Copyright is owned by the Author of the thesis. Permission is given for a copy to be downloaded by an individual for the purpose of research and private study only. The thesis may not be reproduced elsewhere without the permission of the Author.

AN INVESTIGATION OF THE DISPATCHING AND EXPEDITING RULES IN BUFFER MANAGEMENT

A thesis presented in partial fulfilment of the requirements for the degree of Master of Technology in Production Technology at Massey University

Togar Mangihut Simatupang

1997

Abstract

Buffer Management is a proactive way of controlling the flow of materials on a shop floor. For shops using the Drum-Buffer-Rope (DBR) scheduling system, information on the effectiveness of non-constraint resources can be captured by monitoring the buffer status. Practitioners use this information to initiate improvement efforts and to decide to expedite when some of the inevitable disruptions are likely to undermine shop performance. This study attempts to investigate three areas in Buffer Management: dispatching rules, expediting rules, and variance reduction. The selected dispatching rules are First-Come-First-Served (FCFS), Shortest Processing Time (SPT) and Minimum Slack Time (MINSLK). Both static and dynamic expediting rules are compared. Reduction in the coefficient of variance for processing times from 100% to 50% corresponds to the process of quality improvement. Mean protective capacity of non-constraint resources is varied to represent different levels of loading on the shop. Inventory and due date measures are used to appraise shop performance. Simulation results indicate that the FCFS dispatching rule is the method of choice if due date performance is important. The shop using the SPT dispatching rule produces lower cycle times. The dynamic expediting rule is only preferred in the shop using FCFS and when mean protective capacity is low. The reduction in processing time variability renders a dramatically improved shop performance.

Acknowledgements

I give thank to my Saviour and Lord Jesus Christ for His unfailing love in my life. I am also indebted to my chief supervisor Dr. Simon F. Hurley for his patience, guidance and friendship during the completion of this research. Many thanks are due to my second supervisor Dr. Adrian N. Evans for his advice and counselling. Both of them encouraged me a lot to emphasis clarity, necessity and integrity. I am also grateful to Prof. Don J. Barnes, who introduced me to the knowledge of quality systems, for his help in many ways. I would like to thank my colleagues at the Department of Production Technology especially to Jeff Foote for the valuable discussion regarding Buffer Management. I wish to thank the New Zealand Government, through the Ministry of Foreign Affairs and Trade, Overseas Development Assistance (ODA) Program, for sponsoring my study at Massey University. Finally, I would like to thank the Indonesian Government, through the Department of Industrial Engineering, Bandung Institute of Technology, for allowing me to pursue my study in New Zealand.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1. Background

Time as a strategic source of competitive advantage has become more significant in today's changeable competitive markets (Stalk, 1988). To outperform the competition, world class companies build the distinctive capabilities of managing time into production, new product development and distribution.

The marketing function aims to capture information about customers' expectations (Lockamy and Cox, 1994). This information then needs to be translated to a form suitable for the production function. Only companies that can utilise their resources effectively will achieve fast response times to customer demands. Lead time is simply the total time required to deliver products to the customers and has been identified by many companies as a strategic measure of competitive advantage (Stalk, 1988; Carter *et al.*, 1995; Tersine and Hummingbird, 1995).

The production function critically determines a company's ability to compete on time-based dimensions. The link between actions and decisions at an operational level and competitive position must be explicit if a company is going to survive. However, the inherent variability in the shop floor makes this process non-trivial. A shop floor is characterised by two phenomena, namely dependent events and statistical fluctuations (Goldratt and Cox, 1992). Failure to accommodate these phenomena leads to poor performance, such as long lead time, long queues, broken setups, lost and defective parts, late deliveries and large work-in process inventories (Umble and Srikanth, 1990).

One way to simplify the complexity of operations is to use the Drum-Buffer-Rope (DBR) technique as a material control mechanism on the shop floor (Graves *et al.*, 1995). DBR is a novel way of managing the flow of materials on the shop floor by focusing on a few capacity constraints (Goldratt and Cox, 1992). The central premise of DBR is that the resource with the lowest capacity determines the production rate of the entire manufacturing line. Orders are scheduled according to the time availability of the constraint resource and listed in the master production schedule. A time buffer is maintained in front of the constraint resource to keep the constraint or "drum" working effectively. This buffer is required because of the variable processing times and unavoidable disruptions introduced by the upstream resources. The pace of the non-constraint resources is synchronized by controlling the input of materials to the gating operations. The "rope" regulates the order release rate to be the same as the consumption rate of the drum, so undesirable work in process (WIP) inventories do not build up. Parts are processed in a First Come First Served (FCFS) order by non-constraint resources. In this manner, DBR can achieve high due date performance of the orders and at the same time maintain the low WIP inventory (Umble and Srikanth, 1990).

The success of DBR depends on the method used to ensure that the constraint resource is never starved of work. The first task is to eliminate the worst case variability at the constraint. Thus, the constraint can effectively process the parts according to the drum schedule. The second step uses information in the buffer. Severe disruptions at the upstream resources may cause expected parts to be missing from the buffer. Workers can monitor the buffer status and from there determine the cause of those upstream disruptions that endanger the output rate of the production line. Expediting parts at the non-constraint resources is suggested if the missing part has passed the critical point in the buffer. The use of buffer status to improve the shop floor performance is called Buffer Management.

The main advantage of DBR is its ability to stabilise the amount of WIP in a production facility and to protect the critical resource, thereby providing stable lead times. A further advantage of applying DBR and Buffer Management is the ability to help decision makers characterise the key strategic factors, as a basis for planning an appropriate course of action for operations improvements on the shop floor. These efforts enable practitioners to focus on ongoing improvement and to increase profit, as a result of fast response time to customers' needs (Goldratt and Cox, 1992). Therefore, the method of DBR and Buffer Management simplifies and improves the manufacturing shop and also highlights many opportunities to improve the shop. The next section addresses these opportunities.

1.2. Problem Statement

Under DBR, the methods managers use to load a shop is based on a relationship between resource capacity and queue time. Delivery promises are given according to the constraint's schedule. To protect the constraint from starving, due to inevitable processing disruptions at non-constraint resources, a buffer of work, measured in time, can be placed in front of the capacity constraint. If one of the upstream resources breaks down, its spare capacity, termed protective capacity can be used to rebuild the constraint's buffer, once it has been repaired.

Materials or parts are released to the shop to arrive at the constraint resource a buffer time before they are due to be processed. Materials move through the non-constraint resources in FCFS order. A resource is authorised to work as soon as possible with the earliest part that has arrived. Instead of FCFS, there is no requirement for sophisticated priority dispatching rule used to choose the part from a queue, since the WIP inventory is only allowed to build up in front of the constraint resource (Gardiner *et al.*, 1993). However, in the situation when the material release control is steady and the protective capacities vary, managers may need alternative priority dispatching rule in order to increase due date performance such as minimizing lateness and thereby lead time. Therefore, the opinion that it is sufficient to merely use simple dispatching rules with DBR needs to be verified.

Buffer Management regularly evaluates the ability of non-constraint resources to process parts through to the buffer. Buffer status data is used to ensure the performance of the drum, as well as maintaining timely product deliveries. Buffer Management reacts to expedite the missing part at the non-constraint resources only when it has passed a half time buffer (Schragenheim and Ronen, 1991). This half time buffer is called the critical point because a late part after this point potentially endangers the master schedule. This expediting rule refers to the static checking point at the time buffer, after Goldratt and Fox (1986). As an alternative, Hurley (1996) proposes a more proactive approach to determine an action to expedite a missing part, using a dynamic checking point. An expediting action is taken if the remaining production time needed by a part to arrive at the buffer is less than the time remaining of a part to arrive at the critical point. This delta time indicates a need of expediting to give priority for a late part processing through upstream resources, enabling it to arrive when it is needed by the constraint. Since the protective capacities of the non-constraint resources vary, it is important to know the effect of the different expediting rules on the shop performance. A comparison test would also indicate the effectiveness of different expediting rules.

If an action to expedite a part is taken, the primary dispatching rule is temporarily ignored. However, expediting should not be used to "cover up" problems in the process or at the expense of increased WIP inventory. Process improvements, such as reducing variance, help to ensure timely product deliveries. Hence there is a need to examine the effect of reducing variance on shop performance. Thus improvement efforts can be carried out to reduce variability of processing time, as well as the possibility to reduce WIP inventory. More stable processing times lead to reduced protective capacity and buffer sizes.

In summary, three problems in DBR and Buffer Management have been identified. First, the benefit of simple dispatching rules such as FCFS is under question. The second problem is the comparison of static checking point and dynamic checking point to determine an action for expediting. The last problem is to evaluate the importance of quality improvement in Buffer Management, such as reducing processing time variance. Implementing this quality improvement can reduce the frequency of expediting. Thus Buffer Management should fully protect the constraint schedule and guide quality improvement efforts, that have a positive impact on the organisational performance along time-based dimensions.

1.3. Outline of the Study

Considering the objectives of the study, three research questions can be identified that guide the application of simulation to provide the appropriate answers. The questions are stated as follows:

1. Can it be shown statistically that the effect of the FCFS dispatching rule under DBR on shop performance is superior to other selected dispatching rules, given various levels of mean protective capacities?
2. Is the effectiveness of dynamic checking point superior to the static checking point for determining when to expedite?

3. Does the reduction of variance on the processing times of the upstream resources positively impact shop performance?

1.4. Importance of the Study

The findings can be used to generate new knowledge about the strategy used to effectively implement DBR and Buffer Management. The interpretation of the results also provides some implications for the practice of DBR and Buffer Management. Little is known of the effect of dispatching rules, expediting rules, and variance reduction under DBR environments. The need for this extension to Buffer Management research has been identified by Hurley (1996).

Research results will assist managers in conducting ongoing improvement. Managers will know where improvement can be used to cut current lead times without having a detrimental effect on organisational performance. For instance, the efforts of expediting can be reduced through implementing variance reduction program. In addition, managers will know how to determine the operating policies their shops use to counteract fluctuations in protective capacity, such what dispatching and expediting rules should be used.

1.5. Organisation of the Study

This introductory chapter has outlined the background, the problem, the purpose and the significance of this study.

Chapter two presents the theoretical frameworks of this study. It contains an outline of the fundamental concepts of DBR and Buffer Management, previous research on DBR, and a critique of current research and a summary.

Chapter three describes the methodology of this study, including model formulation. Model formulation incorporates dispatching rules, expediting rules and variance reduction. Chapter four discusses the simulation results and analysis. Chapter five provides some recommendations for future study. Finally, Chapter six presents a summary of significant findings and conclusions.

Chapter 2
Subject Review

2.1. Introduction

The Drum-Buffer-Rope (DBR) scheduling technique is the Production Application of the Theory of Constraints (TOC). In terms of production scheduling, TOC is simply a way of thinking that forms the perception of the true state of things (Noreen *et al.*, 1995). It has developed through practice over time and has a pragmatic emphasis on acquiring knowledge about how to attain the desired results from practice.

