. We attempt to find out the robustness of the simple transfer-pricing mechanism against systematic deviations from truth telling through simulation experiments. Additional state dependent charges might be needed to prevent individual gain from overstating the delay penalty.

Second, the transfer-pricing mechanism, even if efficient in in the weighted lateness case, would break down with due dates. The changes of the costs delay would require "AU-like" look-ahead for finding a globally efficient schedule. We propose to study dynamic bidding mechanisms, administered according to well-known auction procedures. The complexity of the strategy space suggests the analysis of small problems with just a few jobs to find some good candidates for general bidding rules. Obviously a job would not bid more than the minimum of the expected sum of waiting time and bid costs later on. Thus the bidding rules are sensitive to the assumptions concerning the status information available either publicly or to the jobs individually. The role of the station manager is to provide approximate load information to guide the customers' expectation in deciding on their bidding strategy. Furthermore, only those jobs which are delayed beyond their due dates need compensation for the delay penalty. The design of a side payment scheme to achieve this provides another interesting problem. The potential advantages of transfer-pricing and bidding mechanism in more complex multi-stage scheduling problems should be studied further. Another goal of an incentive-compatible pricing mechanism is to improve the estimation of the strategic tardiness costs which are usually unobservable. Repeated bidding for priority reveals implicitly the cost that a manager assign to the consequences of being late.

---

[11] A simple example of an optimal payment policy with homogeneous service time expectations and constant waiting costs is to pay nothing, no matter how many jobs are in the queue. However, this policy is obviously not stable since a customer joining a long queue would have an incentive to pay a positive bribe to avoid the waiting cost. An example of a stable rule is a strictly increasing payment with the length of the queue. This policy, however if followed by all customers, implemets the LIFO discipline that is known to be non-optimal.

7.2.4 Diagnostic Analysis for Scheduling Expert Systems

The design of most scheduling systems has evolved around a few analytical models of the scheduling problem. Different users often adapt their own thinking to the limited view presented by the system to maintain communication. An alternative scenario is offered by the development of several expert systems and a technology that facilitates the extraction of the inference rules and multiple representations of the problem domain, or the knowledge base embodied in the experience of the human schedulers [23, 24, 37]. The inherent complexity of scheduling and the computational burden of the main tools, rule-based languages and frame-like knowledge representation systems, presuppose the application in conceptually rich tasks, such as checking the consistency and completeness of the formulation of a scheduling problem, analysis of the feasibility and structure of a given schedule, and explanation of the trade-offs among decision alternatives, rather than in computationally intensive tasks such as schedule generation. Hence a *diagnostic analysis* of production-inventory systems with problem-driven data analysis, trouble-shooting and incremental scheduling, one of the underrated areas of research [33, 60, 69], appears to be well suited for an application of knowledge-based systems. The following topics provide our challenge for new heuristic approaches and their practical implementation.

The precedence relations among jobs and their operations are important in the job shops with product assembly and project managament. Usually jobs have several alternative routings through the shop [30, 54], and the setup costs depend on the sequence of the jobs [7]. The allocation of other resources besides machines, such as labor and tools, would require an automatic link between the aggregate production planning and the detailed scheduling models. A hierarchical planning framework, similar to the concept in closed job shops [32, 34], would allow aggregate planning decisions as preconditions of scheduling, and a detailed schedule as an input to the loading of capacity. There are empirical studies in closed shop environment [9], and some analytical approaches have been tested for the coupling of capacity planning and scheduling decisions in single-machine open shop [26, 27]. We will simplify the analysis on the detailed scheduling level by using state dependent dispatching rules to generate an approximate timing of the jobs, given their tardiness penalties and initial estimates of the opportunity costs of the resources. Then the capacity planning could assign the jobs to certain periods. The expert system should also be capable of scheduling on different levels of aggregation to allow a hierarchical analysis of the problem.

Equipped with the hierarchical, multi-resource model of the scheduling environment, the expert system would be capable of useful diagnostic analysis. The data of a trouble-shooting problem would include the records of the operations in question, the specifiaction of any existing rules, and the measures of performance. In addition, the system would require a history of the implemented decisions, including possible previous plans and changes of operational policies. The system could check the consistency of past scheduling decisions by using the priority index of a state dependent rule. Given some symptoms of an unusual situation, the system would identify the possible problems involved. The task of the system would be to find out the possible causes of the problems, for example if there has been errors in the data, if poor decisions have been made, or if the decisions have not been implemented properly. The same diagnostic approach would suffice to examine future plans for consistency, sensitivity to changes in data, sensitivity to implementation errors (systematic bias or random errors), and to ensure contingency plans for flexibility.