Section 2.2 describes the DBR scheduling technique. DBR concentrates on the resource constraint, which determines the overall output rate of a shop. DBR can perform well in providing a fast delivery response and a low work in process (WIP) inventory. Through Buffer Management, DBR can also guide managers in the implementation of ongoing improvement. A review of Buffer Management is the subject of Section 2.3. Section 2.4 outlines the state of current research on DBR and Buffer Management. A classification of the research is used to aid in comprehension of the spectrum of studies presented, ranging from a theoretical perspective to practical orientation. Section 2.5 specifies recent research on expediting under Buffer Management.

A critique of current research, detailed in Section 2.6, gives some insights into how to solve the emerging problems of DBR and Buffer Management.

2.2. Drum-Buffer-Rope

The name drum-buffer-rope stems from a metaphor used by Goldratt and Cox (1984) to explain the reality of a manufacturing firm. Metaphors provide common language that contain a feature of resemblance between different contexts and a basis for communication between the people in a firm to create and share understanding (Morgan, 1980).

The metaphor used to describe DBR is a line of scouts or soldiers marching single file, as shown in Figure 2.1. The common purpose is to arrive at predetermined destination as a troop as soon as possible. Different scouts that march in the line have different speeds. Moreover, each scout walks with wide fluctuations of velocity and often with some disruptions. The slowest scout in the line governs the pace of the troop. This slowest scout should walk constantly and be responsible to beat the drum. In front of the slowest scout, a spare space is given as the buffer to ensure the slowest scout still can walk if any disruptions taken place in the upstream scouts. The scouts in front of the slowest scout can directly adjust their speed after they recover from disruptions. Since the leading scouts are faster than the slowest scout, a rope is tied between the lead scout and the slowest scout to prevent any unnecessary gaps between them. The lead scout only walks as fast as the slowest scout. The scouts behind the slowest scout can easily close the gap if there are any disruptions to the slowest scout. Finally, this line of scouts are able to arrive as a troop since they can keep walking at the same speed.

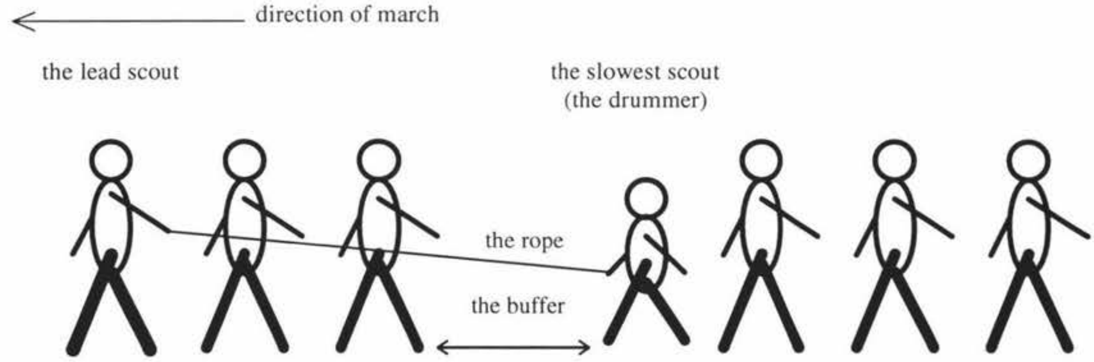


Figure 2.1. A metaphor of Drum-Buffer-Rope

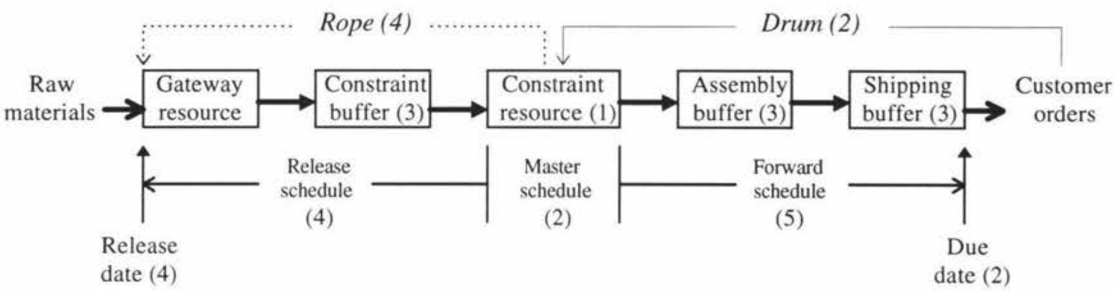
The DBR scout metaphor provides ideas about the relationship of capacity and queue time in the manufacturing firm required if it is to attain excellent due date performance and a low WIP inventory. The sequence of the troop refers to the routing of a product and the scouts are similar to the resources. Each resource exhibits statistical fluctuations and disruptions when it processes the parts. Resources in the firm have different capacities and the resource with the least capacity is called the bottleneck.

In order to keep the bottleneck running at all times, a time buffer is placed in front of the bottleneck. This is the gap between time when parts arrive in front of the bottleneck and when they are needed by the bottleneck. The rope is analogous to the control point which determines when to release the materials to the shop. Materials can only be released as fast as the rate of the bottleneck. In doing so the WIP inventory, as analogous to the gaps between the scouts, only accumulates in front of the bottleneck. The scout metaphor needs some extension in order to find the way it works in the manufacturing firm.

As defined above, the resource with the least relative capacity is called the bottleneck or constraint. This resource does not have sufficient capacity to fulfil demand. The constraint should be protected from any statistical fluctuations at the upstream resources. To avoid the constraint from starving, the time buffer is placed in front of the constraint. This buffer time is also used to protect throughput and is thereby called the protective buffer. In addition to constraint resources, assembly and shipping resources also need time buffers (Goldratt, 1990; Umble and Srikanth, 1990). Assembly buffers ensure all parts have arrived before they are required to be assembled. Shipping buffers are needed to protect product deliveries from the statistical fluctuations and disruptions of the prior resources and from demand fluctuations.

Other non-constraint resources are utilised to support the schedule of the constraint and to reduce unnecessary WIP inventory. As each non-constraint resource has extra capacity compared to the constraint, WIP inventory can build up beyond the desired level. To overcome this problem, the rate at which jobs are released to the gateway point must be restricted to the consumption rate at the constraint. A rope is tied from the constraint resource, which regulates the release of the correct quantity of jobs into the system to support customer orders. Thus all non-constraint resources only work at the pace dictated by the constraint. The rope technically represents material release that covers the time of processing jobs and the buffer time in front of the constraint resource. Non-constraint resources can recover from processing problems and rebuild the constraint's buffer because they have extra capacity. This extra capacity at the non-constraint resources is called protective capacity. Any other extra capacity, that is not needed as protective capacity is termed excess capacity and either can be sold or used to produce other products.

In the DBR environment, the schedule can be simplified to deal with three main decisions: the drum or the master production schedule, the time buffer, and the rope or order review and release (Goldratt and Fox, 1986). Once the constraint resource is identified, the master production schedule is developed to sequence jobs on the constraint, according to the received orders. Then, the release time for each job (the rope) can be back calculated by considering the processing times of the upstream resources and the desired length of the protective buffer. Figure 2.2 illustrates the summary of DBR application in production management.



The DBR steps:

1. Identify the constraint resource.
2. Determine the master production schedule for the constraint resource.
3. Place time buffers at constraint resource, assembly, and shipping.
4. Schedule release date.
5. Determine the forward schedule for upstream resources.

Figure 2.2. Drum-Buffer-Rope in production management

DBR succeeds in its attempt to accommodate reality in manufacturing systems, producing a better planned schedule in which the flow of jobs can be synchronised and a clear utilisation of resources to maximise throughput. However, problems always emerge when production planning is implemented in the real world. For instance because statistical variation exist, the actual rate of jobs arriving at the protective buffer must be less than the planned arrival rate. In reality, there is no buffer length that provides a 100% protection and this may lead to lost throughput, expediting, and a host of other consequences. The buffer status represents this imperfect situation by reflecting what has taken place at the upstream resources. Production control called Buffer Management is needed to correct any deviations from the production plan based on information of buffer status. The next section describes the main ideas of Buffer Management and how it controls the flow of jobs according to the constraint schedule.

2.3. Buffer Management

Buffer Management has three basic activities: to monitor the protective buffer in front of the constraint resource, to compare the actual versus the planned performance and to advise necessary actions for improvement (Schragenheim and Ronen, 1991). Data to improve a production line's performance is captured by close monitoring of the buffer content. Deviations between the planned and actual content of the buffer indicate that disruptions have taken place at the upstream resources. Buffer profile information can be used to discover the causes of the disruptions and to set the priority for focusing improvement efforts. In other words, BM provides a signal about the health of the system and a basis to cure the system. The planned protective buffer content can be depicted as a rectangle (Goldratt and Fox, 1986). It consists of a vertical axis that represents the number of hours that a particular job will require at the constraint resource and a horizontal axis that measures when (e.g., what day) these jobs are scheduled to be processed by the constraint resource. By portraying a buffer in this way, it is possible to know the number of jobs in the protective buffer, the sequence in which they are going to be processed and the number of hours of work for which the constraint resource is protected. Figure 2.3 shows an example of a visual display of the hypothetical constraint buffer.

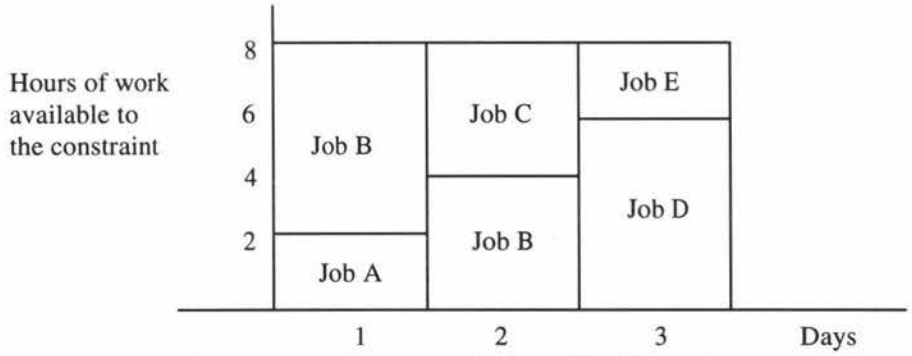
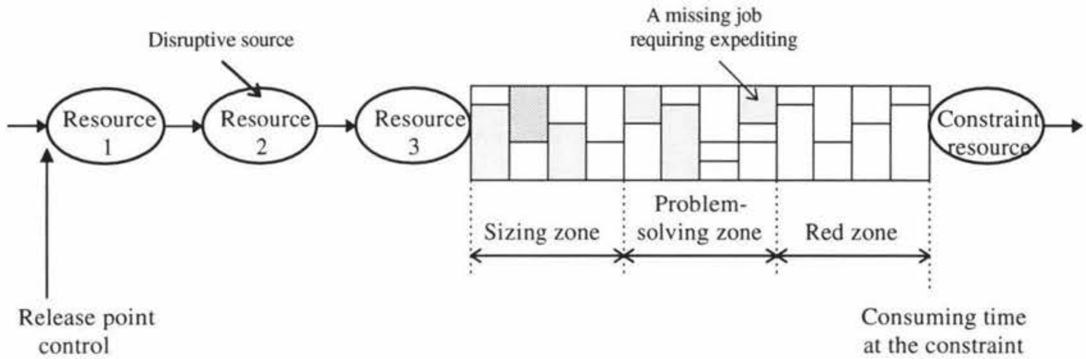


Figure 2.3. A visual display of buffer status

In practical situations the upstream resources never perform perfectly because of statistical fluctuations and disruptions. As a result, the actual buffer is rarely the same as the planned buffer. The missing jobs or parts can be described as holes in the protective buffer. The size of each hole indicates the amount of processing time required at the constraint resource. The position of a hole in the buffer also shows information on time remaining before the constraint resource is to process the part, the delay time of a part arriving at the buffer and the remaining time required by the prior resources to complete the missing part.

The buffer can be broken up into three zones (Schragenheim and Ronen, 1991) as shown in Figure 2.4. The jobs in the first third of the time buffer must always be present. This zone is named the red zone, because holes in this zone require immediate action. For instance, Schragenheim and Ronen (1991) use the delay time at the buffer to decide when to expedite a part; if the delay time has passed a checking point at half the time buffer, then the decision to expedite is taken. Some jobs are also missing in the second zone. This zone is called the problem solving zone. In this zone, the missing part can be found by tracing its routing and taking special action to eliminate the causes of delays. The hole should be noticed when it was still in zone two and serves as a focusing mechanism for expediting efforts. For example, Umble and Srikanth (1990) define the individual disruption factor for each missing part as the delay time multiplied by processing time at the constraint. The most significant sources of disruption in the production line can be determined by calculating a cumulative disruption factor for each resource. This information may be used to prioritise improvement efforts.

Ultimately, most of the jobs planned for the last third of the buffer are missing. This third zone is termed a sizing zone, since the majority of the holes appear in this zone. Any hole that occurs in this sizing zone can be disregarded as the upstream resources have enough time to replenish the hole by using their protective capacities. Data from monitoring the sizing zone provides a basis for adjusting order release and modifying the buffer length. If a job has arrived earlier than its schedule at the buffer, it indicates the materials are released prematurely and the gating point can be re-adjusted to follow the rate of the constraint schedule. On the other hand, if there are only a few jobs in zone three, the buffer is too short to display the disruptions and buffer length can be modified accordingly.



The Buffer Management steps:

1. Monitoring the holes.
2. Diagnosis to decide actions: disregarding, problem-solving, expediting.
3. Actions:
   1. Schedule adjustment:
      - Control the release point.
      - Correct the length of protective buffer.
   2. Implementing the improvement to eliminate the problems.

Figure 2.4. The mechanism of Buffer Management

The size of the protective buffer is a function of the amount of variation of the resources that feed the constraint resource. If the protective buffer is too large, there may be problems such as a long lead time and a high cost of holding work in process inventory. Problems associated with the protective buffer being too small include excessive expediting and the risk of starving the constraint resource and losing throughput. Therefore, the size of the protective buffer is usually determined subjectively, trading-off the cost of holding work-in process against the consequences of starving the constraint resource. This judgement includes consideration of the size and frequency of the delays that can occur because of normal fluctuations and disruptions in the prior resources. In addition, the protective capacities of the non-constraint resources also protect the resource constraint.

Buffer Management (BM) provides a new perspective in production management in terms of problem solving and expediting. Problems can be detected early before they affect the master production schedule and hence throughput. BM solves these problems in a proactive manner rather than a reactive one. Expediting is not viewed as the precept but the exemption. These advantages stem from the benefits of DBR that results in lower WIP and thereby reduced lead times. Recently, the growing amount of research does not merely highlight these advantages but also advances the concept of DBR and BM and details the success stories. The following section presents the current research into DBR and BM.

2.4. Current Research into Drum-Buffer-Rope and Buffer Management

Research into DBR and BM addresses the problem of how the production function of an organisation can respond effectively to the demands of the market. DBR attempts to provide a realistic production schedule focused on the constraints. Research into DBR can be classified into three streams: theoretical frameworks, simulation models, and case studies (Spencer and Cox, 1995b). The theoretical frameworks attempt to reveal how DBR and BM should work in the real world. It includes definitions, conceptual issues, conceptual models, normative suggestions, and other modifications of DBR and BM. Examples are the metaphor of DBR (Goldratt and Cox, 1984; Goldratt and Fox, 1986), performance measurement (Goldratt, 1990), static expediting (Schragenheim and Ronen, 1990), dynamic expediting (Hurley, 1996), and the Five-Step Focusing Process of how to implement DBR and BM (Goldratt, 1990; Schragenheim and Ronen, 1990). Simulation models test theory using computer models. Wu *et al.* (1994) and Guide (1995) use empirical data to test DBR and BM as alternatives to current production methods. Some researchers suggest specific models and conduct comparative simulation studies (Fry *et al.*, 1988, Cook, 1994, Schragenheim and Ronen, 1995). Case studies provide information on the assessment of current practices in DBR and BM in specific firms or industries. For example, Ashcroft (1989) reports the application of DBR and BM in a small furniture manufacturing firm. Table 2.1 depicts the selected references that illustrate the ongoing research into DBR and BM.

Table 2.1. Selected references discussing DBR and BM

References	***Research streams***	***Focus or Findings***
1984 Goldratt & Cox	T	• The basic concept of Drum-Buffer-Rope
1986 Goldratt & Fox	T	• Sequencing rule on the constraint schedule using early due date (EDD), simple dispatching rule: FCFS • Static buffering and static expediting rule
1987 Fry *et al.*	T, S	• Bottleneck position
1988 Goldratt	T	• Conceptual framework for computerising the DBR technique
1989 Ashcroft	C	• The constraint scheduling in a small firm
1990 Goldratt	T	• Performance measurement and the use of Five-Step Focusing Process • Advanced DBR and BM such as Rods & Dynamic Buffering
Ramsay *et al.*	S	• Squeeze shop floor with bottleneck capacity, buffer inventory, input control, and FCFS
Schragenheim & Ronen	S	• General overview of DBR and estimation of time buffer
1991 Fry *et al.*	C	• DBR to control lead time
Reimer	C	• General overview of DBR and BM
Schragenheim & Ronen	S	• General overview of BM and static expediting
Spencer	T	• General overview of DBR
1992 Colvenaer *et al.*	C	• Application of DBR with a strict FCFS
1993 Gardiner *et al.*	T	• The impact of DBR and BM on production management
1994 Atwater & Chakravorty	S	• Protective capacity and variance reduction
Cook	S	• A flow shop with FCFS, WIP inventory, and static expediting
Radovilsky	S	• Estimation of buffer size
Schragenheim *et al.*	T	• DBR and BM in Process Flow Industry
Wu *et al.*	C, S	• Application of DBR in a furniture manufacturing
1995 Chakravorty & Sessum	T	• Set-up reduction
Guide	C, S	• DBR application, estimation of time buffer, static expediting, and dispatching rules: FCFS, EDD
Schragenheim & Ronen	S	• Dealing with conflict in expediting
Spencer & Cox (1995a)	T, S	• Master Production Scheduling in DBR
1996 Chakravorty & Atwater	S	• A flow shop with FCFS, Line Variability, Down Time, and WIP inventory
Hurley	T	• Dynamic expediting

Legend: T = theoretical and conceptual framework discussing how the system should work.
S = simulation model presenting results and comparison.
C = case study describing how the system operates in a specific firm.

2.5. Expediting under Buffer Management

A research of Buffer Management that remains to be fully understood is expediting. Hurley (1996) suggests an alternative expediting rule to that Goldratt and Fox (1986). Somewhere between the first zone and the middle zone of a buffer lies the critical point at which the missing jobs are not acceptable. Thus without loosing generality, the protective buffer can be said to consist of two parts: the critical and the non-critical regions. At the critical point, the jobs that become lost or delayed often force costly expediting. The decision of when to expedite depends on the expediting rule used. Expediting rules show the degree to which a job's processing can be speeded up through the upstream resources without jeopardising throughput. Goldratt and Fox (1986) suggest an expediting rule in which a job must be expedited if it has not arrived at the critical point, the border between the non-critical region and the critical region. If the time length of the critical region is too short, the constraint resource will be impacted by the absence of the job. The magnitude of the damage depends on the processing time of the missing job on the constraint resource (Goldratt and Fox, 1986). To avoid this situation, usually the critical region should be long enough to absorb the sum of expediting times to deliver the job on time when it is consumed by the constraint.

Instead of checking at the critical point, Hurley (1996) suggests the use of dynamic checking by continuously monitoring the discrepancy between planned and actual processing time before the job arrives at the protective buffer zone. A missing job in the non-critical region can be anticipated by monitoring and comparing two variables. The first variable is the planned time remaining before a job arrives at the protective buffer and is set equal to the planned backward schedule. The second variable is the actual processing time remaining before the job arrives at the protective buffer. If there are no disruptions in the system, the planned time remaining equals the required processing time and the job can arrive at the protective buffer as planned. Both those time variables start and finish at the same time. However, the actual processing time remaining will always be greater than the planned time remaining due to non-instant availability of the resource or statistical variation (Goldratt, 1990). If the sum of the planned time remaining and the non-critical region is more than the actual time remaining, expediting is unnecessary. Expediting has to be taken if the difference equals zero or is negative, otherwise the job will arrive in the critical region and hence jeopardise throughput. Hurley (1996) terms this discrepancy as the slack time. This slack time represents the difference between the time remaining before the job is expected to arrive in the critical region and the amount of processing remaining before the job actually arrives at the protective buffer. This second buffering method is more proactive in the protection of throughput, in the way in which the critical buffer region is used. In the static expediting rule, the critical region is used to absorb the processing time still left to complete the part. In the dynamic expediting rule, the critical region is used to overcome the variability of the processing time up to the constraint buffer. Consequently, the dynamic expediting rule gives a shorter critical region than the first buffering style. Continually reducing buffer size means reducing manufacturing cycle time, which may translate into decreased quoted lead time (Gardiner *et al.*, 1993).

2.6. A Critique on Current Research

Buffer Management is the focus of most attention in current research because it helps practitioners execute planning in a practical and effective manner. One dispatching rule that has been widely accepted as a norm in BM is First-Come-First-Served (FCFS). A resource is authorised to work with the earliest part in the queue. Since non-constraint resources have a few parts in the queue, BM only needs dispatching rules based on local performance such as throughput-dollar-days and inventory-dollar-days (Goldratt, 1990; Gardiner *et al.*, 1993). This is the combination of the FCFS and expediting rules. When a part is identified as being expected to arrive late at the resource constraint, then the non-constraint resources is authorised to work with this part. Melinyk *et al.* (1986) found that the dispatching rules have a little impact on shop performance. However, there is no study to confirm that FCFS is superior to other dispatching rules under a variety of job conditions, such as different levels of mean protective capacity of non-constraint resources. Therefore, this study aims to test the hypothesis that FCFS is superior to other popular dispatching rules.

Expediting rules also have a little attention from researchers. Recently, Hurley (1996) suggests a dynamic checking point method to identify when to expedite a job.

However, this dynamic expediting rule has yet to be tested. This study attempts to evaluate the dynamic expediting rule in a simulated job shop.