The diagnostic analysis of scheduling problems should orient to the most appropriate level of abstraction, indicating the need for hierarchical knowledge representation [24, 68]. The hierarchy would encompass different aggregations of resources and schedules and a focusing capability to concentrate on most relevant issues, such as the scheduling of bottleneck facilities and the availability of critical parts. Most interesting extension to the existing scheduling expert systems [24, 37] should be, however, the capability of representing the diagnosis problem, in terms of some (tentative) solutions, as several internally cosistent mappings between the primal (measurable) and dual (opportunity cost) dimensions.

8. References

[1] Adam, N. R., and J. Surkis.
Priority Update Intervals and Anomalies in Dynamic Ratio Type Job Shop Scheduling Rules.
Management Science 26(12):1227-1237, December, 1980.

[2] Adiri, I. and U. Yechiali.
Optimal Priority-Purchasing and Pricing Decisions in Nonmonopoly and Monopoly Queues.
Operations Research 22(5):1051-1066, September-October, 1974.

[3] Baker, K. R.
Introduction to Sequencing and Scheduling.
John Wiley & Sons, Inc., New York, 1974.

[4] Baker, K. R.
A Dynamic Priority Rule for Scheduling Against Due Dates.
In *IMS-ORSA Conference in Houston.* IMS-ORSA, Houston, October, 1981.

[5] Baker, K. R., and Martin, J. B.
An Experimental Comparison of Solution Algorithms for the Single Machine Tardiness Problem .
Naval Research Logistics Quarterly 21(1):187-199, January, 1974.

[6] Balachandran, K. R.
Purchasing Priorities in Queues.
Management Science 18(5, Part I):319-326, January, 1972.

[7] Barnes, J. W. and L. K. Vanston.
Scheduling Jobs with Linear Delay Penalties and Sequence Dependent Setup Costs.
Operations Research 29(1):146-160, January-February, 1981.

[8] Beja, Avraham and Esther Sid.
Optimal Priority Assignment with Heterogeneous Waiting Costs.
Operations Research 23(1):107-117, January-February, 1975.

[9] Berry, W. L., and V. Rao.
Critical Ratio Scheduling: An Experimental Analysis.
Management Science 22(2):192-201, October, 1975.

[10] Bertrand, J. W. M.
The Effect of Workload Dependent Due-Dates on Job Shop Performance.
Management Science 29(7):799 - 816, July, 1983.

[11] Bestwick, P.F., and Hastings, N.A.J.
A new bound for machine scheduling.
Operational Research Quarterly 27(2):479-487, 1976.

[12] Bonney, M.C., and Grundy, S.W.
Solutions to the constrained flowshop sequencing problem.
Operational Research Quarterly 27(4):869-883, 1976.

[13] Bowman, E. J.
Consistency and Optimality in Managerial Decisions.
Management Science 9(2):310-322, October, 1962.

[14] Brooks, G.H., and White, C.R.
An algorithm for finding optimal or near optimal solutions to the production scheduling problem.
Journal of Industrial Engineering 16:34, 1965.

[15] Buffa, E. S., and J. G. Miller.
Production-Inventory Systems: Planning and Control.
Richard Irwin Inc., Homewood, Ill., 1979.
3rd Edition.

[16] Carroll, D. C.
Heuristic Sequencing of Jobs with Single and Multiple Components.
PhD thesis, Sloan School of Management, M. I. T., 1965.

[17] Conway, R. W., Maxwell, W. L. and Miller, L. W.
Theory of Scheduling.
Addison-Wesley Inc., Reading, Mass., 1967.

[18] Dannenbring, D. G.
An Evaluation of Flow Shop Sequencing Heuristics.
Management Science 23(11):1174-1182, July, 1977.

[19] Dolan, Robert J.
Incentive mechanisms for priority queuing problems.
Bell Journal of Economics 9(2):421-436, Autumn, 1978.

[20] Emery, James C.
Job Shop Scheduling by Means of Simulation and Optimum-Seeking Search.
In *Proceedings of the Third Conference on Application of Simulation, Los angeles.*
ACM/AIIE/IEEE/SHARE/SCi/TIMS, Los Angeles, December, 1969.

[21] Emmons, H.
One-Machine Sequencing to Minimize Certain Functions of Job Tardiness.
Operations Research 17(4):701-715, July-August, 1969.

[22] Fisher, Marshall L.
Optimal Solution of Scheduling Problems Using Lagrange Multipliers: Part I.
Operations Research 21(5):1114-1127, September-October, 1973.