Buffer Management encourages practitioners to conduct continuous improvement in the shop floor, such as variance reduction, defect reduction, maximised equipment utilisation, reducing lead time, and disruptions elimination. Since expediting is an expensive activity, continuous improvement is worthwhile to reduce the amount of expediting required. Hence, this study will demonstrate the effect of continuous improvement on the shop performance in terms of reducing processing time variance.

2.7. Summary

The DBR scheduling technique has been shown to be a good solution to the scheduling problem, providing excellent due date performance while at the same time minimising WIP inventories.

Three open research areas have been identified: the untested conjectures in the simple dispatching rule of DBR, the theoretical dynamic expediting rules (Hurley, 1996), and the improvements possible by reducing the variance of processing times. These problems need verification and testing using a simulation model.

The next chapter deals with the research method of dispatching rules, expediting rules, and variance reduction. It covers a basic model, so the conjectures can be evaluated through simulation.

Chapter 3
Research Methodology

3.1. Introduction

This chapter outlines research methodology to investigate the impact of operation decisions on the shop floor. The hypothetical dynamic job shop is used to test the impact of the various options of decisions on shop performance.The next section describes the experimental scenario that contains three decisions needed for confirmation and testing. Section 3.3 presents the model used in this study. Measures of performance are outlined in Section 3.4. A simulation model that emulates the real shop is presented in the subsequent section. Section 3.6 describes the experimental design used in this study. Finally, Section 3.7 highlights important issues pertaining data analysis.

3.2. Experimental Scenario

Experiments are carried out based on three scenarios, namely: confirmation of dispatching rules, testing of expediting rules, and examination of the effect of variance reduction. In the first experiment, alternative dispatching rules would be selected based on the simplicity and practicality in the real shop. The second scenario serves as the main experiment in this study in which two buffer monitoring styles are compared. The first buffer style is from Goldratt and Fox (1986) and the second buffer style is proposed by Hurley (1996). Variance reduction involves the effect of minimising the variability of processing time on shop performance. Since this research is interested in Buffer Management, a job shop is designed such that the final resource can serve as the resource constraint. This resource constraint is also an assembly point in the shop. Figure 3.1 shows the example of a simplified production flow through some of the resources. Other products can be generated randomly with the different patterns. This typical job shop only absorbs the statistical fluctuations of the upstream resources with the buffer in front of the last resource. The variation of the resource constraint is negligible because this study emphasises the behaviour of the upstream non-constraint resources. Thus the influence of variation on the shop performance can be observed during the simulation under predetermined scenarios. The following sub-sections discusses the elements of experimental models.

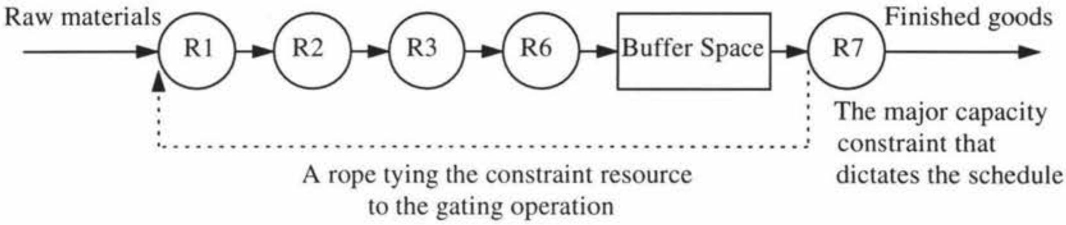


Figure 3.1. An example of the routing for one product

3.2.1. Dispatching rules

The selection method to choose the next job to be processed from a queue of jobs waiting to be processed at a machine is termed a dispatching rule. Dispatching rules are commonly used to control work-in-process and to improve delivery time performance (Blackstone *et al.*, 1982).

First-Come-First-Served (FCFS) is the dispatching rule typically used in the drum-buffer rope environment (Goldratt and Fox, 1986). Under its operation, FCFS gives the highest priority to job that arrives at a queue first. It is also a very simple system to use in a practical manufacturing environment. Ashton and Cook (1989) advocates that FCFS is a suitable dispatching rule to simplify job shop operation.

The second dispatching rule, the minimum slack time rule (MINSLK) comes from considering the time discrepancy between planned and actual processing time before a job arrives in front of a resource. The job with the minimum slack time should be the next to be processed.

Shortest processing time (SPT) gives a priority to a job in queue with the shortest imminent processing time to be processed next at a resource. SPT is primarily used to control tardiness and the flow time. The flow time is the amount of time job i spends in the system. The flow time is also a measure of work-in-process inventory. By minimising flow time, work-in-process is also reduced (Conway *et al.*, 1967).

In summary, the three selected dispatching rules used in this study are shown in Table 3.1. SPT and MINSLK are charegorised as local-dynamic dispatching rules as they change the order of a queue. FCFS is a local-random dispatching rule as it lets any job to arbitrality join a queue arbitralily.

Table 3.1. Selected dispatching rules

Criterion	*Description*	*Formula*
Local-random FCFS	First arrival at machine	r_{ik}
Local-dynamic SPT	Shortest processing time	$\min\{p_{ij}\}$
Local-dynamic MINSLK	Minimum slack time	$\min\{(d_i - t_{now}) - P_{ij}\}$

Notation
r_{ik} = the time when job i arrives at machine k,
p_{ij} = the processing time of the operation j for job i,
d_i = due date for job i,
t_{now} = simulation time-clock,
P_{ij} = total remaining processing time for job i at jth operation, where

$$Pij = \sum_{u=v_i}^{m_i} p_{iu}$$

m_i = total number of operations required for job i,
v_i = number of operations already completed for job I
p_{iu} = processing time for job i at uth operation

3.2.2. Expediting rules

The critical zone is the remaining time to expedite jobs to arrive on time when the constraint is to start consuming them. There are two views how to monitor holes and how to use the critical zone of a buffer. Firstly, Goldratt and Fox (1986) suggest expediting is taken if the hole has elapsed half buffer length. This critical zone is used to cover the remaining processing times of job to meet the schedule. Every job will be checked at half time buffer (TB_i). From the gating point, the checking point is the sum of total processing times (ΣPT_i) and half time buffer ($1/2TB_i$). It can be written as $CP_i = \Sigma PT_i + 1/2TB_i$. Since this checking point does not change, it is called static checking point as shown in the Figure 3.2.

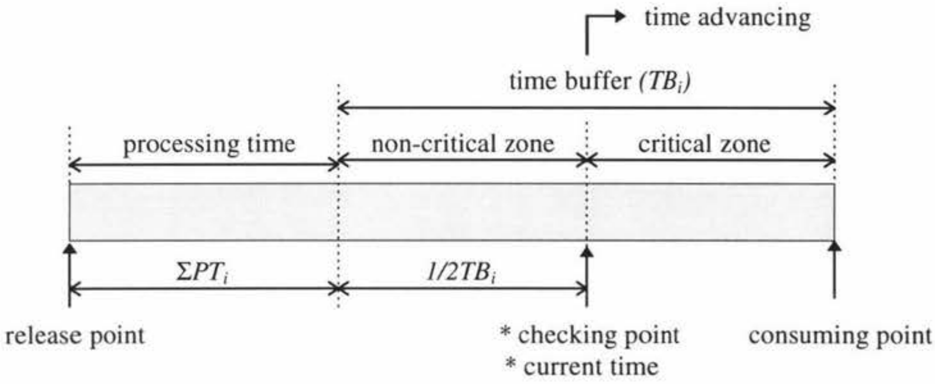


Figure 3.2. The static-checking point of Goldratt and Fox (1986)

The second buffering style comes from Hurley (1996). In this second buffering style, the critical zone is only used to recover the manufacturing line from exceptional circumstances when expediting is required, whereas the non-critical zone is applied to overcome the inherent variability in the processing time up to the constraint. Hurley (1996) recommends expediting should be determined based on the time discrepancy between planned and actual processing time before the jobs arrive at the protective buffer zone. If there are no disruptions in the system, the planned time remaining equals the actual time remaining and the job can arrive at the protective buffer as planned. On the other hand, any processing problems cause the actual processing time remaining to be greater than the planned time remaining. In fact, the actual processing time remaining is always greater than the planned time remaining due to non-instant availability of resources (Goldratt, 1990). Expediting rule using the time discrepancy is also termed a slack time rule. This slack time indicates the difference between the time remaining before the job is expected to arrive in the critical region and the amount of processing remaining before the job will actually arrive at the protective buffer. Thus the decision to expedite a job depends on the amount of its slack time.

Hurley (1996) points out that a job will be expedited if its time remaining before the job is expected to arrive at the constraint buffer (TR_i) and its non-critical zone [$(1-\alpha)TB_i$] is less than the time required by the remaining operations of job i (PR_i). The checking point is based on current time and monitored periodically as time advances. The expediting indicator can be expressed as the slack time of job i, $ST_i = [TR_i + (1-\alpha)TB_i] - PR_i$. If the slack time ($ST_i$) is less than zero, the job may need to expedite. In this study, the proportion of critical zone or coefficient α can be chosen at approximately one half to provide a good starting point for comparison test to the static expediting rule. This expediting rule is shown in Figure 3.3.

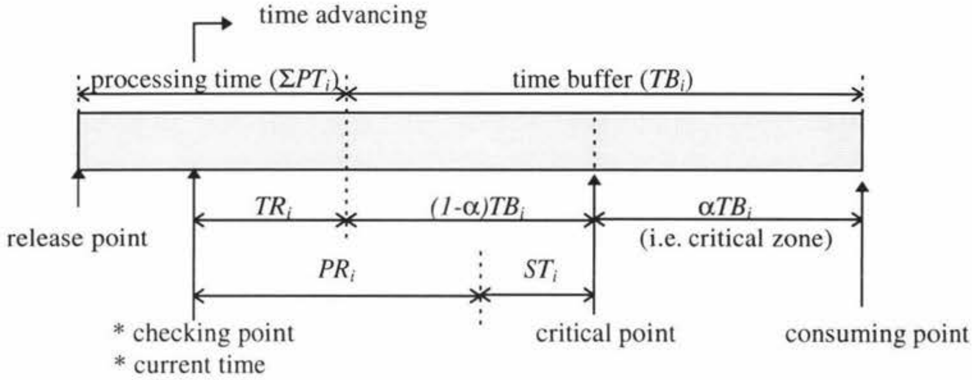


Figure 3.3. The dynamic-checking point of Hurley (1996)

The dynamic expediting rule claims to result in a shorter time buffer than the static expediting rule (Hurley, 1996). This implies that the expediting rules hold an important role in Buffer Management for allowing a firm to achieve a shorter lead time.

3.2.3. Variance reduction

One role of Buffer Management is to enable practitioners to identify and reduce causes of process variation. Statistical quality control (SQC) can be applied based on data from monitoring holes in the problem-solving zone of a buffer.

Reducing variability reduces dependability on expediting, and provides the opportunity to reduce the length of protective buffer and hence reduce WIP inventory. Practitioners are concerned on how efforts to reduce this variability affects shop performance. Continuous improvement is a means to reduce the variability of a facility. In this study, minimising the variance of operation processing times represents the effort of variance reduction.

The distribution of operation processing times indicates the level of variability in the shop. The exponential distribution represents high variability because its coefficient of variance (CV) is equal to one (Conway *et al.*, 1988). The 50% CV of the uniform distribution can be chosen to denote a medium variability environment (Melynk *et al.*, 1992; Crandall and Burwell, 1993).

Considering the three scenarios that have been discussed above, Figure 3.4 shows the experimental scenario as a basis to conduct simulation experiments. The outcome of scenario 1 is a favourable dispatching rule that is robust under a variety of mean protective capacities. This dispatching rule would be used in scenario 2 to test two expediting rules, whereas scenario 3 would employ the favourable expediting rule resulting from scenario 2. The next section describes the model including the experimental parameters.

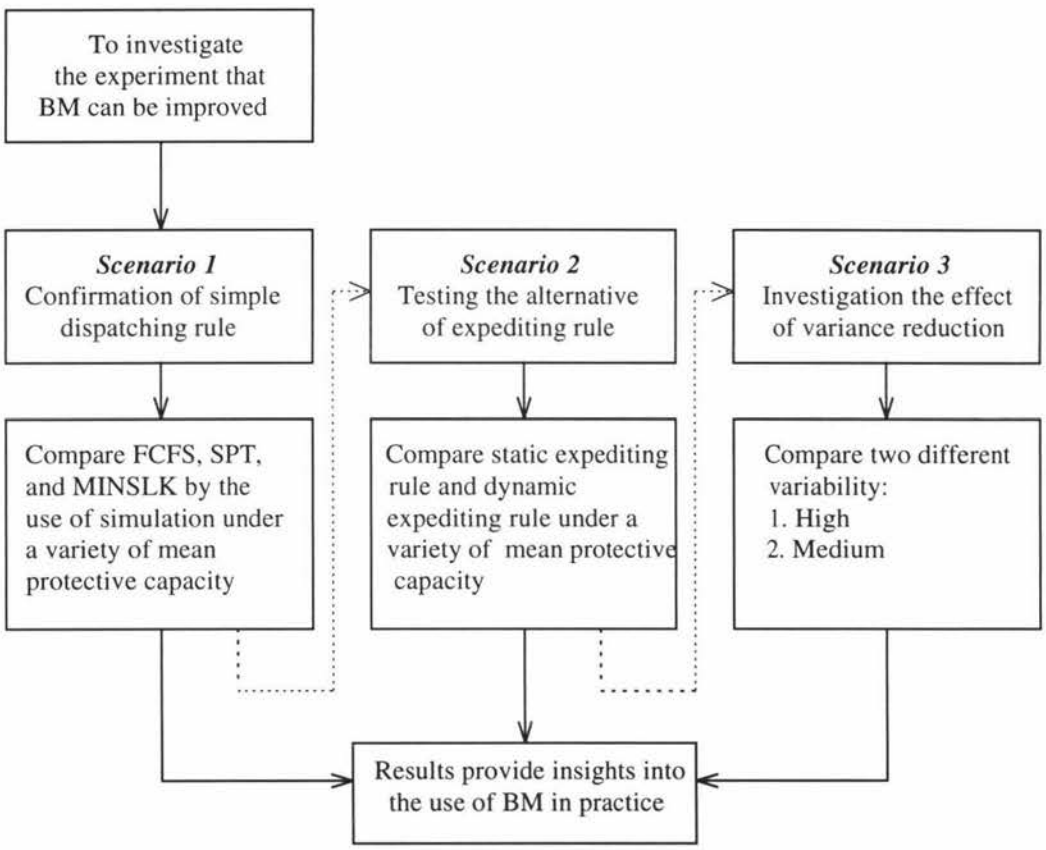


Figure 3.4. The experimental scenario

3.3. Model Description

A hypothetical general dynamic job shop model is used in this study. The control variables are dispatching rules, expediting rules and reduction in processing time variability.

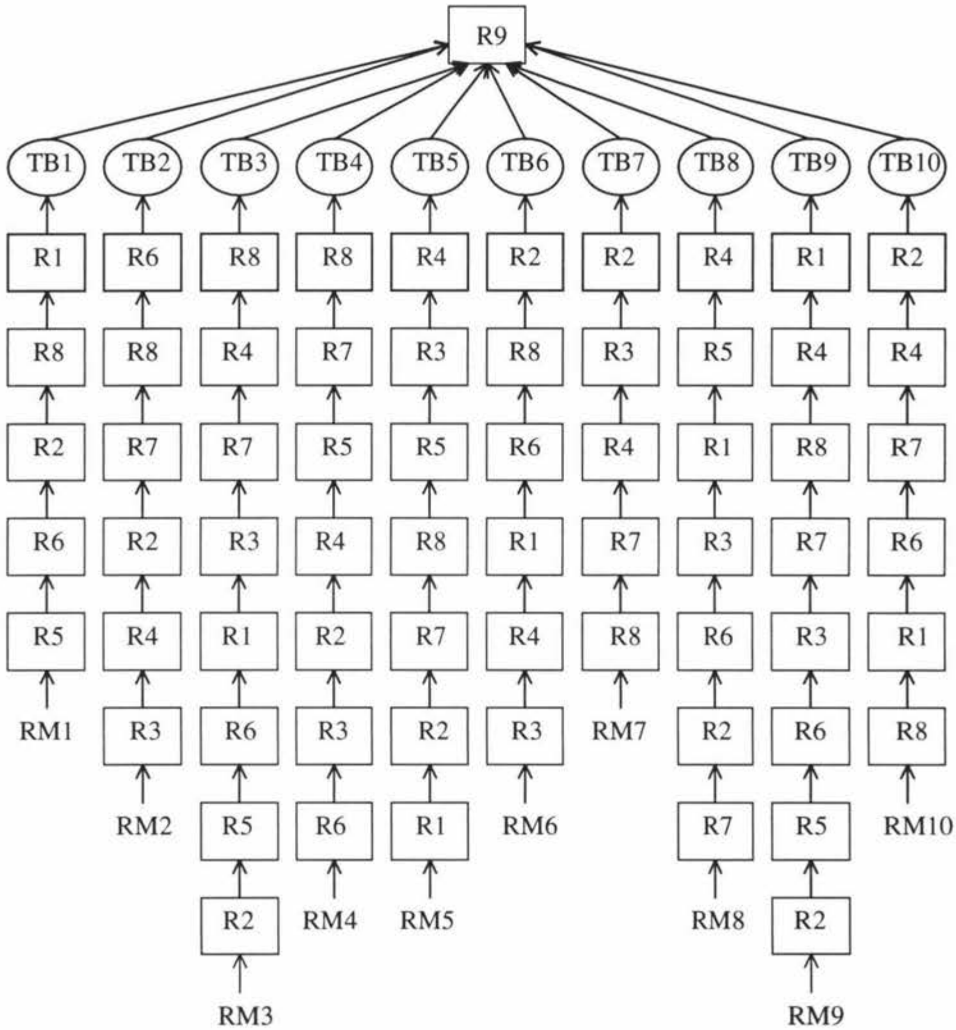
3.3.1. Job shop model

The DBR model operates in a make-to-order environment with ten types of products. The number of operations for each product is uniformly distributed between four and eight with the average being six operations. Random routing is assigned to each product. The processing times are generated randomly to different operations due to the routing (Appendix I shows the complete list of processing times for each routing). Order arrival is generated randomly and pooled in the master production schedule. Each order requires product type and demand, selected using the uniform distribution. The product demands range between ten to twenty products per order. Figure 3.5 depicts the structure of the products in this study.

Each machine is able to process a single operation at any point in time without preemptive operations. Processing times are exponentially distributed for high variability and uniform distribution for medium variability (Conway *et al.*, 1988). This shop conforms to the basic assumptions of a job shop (Conway *et al.*, 1967; Baker, 1974). This kind of job shop has been used widely in many previous studies (Ragatz and Mabert, 1988; Philipoom *et al.*, 1993; Huq and Bernardo, 1995).

3.3.2. Mean protective capacity

Resource nine serves as the constraint. The protective capacity for a non-constraint resource is the extra capacity above the productive capacity of the constraint resource. Its is available to recover from statistical fluctuations and disruptions taking place during the normal operation of a shop. The productive capacity reflects the processing rate of the resource constraint. Processing times are allocated to each routings to obtain a certain level of mean protective capacity of the non-constraint resources. Let T_c denote the processing time of the constraint, a period of time per unit of product and T_{nc} is the processing time of a non-constraint resource. The protective capacity (PC) of a non-constraint resource i can be written as $PC_i = T_c - T_{nc}$. It can be said that the processing time of non-constraint resource i is a function of its protective capacity PC_i, or $T_{nc} = T_c(1 - PC_i)$. Thus the mean protective capacity can be seen as the average proportion of the mean extra capacity above the constraint capacity of the non-constraint resources.



Legend:
Rj = resource *j*th.
RMi = raw material for product *i*.
TBi = time buffer for product *i*.

Figure 3.5. The product flow diagram

If *MPC* denotes the mean protective capacity as a percentage, the relationship between constraint processing time and the non-constraint processing time given a certain level of *MPC* and k number of non-constraint resources is given below.

$$MPC = \frac{\sum_{i=1}^{k} \frac{Tc - Tnc_i}{Tc}}{k} \text{ x } 100\% \qquad (1)$$

The four levels of mean protective capacities used in this study are 10%, 20%, 30%, and 40% (Appendix II provides the planned mean protective capacity). Mean protective capacity at each resource is sampled from a uniform distribution but the overall mean protective capacity for the shop should be the given percentage.

The reason to select different mean protective capacity is to consider the trade-off between the cost of WIP in front of the resource constraint and the protective capacity in attaining good customer response and a stable lead time.

3.3.3. Master production schedule

The model was developed so that the last is the constraint resource. This machine is used to build a master schedule from the arriving orders. The purpose of this master schedule is to maximize throughput (Spencer and Cox, 1995a). The master schedule is developed based on the First-Come-First-Served rule. The order of arrival is assumed to reflect the order of the due dates. This rule becomes equivalent to Earliest-Due-Date rule as suggested by Godratt and Fox (1986).

3.3.4. Time buffer

A time buffer between material release and the constraint resource is proportional in length to the lead time from the first operation up to the constraint resource. Guide (1995) suggests the time from the first operation to reach the constraint resource is equal to total processing times and time buffer. The buffer time is set equal to a selected buffer multiplier times the total time for processing. The buffer multiplier then is adjusted depending on the fluctuations in the manufacturing line.

The buffer length for each job is different. It depends on the lead time to the constraint and the constant buffer length multiplier (Guide, 1995; Guide and Ghiselli, 1995). The buffer length of any job *i* is given as follows.

$$BL_i = M \sum_{j=1}^{m} PT_{ij} \qquad (2)$$

where

BL_i = buffer length for job i $(i = 1,2,..,n)$
PT_{ij} = minimum processing time for operation j $(j=1,2,...,m)$ on job i
M = constant buffer length multiplier.

Buffer length multipliers are selected to yield a desired percentage of tardy jobs (Philipoom *et al.*, 1993). If the percentage of tardiness is more than ten percent of operations that should arrive at the constraint resource, the buffer needs to be longer. The similar consideration is taken if there are too few jobs tardy.