[23] Fox, Mark S.
The Intelligent Management System: An Overview.
Technical Report, The Robotics Institute, Carnegie-Mellon University, August, 1981.

[24] Fox, Mark S.
Job Shop Scheduling: An Investigation in Constraint-Directed Reasoning.
PhD thesis, Carnegie-Mellon University, April, 1983.

[25] Fox, P. D. and C. H. Kriebel.
An Empirical Study of Scheduling Decision Behavior.
The Journal of Industrial Engineering XVIII(6):354-360, June, 1967.

[26] Gelders, L. and P. R. Kleindorfer.
Coordinating Aggregate and Detailed Scheduling Decisions in the One-Machine Job Shop: Part I - Theory.
Operations Research 22(1):46-60, January-February, 1974.

[27] Gelders, L. and P. R. Kleindorfer.
Coordinating Aggregate and Detailed Scheduling Decisions in the One-Machine Job Shop: Part II - Computation and Structure.
Operations Research 23(2):312-324, March - April, 1975.

[28] Gere, W. S.
Heuristics in Job Shop Scheduling.
Management Science 13(2):167-190, October, 1966.

[29] Gilmore, P. C. and R. E. Gomory.
Sequencing a One-state Variable Machine: A Solvable Case of the Travelling Salesman Problem.
Operations Research 12:345-357, 1964.

[30] Glazebrook, K. D., and J. C. Gittins.
On Single-Machine Scheduling with Precedence Relations and Linear or Discounted Costs.
Operations Research 29(1):161-173, January-February, 1981.

[31] Goldratt, E. M.
Optimized Production Timetable: Revolutionary Program for Industry.
In *Proceedings of the American Production and Inventory Control Society*, pages 172-176. APICS, 1982.

[32] Graves, Stephen C.
The Introduction of Feedback into a Hierarchical Production Planning System.
Technical Report 177, Operations Research Center, Massachusetts Institute of Technology, 1980.

[33] Graves, Stephen C.
A Review of Production Scheduling.
Operations Research 29(4):646-675, July-August, 1981.

[34] Graves, Stephen C.
Lagrangian Techniques to Solve Hierarchical Production Planning Problems.
Management Science 28(3):269-274, March, 1982.

[35] Gupta, J.N.D.
The generalized n-job, m-machine scheduling problem.
Operations Research 8:173, 1971.

[36] Haley, Paul.
Adding Knowledge to the Critical Path Method.
In *14th Pittsburgh Conference on Simulation and Modeling*. University of Pittsburgh, Pittsburgh, Pa, 1983.

[37] Haley, P., J. Kowalski, J. McDermott and R. McWhorter.
PTRANS: A Rule-based Management Assistant.
Unpublished, Carnegie-Mellon University, May, 1983.

[38] Hitz, K.L.
Scheduling of Flexible Flowshops I.
Research Report Lids-R-879, Massachusetts Institute of Technology, March, 1979.
Report from the Laboratory for Information and Decision Systems.

[39] Hitz, K.L.
Scheduling of Flexible Flowshops II.
Research Report Lids-R-1049, Massachusetts Institute of Technology, October, 1980.
Report from the Laboratory for Information and Decision Systems.

[40] Holloway, C. A., and R. T. Nelson.
Job Shop Scheduling with Due Dates and Variable Processing Times.
Management Science 20(9):1264-1275, May, 1974.

[41] Johnson, S. M.
Optimal Two and Three-Stage Production Schedules with Set-up Times Included.
Naval Research Logistics Quarterly 1(1), March, 1954.

[42] Jones, C. H.
An Economic Evaluation of Job Shop Dispatching Rules.
Management Science 20(3):293-307, November, 1973.

[43] King, J.R., and Spachis, A.S.
Heuristics for flow-shop scheduling.
International Journal of Production Research 18(3):345-357, May, 1980.

[44] Kleinrock, Leonard.
Optimal Bribing for Queue Position.
Operations Research 15(2):304-318, March-April, 1967.

[45] Lenstra, J. K.
Sequencing by Enumerative Methods.
Mathematisch Centrum, Amsterdam, 1977.

[46] Levhari, D. and E. Sheshinski.
The Economics of Queues: A Brief Survey.
In *Economics and Uncertainty*, chapter 7, pages 195-212. North Holland, New York, 1974.

[47] Lomnicki, Z.A.
A branch-and-bound algorithm for the exact solution of the three machine scheduling problem.
Operational Research Quarterly 16:89, 1965.

[48] Marchand, M.
Priority pricing with application to time-shared computers.
In *AFIPS - Conference Proceedings*, pages 511-519. 1968.