3.3.5. Release mechanism

From the master schedule, the quantity and release time of a job can be planned (Wisner, 1995). This is the rope component of DBR. The release date is calculated backwards from the date the job is required by the constraint. It is simply the amount of time a job spends in operations, processing the buffer length and time spent by the previous jobs at the constraint. The release mechanism for a job i is represented below.

$$r_i = MS_t - P_i - BL_i + S_i \tag{3}$$

Where:

r_i = release time for job i.
MS_t = master schedule time when the first job will be processed at the drum.
P_i = total processing time for job i to the constraint.
BL_i = buffer length for job i.
$RL_i = P_i + BL_i$ = rope length for job i.
S_i = sequence factor at the master schedule for job i.

The release time for a certain job is simply offsetting from the master schedule, that is the amount of rope length (RL_i). Sequence factor for a job refers to the sum of processing time previous jobs before the job in question is processed by the constraint machine. The first job released to the shop depends on the rope length and sequence factor (SFi) in the first few jobs in the master schedule. An example of this relationship can be seen in the Figure 3.6.

The sequence factor for job 1 is zero or equal to the starting time of the first job on the constraint machine. The sequence factor for job 2 is the sum of previous sequence factor (SF_1) and previous processing time at the constraint machine (CT_1). In general, it can be stated as follows.

$$SF_1 = 0 \tag{4}$$

$$SF_i = SF_{i-1} + CT_{i-1} \tag{5}$$

Delta (δ) is the maximum difference between rope length (RL_i) and sequence factor (SF_i) or

$$\delta = max\{ RL_i - SF_i \} \tag{6}$$

The starting time for the master schedule is the present time (setting at $t = 0$) plus δ, or

$$MS = t_{now} + \delta \tag{7}$$

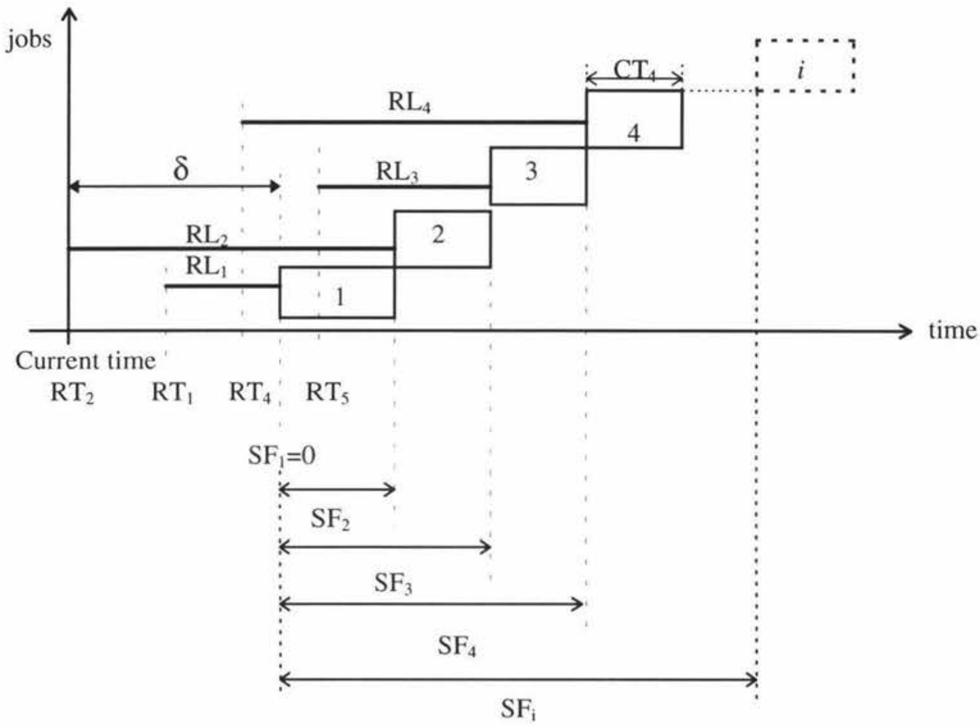


Figure 3.6. The gantt chart for job time releasing

3.3.6. Due date assignment

Due date is simply the job's arrival date plus an estimated lead time. Lead time is the sum of time a job spends queuing for individual operations and actual processing times. The due date rule used is total work content (TWK). Baker (1984) concluded that TWK was a reliable and effective technique for setting due dates. Time buffer reflects the flow allowance that is proportional to total work. The due date rule used is an adaptation of the TWK rule as follows.

$$d_i = r_i + RLi + p_i \tag{8}$$

where

d_i = due date of the ith job

p_i = processing time of the ith job at the constraint
r_i = release time of job i, and
RL_i = rope length of job i

3.3.7. Dispatching mechanisms

There are three different rules used in this study. The first rule is First Come First Served (FCFS), that is to authorise a resource to work with the first product in the queue. Goldratt and Fox (1986) adopt FCFS dispatching rule in the DBR and BM environment.

The second rule is minimum slack time (MINSLK). A priority is given to the job with the least amount of slack. Jobs are sequenced in order of increasing slack time. Slack time is the different between time remaining before the due date and work remaining. Typically the job with positive slack is ahead of schedule, zero slack is on time, and negative slack is behind schedule. The job selected from the queue to be processed is the one with the most negative, or at least positive slack time.

For comparison to the dispatching mechanisms above, the shortest processing time (SPT) is adopted. SPT gives priority of selecting a job whose the shortest processing time to be performed by a resource.

3.3.8. Expediting mechanisms

Two different ideas of expediting are tested in this simulation model. The first expediting rule is suggested by Goldratt and Fox (1986) and the second expediting rule is introduced by Hurley (1996). Goldratt and Fox (1986) assert that a job that has not arrived at the critical buffer zone is likely to be late if the remaining time required to complete the job is longer than the length of the critical buffer zone. This job should receive special attention such as expediting. The job's operations are given priority to be at the constraint resource on time.

Hurley (1996) defines slack time for job i as the difference between the time remaining before the job is expected to arrive in the buffer region and the amount of time required before the job actually arrives at the buffer. This slack time is shown below.

$$ST_i = [TR_i + (1-\alpha)TB_i] - PR_i \qquad (9)$$

where

ST_i = slack time for job i.
TR_i = time remaining before the job expected to arrive at the constraint buffer.

α = the proportion of time buffer that indicates the critical zone.
TB_i = time buffer of job *i*.
$(1-\alpha)TB_i$ = the non-critical zone for job *i*.
PR_i = time required by the remaining operations of job *i* to arrive at the constraint resource.

Equation (9) can be rewritten so that slack time is the different between theoretical time to the critical point and theoretical time remaining to complete the job or

$$ST_i = (t_{critical\text{-}point} - t_{now}) - P_{ij} \qquad (10)$$

where

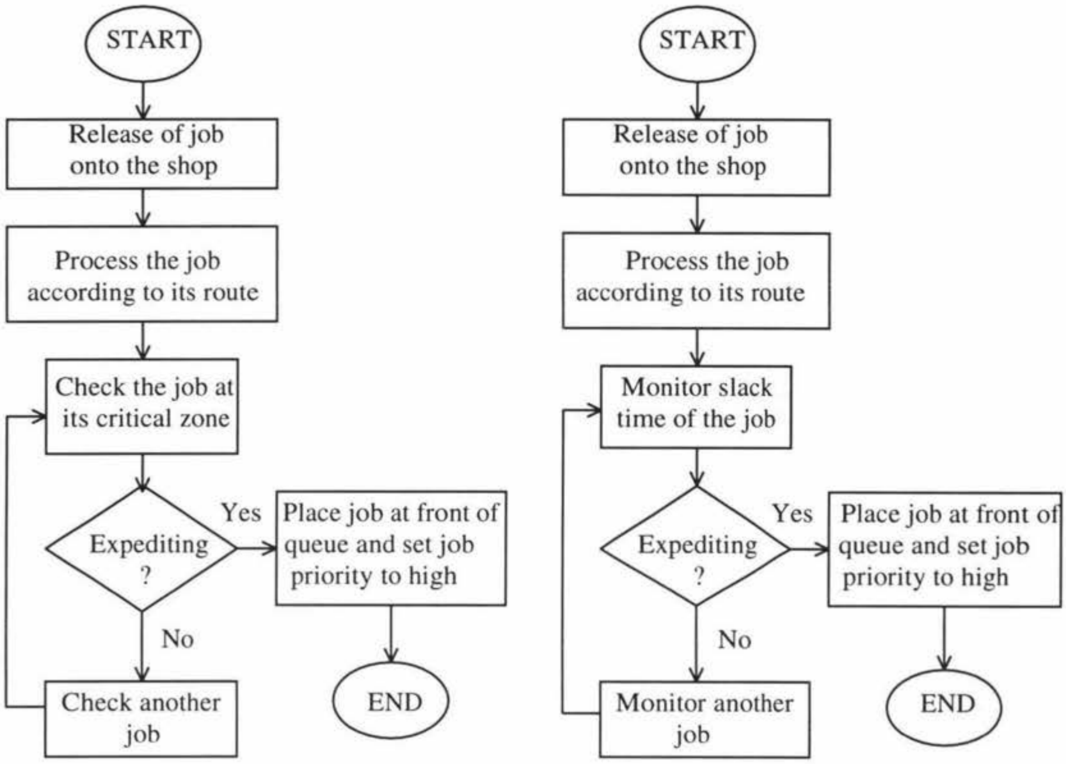
ST_i = slack time for job *i*.
$t_{critical\text{-}point}$ = the time for job i to arrive at the critical point.
t_{now} = simulation time-clock,
P_{ij} = total remaining processing time for job *i* at *j*th operation, where

$$P_{ij} = \sum_{u=v_i}^{m_i} p_{iu}$$

p_{iu} = processing time for job *i* at *u*th operation
m_i = total number of operations required for job *i*,
v_i = number of operations already completed for job *i*.

When the slack time is negative, the job will not arrive at the critical region and should be expedited. Figure 3.7 gives the flow chart of two expediting rules used in the simulation model.

The amount of expediting provides a clue to revising the buffer size. If more than ten percent of the jobs are expedited, the buffer size needs to be enlarged. If there is no expediting, the buffer size needs to be shortened. This mechanism helps DBR to attain a suitable buffer size.



(a) Static buffer control (b) Dynamic buffer control

Figure 3.7. The procedure of expediting rule

3.4. Performance Criteria

The primary measures for this study are cycle time or flow time (*F*), queue length in front of the constraint resource (*L*), proportion of jobs tardy (*T%*), and delay time in arriving at protective buffer (*DT*). Cycle time indicates how long a job spends in the system. Cycle time is measured as the elapsed time from material release until a job's completion. Queue length in front of the constraint is the average number of jobs in the buffer. Proportion of jobs tardy refers to the number of jobs late relative to due date. The delay for a job arriving at the protective buffer is the difference between lead time to the buffer and actual time for a job to arrive at the buffer. Table 3.2 summaries the performance measurements and nomenclature used in this study.