[49] Maxwell, William.
On the Generality of the Equation $L = \lambda W$.
Operations Research 18(2):172-174, 1969.

[50] McMahon, G.B., and Burton, P.G.
Flow-shop scheduling with the branch and bound method.
Operations Research 15:473, 1967.

[51] Moore, J. M., and R. C. Wilson.
A Review of Simulation Research in Job Shop Scheduling.
Product Invent. Mgmt. 8:1-10, 1967.

[52] Morton, T. E., and B. G. Dharan.
Algoristics for Single Machine Sequencing with Precedence Constraints.
Management Science 24(10):1011-1020, June, 1978.

[53] Morton, T. E.
Forward Algorithms for Forward-Thinking Managers.
In *Applications of Management Science*, chapter 1, pages 1-55. JAI Press, Inc., 1981.

[54] Morton, Thomas E.
Activity Scheduling, Ch. 7 in Callisto, An Intelligent Project Management System.
Technical Report, Robotics Institute, Carnegie-Mellon University, November , 1983.

[55] Muth, J. F., and G. L. Thompson (editors).
Industrial Scheduling.
Prentice-Hall, Englewood Cliffs, N. J., 1963.

[56] Naor, P.
The Regulation of Queue Size by Levying Tolls.
Econometrica 37(1):15-24, January, 1969.

[57] Palmer, D.S.
Sequencing jobs through a multi-stage process in the minimum total time -- a quick method of obtaining a near optimum.
Operational Research Quarterly 16:101, 1965.

[58] Panwalker, S. S., and W. Iskander.
A Survey of Scheduling Rules.
Operations Research 25(1):45-61, January-February, 1977.

[59] Peng Si Ow.
Heuristic Knowledge and Search for Scheduling.
PhD thesis, Graduate School of Industrial Administration, Carnegie-Mellon University, April, 1984.

[60] Pounds, William F.
The Process of Problem Finding.
Industrial Management Review 11(1), Fall, 1969.

[61] Rachamadugu, R.V., and T.E. Morton.
Myopic Heuristics for the Single Machine Weighted Tardiness Problem.
Working Paper # 28-81-82, Graduate School of Industrial Administration, Carnegie-Mellon University, 1982.

[62] Rachamadugu, R. R.
Myopic Heuristics in Job Shop Scheduling.
PhD thesis, Carnegie-Mellon University, September, 1982.

[63] Rachamadugu, R. V., A.P.J. Vepsalainen and T.E. Morton.
Scheduling in Proportionate Flow Shops.
Working paper CMU-RI-TR-83-10, The Robotics Institute, Carnegie-Mellon University, 1982.

[64] Schild, A., and Fredman, I. J.
On Scheduling Tasks with Associated Linear Loss Functions.
Management Science 7:280-285, 1961.

[65] Shwimer, J.
On the N-job, One Machine Sequence-dependent Scheduling Problem with Tardiness Penalties: A Branch and Bound Solution.
Management Science 18 B:301-313, 1972.

[66] Vepsalainen, A. P., R. V. Rachamadugu, and T. E. Morton.
Lead Time Iteration in Flow Shop Scheduling.
Unpublished working paper, GSIA, Carnegie-Mellon University, August, 1982.

[67] Vepsalainen, A. P. J.
An Adaptive Heuristic for Weighted Tardiness Scheduling.
Unpublished working paper, GSIA, Carnegie-Mellon University, April, 1982.

[68] Vepsalainen, A. and M. Greenberg.
Knowledge Representation in a Project Scheduling System.
In *14th Pittsburgh Conference on Simulation and Modeling.* University of Pittsburgh, Pittsburgh, Pa, 1983.

[69] Wagner, Harvey M.
Research Portfolio for Inventory Management and Production Planning Systems.
Operations Research 28(3 Part I), May-June, 1980.

[70] Wiest, J. D.
A Heuristic Model for Scheduling Large Projects with Limited Resources.
Management Science 13 (6), February, 1967.

[71] Wismer, D.A.,
Solution of the flow-shop scheduling problem with no intermediate queues.
Operations Research 20:689, 1972.

State Dependent Priority Rules for Scheduling

Ari P. J. Vepsalainen

CMU-RI-TR-84-

5 July 1984

Graduate School of Industrial Administration
and
The Robotics Institute
Carnegie-Mellon University
Pittsburgh, Pa 15213

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

Copyright © 1984 Ari P. J. Vepsalainen

This research was sponsored, in part, by the Air Force Office of Scientific Research under contract F49620-82-K0017, and the Westinghouse Corporation.