Table 3.2. Shop performance criteria

Measure	*Description*	*Formula*
Inventory performance		
Flow time F	Average job flow time	$\sum_{i=1}^{N}(Ci - ri)/N$
Queue length of the constraint L	Average queue length	$\sum_{t=1}^{T} Lt/T$
Delay time D	Average delay time	$\sum_{i=1}^{N}(Ei - li)/N$
Due date performance		
Percent jobs Tardy $T\%$	Percentage of jobs completed after their due date	N_T/N

Notation

N = number of jobs completed or processed for which statistics are collected
r_i = time job i is released to the shop
C_i = completion time of job i
l_i = lead time of job i to be arrived at protective buffer
L_t = queue length of jobs in front of the constraint at period t
T = length of period for which statistics are collected
E_i = actual entering time of job i in protective buffer
N_T = number of jobs completed after their due date

When studying expediting rules and the effects of variance reductions, number of parts requiring expediting per day is used as an additional performance measure. The performance rankings of dispatching rules and expediting rules are assumed not affected by other factors such as machine breakdowns, overtime, batch splitting, and alternative routings.

3.5. Simulation Model

Due to the complex dynamic nature of a manufacturing facility arises there is no practical and general method of ensuring an optimum solution for scheduling. It is difficult to embrace all complexities of manufacturing line such as inherent variability, non-instant availability, and breakdown into a scheduling system. If this fact is accepted, then it is reasonable to consider such problems using a simulation model, rather than attempting explicitly to build an analytical model.

Simulation of hypothetical job shops are used to investigate the impact of dispatching, expediting, and variance reduction policies on cycle time and other performance criteria. The hypothetical job shop makes it possible to determine the system performance under different scenarios. Ragatz and Mabert (1988), Jacobs and Bragg (1988), and Philipoom *et al.* (1993) also suggest that the different is not significant between a hypothetical job shop and an actual shop and by using hypothetical job shops the previous studies are more readily to compare.

Simulation programs have been written in SIMSCRIPT II.5 (Russell, 1989; Huq and Bernardo, 1995). Appendix III presents the flow chart of simulation logic. Appendices IV and V contain the examples of program listings used in this study.

The next section summarises the parameters and decision variables used in the experimental design. The simulation results are interpreted in order to give insights for practitioners as well as future research.

3.6. Experimental Design

This study seeks to determine the effects of experimental factors on due date and inventory performance. There are three experimental scenarios designed in this study. Firstly, scenario one is dedicated to examine three dispatching rules. Two expediting rules are compared in scenario two. Finally, scenario three investigates the effects of reduction in processing times. Experimental parameters are the release mechanism, due date assignment, time buffer, mean protective capacity, master production schedule, and product routings. The complete experimental design for the three scenarios in this study is shown in Table 3.3.

Measures of shop performance used are average cycle time, average queue length in front of the resource constraint, average delay time to the buffer, and proportion of jobs tardy. Cycle time, average queue length, and delay time are used as the inventory performance measures, whereas percentage jobs tardy measures the due date performance. The results are subjected to the usual statistical tests such as autocorrelation, normality, data sufficiency, and homogeneity variance (Fishman, 1978; Hicks, 1982; Law and Kelton, 1991). Scenario one is treated as a one-way analysis of variance (ANOVA) to determine if the experimental factors significantly affect the shop performance at a five percent level of significance. The dependent variables are then compared using Tukey's multiple comparison test (Hicks, 1982; Steel and Torrie, 1980). The level of significant used for the multiple comparison is 0.05. Scenario two uses comparison tests to determine the effect of expediting rules on shop performance. Scenario three uses comparison tests to determine the effect of variance reduction and expediting rules on shop performance.

Table 3.3. Experimental design

Experimental Conditions	*Description*
Simulation Parameters	
- Product types (routings)	User defined discrete uniform probability distribution
- Mean protective capacity	MPCs of 10%, 20%, 30%, and 40%
- Master production schedule	Earliest-Due-Date
- Release rule	Built based on offset time from the master production schedule
- Processing times for non-constraint resources (only for scenario 1 and 2)	Exponential distribution
- Processing times for the constraint resource	Constant distribution
Simulation Factors	
- Dispatching rules	FCFS, SPT, MINSLK
- Expediting rules	Static and dynamic rules
- Variance reduction	User defined processing times: Exponential distribution (high) and Uniform distribution (medium) or coefficient Variations of 100% and 50%.

3.7. Data Analysis Issues

This study uses a simulation approach to investigate the effects of control variables on shop performances under the DBR environment. During the simulation, data is gathered in a particular time period or batch. A horizon of simulation run is divided into batches or sub-runs. Each batch produces one observation for each performance measure as shown in Figure 3.8.

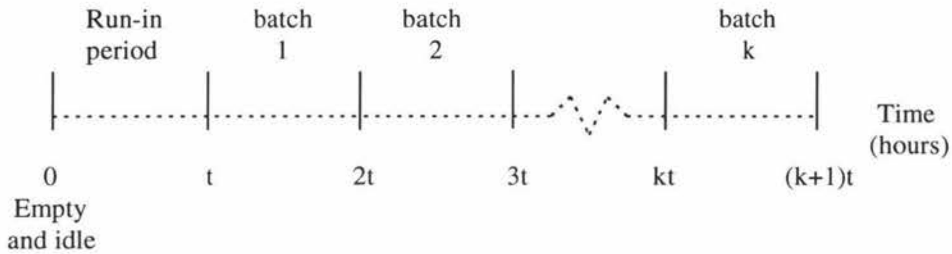


Figure 3.8. Batch means data collection

Since the simulation model is a non-terminating system, the concern is the behaviour of steady-state responses. Data from the initial phase or transient period is discarded. The period of observation *t* after the shop reaches steady state condition is determined so that each batch is independent. Fishman(1978) provides a method to determine this observation period length. Sufficient samples are drawn from each batch in order to evaluate the performance of the job shop.

The data collected from the simulation results are analysed to derive some inferences about the model. Statistical tests are used to verify the simulation model such that the inferences made are not biased. This section attempts to describe the statistical methods used in this study, such as, tests for steady state conditions, autocorrelation, data sufficiency, normality, homogeneity of variances, one-way analysis of variance and *t*-tests for comparison.

3.7.1. Test for steady state conditions

The output (Y_i) of a non-terminating system is an estimate of the steady state mean μ = E(Y), that can be generally defined by μ = lim E(Y_i) where $i \to \infty$. For non-terminating systems, the initial data in the transient phase has to be discarded and the remaining data used to estimate the expected steady state response μ. This process is called the initial-data deletion method. The practical way of determining the initial phase or warm-up period is a graphical period based on modified Welch's procedure (Law and Kelton, 1991). The objective is to select a warm-up period l, such that $E(Y_i) \approx \mu$ for $i > l$. Welch's procedures consists of four steps to determine the warm-up period l for n independent replications of the simulation.

1. Make n independent replications ($n \geq 5$). Each replication consists of m simulation runs. The length m is chosen large enough to ensure that the transient period is over. Let Y_{ij} be the ith observation from the jth replication ($i = 1, 2, ..., m; j = 1, 2, ..., n$).
2. Let $\overline{Y}_i = \Sigma Y_{ij}/n$ for $i = 1,2, ..., m$. The averaged process has mean $E(\overline{Y}_i) = E(Y_i)$ and variances $Var(\overline{Y}_i) = Var(Y_i)/n$.
3. The moving average $\overline{Y}(w)$, where w is a positive integer and $w \leq \lfloor m/2 \rfloor$, is used to smooth out the high-frequency oscillations in the means $\overline{Y}_1$, $\overline{Y}_2$, ... , but leave the low-frequency oscillations or long-run trend of interest.
4. Plot the moving average $\overline{Y}_i(w)$ for i = 1,2, ..., m-w and choose l as the value of i beyond which $\overline{Y}_1$(w), $\overline{Y}_2$(w), ... appears to have converged.

A statistical test should be conducted to confirm that the system reaches the steady state condition after the warm-up period l. The mean of the first difference between the successive mean of response after smoothing should be non-zero before the shop reaches steady state. This mean should converge to nearly zero after achieving steady state. The mathematical representation is given as follows.

Before steady state: $\Delta y_i = Y_{i+1} - Y_i$; mean $\Sigma \Delta y_i/n \neq 0$ and (11)

After steady state : $\Delta y_i = Y_{i+1} - Y_i$; mean $\Sigma \Delta y_i/n = 0$ (12)

The period length after warm-up was chosen the same as the sample size before warm-up period to provide a balanced comparison. The null hypothesis to be tested was, H_o: mean

$\Sigma\ \Delta y_i/n = 0$ for both periods. If the sample size is more than thirty, the test statistic is based on the Z-test with the standard deviation estimated from the sample data. The test statistic is given by

$$z = \frac{\sum_{i=1}^{n} \frac{\Delta yi}{n} - 0}{s / \sqrt{n}} \qquad (13)$$

The one tailed alternative hypothesis is given by H_a: mean $\Sigma\ \Delta y_i/n > 0$. The five percent tabulated value of z is 1.6449. If the data is normally distributed with the sample size is less than thirty, the following t-test with the standard deviation estimated from the sample data can be used.

$$t = \frac{\sum_{i}^{n} \frac{\Delta y_i}{n} - 0}{s / \sqrt{n}} \quad \text{with } (n\text{-}1) \text{ df} \qquad (14)$$

The one tailed alternative hypothesis H_a: mean $\Sigma\ \Delta y_i/n > 0$ the critical value of t with particular (n-1) df under the 5% level of confidence can be found in the Student's t table.

3.7.2. Autocorrelation test

Batch means is a method to collect non-overlapping subsamples or batches from a simulation experiment. Data collection begins at a point in a run where the initial conditions do not influence the behaviour of the statistical sequence. Each observation taken in a certain range or period is assumed independent. In other words, there is no autocorrelation. The length of observation has to ensure that the subsample averages are approximately independent. The verified data can be used to draw some inferences about a particular stochastic sequence, for instance the sample mean and the estimation of the variance of the sample mean.

Fishman (1978) suggests a statistical test for the hypothesis of independence. Let $Y_{i,m}$ denote the average of the jth replications within m periods of observation, where k is the number of replications within n simulation runs. The grand average is

$$\bar{y} = \sum_{i=1}^{k} \frac{y_{i,m}}{k} \qquad i = 1,\ldots, k = \langle n/m \rangle \qquad (15)$$

If the batch averages $(Y_{1,m},...,Y_{k,m})$ are approximately independent and normally distributed, then the test statistic for independent batches is

$$C_k = 1 - \frac{\sum_{i=1}^{k-1}(Y_{i,m} - Y_{i+1,m})^2}{2\sum_{i=1}^{k}(Y_{i,m} - \bar{y}_k)^2} \tag{16}$$

The C_k statistic has mean zero and variance $(k-2)/(k^2-1)$. The distribution of C_k for normal batches is close to normal for $k \geq 8$, then $C_k/\sqrt{(k-2)/(k^2-1)}$ is treated as standard normal distribution whose means is zero and variance is one. The use of C_k statistic enables a particular stochastic sequence to be obtained from as few as eight batches.

C_k can be used to test for independence among sub-sample averages. If m is small, it is doubtful that subsample averages are normal. One has to increase m such that C_k is less than its critical value, or $C_k \leq Z_{\alpha/2}\sqrt{(k-2)/(k^2-1)}$. Since the autocorrelation value may be positive as well as negative, a two-sided hypothesis test can be defined as follows.

H_o : $C_k = 0$ or independent batches

H_1 : $C_k \neq 0$ or correlated batches.

H_o is accepted in favour of H_1 if the absolute value of

$$Z_k = \frac{C_k}{\sqrt{(k-2)(k^2-1)}} \tag{17}$$

is less than $Z_{\alpha/2}$ where $Z_{\alpha/2}$ is the upper $\alpha/2$ point on the standard normal distribution.

3.7.3. Test for data sufficiency

Data sufficiency test was adopted to determine that the available data were adequate to represent the behaviour of the shop. The behaviour of the shop depends on two factors, that is, the nature of random variability in indicator variables and the degree of confidence desired in the estimates. If the shop has a high variability, a large number of data points are needed to ensure a certain level of confidence. Thus, the formula to determine sample size is based on the premise that the number of replications required in a simulation experiment depends on the variability present in the simulation output. In this study, two methods for testing data sufficiency are adopted. For the low variability of the variable response, the Law and Kelton formula is chosen (Law and Kelton, 1991). Keijnen's formula is used when the variability is high, thereby a large sample size is needed (Kleijnen, 1987).

Law and Kelton (1991) proposes a formula to determine the sample size. After discarding the data before warm-up period and using the initial n runs (pilot runs or epochs), average performance statistics $\bar{\bar{y}}(n)$ will be estimated. The 100(1-α) percent confidence interval for $\bar{\bar{y}}(n)$ is defined as follows.

$$\bar{\bar{y}}(n) \pm t_{1-\alpha/2,n-1}\sqrt{\frac{s^2(n)}{n}} \tag{18}$$

Where

$s^2(n) = \frac{\sum_{i=1}^{n}[\bar{y}_i - \bar{\bar{y}}(n)]^2}{n-1}$, is the variance of the simulation output.

$\bar{\bar{y}}(n)$ is the average of the mean performance statistics from n pilot runs.

$\bar{y}(n)$ is the time average performance statistics from each run.

The performance statistics are collected following the warm-up period. If the runs are not autocorrelated, then the observations can be treated as if they were *IID* normal random variables. The number of replications necessary for a 100(1-α) percent confidence in the performance statistics is given by:

$$\eta^*(\beta) = Min.\{i \geq n: \frac{(t_{1-\alpha/2,i-1})\sqrt{s^2(n)/i}}{\left|\bar{\bar{y}}(n)\right|} \leq \beta \tag{19}$$

Where η* is the number replications, β is the pre-selected relative half confidence interval, i is the number of runs necessary to make the relative half confidence interval less than or equal to β, and n is the initial number of runs.

Kleijnen (1987) suggests a formula to determine the number of runs required to achieve a specified level of precision. The mean μ of the normal population $N(\mu, \sigma^2)$, assuming a known σ^2, is estimated by the sample average $\bar{y}$ from n independent observations $y_1, y_2, ..., y_n$. Kleijnen's formula is based on the premise that the mean μ can be estimated by the estimator $\bar{y}$ such that the error is less than c units. The 100(1 - α)% level of confidence for the estimator $\bar{y}$ is given by:

$$P(|\bar{y} - \mu| \leq c) = 1 - \alpha \quad (20)$$

For the n independent normal variables y_i, the planning error $c = z_{\alpha/2}\sigma(y)/\sqrt{n}$ must hold. If the parameter $\sigma^2(y)$ can be replaced by its estimator, the estimation can use Student's t-statistic. The required sample size then should satisfy:

$$n = \left(\frac{t_{1-\alpha/2;n_o-1}}{c} \right)^2 s_o^2 \quad (21)$$

where:

n = number of simulation runs required

c = half-width of confidence interval

s_o^2 = sample variance for no observations

$t_{1-\alpha/2;no-1}$ = t-statistic with n_o-1 degrees of freedom

n_o = number of pilot runs of simulation

If the required sample size n is more than the number of pilot runs n_o, then the n-n_o additional observations should be taken. The mean will be updated, but the sample variance will remain fixed at s_o^2. Kleijnen (1987) recommends that the planning error c can be estimated by $c = 0.20\bar{y}$. For exploratory research, $c = 0.50\bar{y}$ is common to be used. If there are more than one system performance measure, the largest value of n is used for gathering observations.

3.7.4. Normality test

The output data should satisfy normality tests and homogeneity of variance. The null hypothesis asserts that an output process $F(y)$ is a normal distribution function with unspecified mean and variance, whereas the alternative hypothesis states that $F(y)$ is non-normal. The Shapiro-Wilk's test is adopted to examine that the performance statistics are normally distributed (Conover, 1980). The Shapiro-Wilk's test is often called the W test. The test statistic W is given by

$$W = \frac{1}{D}\left[\sum_{i=1}^{k} a_i (Y^{(n-i+1)} - Y^{(i)})\right]^2 \tag{22}$$

where

$D = \sum_{i=1}^{n} (Y_i - \overline{Y})^2$ is the denominator D of the test statistic,

$\overline{Y}$ is the sample mean, and

$Y^{(i)}$ denotes the ith order statistics.

The coefficients a_1, a_2, ..., a_k are tabulated as given in Table for Coefficients for the Shapiro-Wilk Test (Conover, 1980) where k is approximately $n/2$. The decision rule is to reject H_o at the level of significance α if W is less than the α quantiles as given in the Table for Quantiles of the Shapiro-Wilk's test statistic. The following steps must be carried out to test for normality.

1. Order the n observations as $y_1 \leq y_2 \leq y_3 \leq ... \leq y_n$.
2. Compute the denominator D.
3. If n is even, $n = 2k$, compute

   $$b = \sum_{i=1}^{k} a_i (y^{n-i+1} - y^{i})$$

   where the values of a_i appear in Table for Quantiles of the W test for normality.

   If n is odd, $n = 2k + 1$ then one omits the sample median, y_{k+1}, and calculates b.
4. Compute $W = b^2/D$.
5. Compare W to the percentage points given in Table for percentage points of the W test. Small values of W indicate non-normality.

3.7.5. Test for homogeneity of variances

In most analysis of variance problems, the assumption of homogeneous variances is crucial. If this assumption is violated, the data needs to be transformed to allow the analysis of variance to produce meaningful results. Therefore, tests for homogeneity of variances mainly employ the 1% level of significance. If the homogeneity test is accepted at $\alpha = 0.01$, the use of the analysis of variance for the performance statistics is valid.

Bartlett's test was adopted to test for homogeneity of variances. The test statistic for Bartlett's test is given by

$$B = \frac{\left(\sum_{i=1}^{k} v_i\right)\ln \bar{s}^2 - \sum_{i=1}^{k} v_i \ln s_i^2}{1+\left\{\frac{1}{3(k-1)}\right\}\left\{\sum_{i=1}^{k}\frac{1}{v_i} - \frac{1}{\sum_{i=1}^{k} v_i}\right\}} \tag{23}$$

where

$s_i^2 = \sum_{j=1}^{ni}(Y_{ij} - \overline{Y_{i.}})^2 / (n_i - 1)$ is the variance of the sample; i = 1, 2, ..., k;

j = 1, ..., n_i,

$\bar{s} = \frac{\sum_{i=1}^{k} v_i s_i^2}{\sum_{i=1}^{k} v_i}$

k is the number of treatments or samples, and

$v_i = n_i$ -1 is the number of degree of freedom associated with variance s_i^s.

Bartlett's test uses a chi-square statistic to test the presence of unequal variances in different samples. The B value is equivalent to χ^2 value which has k-1 degrees of freedom. If the B value is greater than the critical value, then the variances are unequal, and the conclusions from the analysis of variance are doubtful. If the p-value is less than the level of significance α, this also indicates there is the absence of homogeneity of variance.

Since the Bartlett's test is sensitive to the assumption of normality, Levene's test is needed to be carried out to confirm the result of Bartlett's test (Levene, 1960). Suppose the direct observed random variables Y_{i1}, Y_{i2}, ..., Y_{ij}, ..., Y_{ini} in a sample of size n_i are taken from a population π_i. Levene's method takes into account the distances of the observations from the sample median rather than the sample mean. It means the direct observations are replaced by random variables $V_{ij} = | Y_{ij}-M_i|$ where M_i is the median of Y_{i1}, Y_{i2}, ..., Y_{ini}. Equality of variances in populations π_1, π_2, ..., π_i, ..., π_k is then equivalent to equality of expected values of the V_{ij}'s. Using the sample median rather than sample mean makes the test more robust for smaller samples as well as making the procedure asymptotically distribution-free. The test statistic is given by

$$L = \frac{(N-k)\sum_{i=1}^{k} n_i(\overline{V_{i.}} - \overline{V_{..}})^2}{(k-1)\sum_{i=1}^{k}\sum_{j=1}^{ni}(V_{ij} - \overline{V_{i.}})} \tag{24}$$

where

$V_{ij} = \left|Y_{ij} - \overline{Y_i}\right|$ is the distance from the median for $i = 1, ..., k$ and $j = 1, ..., n_i$

Y_{ij} is the jth observartion on the ith population

$\overline{Y_i}$ is the median of direct observations $\{Y_{i1}, ..., Y_{ini}\}$

$\overline{V_{i.}}$ is the mean of the distance variable from median for population i

$\overline{V_{..}}$ is the mean of all k populations.

3.7.6. One way analysis of variance

One way analysis of variance is a standard analysis technique for a single factor experiment where only one factor is varied (Hicks, 1982). The level of the factor to be examined can be quantitative or qualitative, fixed or random. The number of observations or runs for each level is determined by the statistical significance desired and cost consideration. One way analysis of variance can be presented with the following mathematical model.

$$Y_{ij} = \mu + \tau_j + \varepsilon_{ij} \tag{25}$$

where Y_{ij} represents the ith observation ($i = 1,2, ..., n_j$) on the jth treatment ($j = 1,2, ..., k$ levels). μ is a common effect in all observations. τ_j represents the effect of the jth treatment. ε_{ij} denotes the random error present in the ith observation of the jth treatment. The random error ε_{ij} is usually assumed as a normal and independently distributed random effect whose mean value is zero and whose variance σ^2 is homogenous for all treatment j.

The null hypothesis is that $\tau_j = 0$ for all j. If the hypothesis is true, then varying the levels of the factor has had no effect and each observation Y_{ij} is made up of its population mean μ and a random error ε_{ij}. If the null hypothesis is rejected, it is concluded that at least one of three dispatching rules is different significantly from the others. The statistical technique that permits further comparisons to be made between treatment means are referred to as multiple comparison methods.

The multiple comparison test is based on Tukey's Studentised Range Test to compare all pairs of treatment means (Steel and Torrie, 1980). Tukey's procedure assumes that the parameter r (number of steps between averages) is equal to t for each pair of means. Tukey's (*HSD*) Studentized Range Test was carried out by calculating,

$$HSD = q_{\alpha}(t, f_e)\sqrt{\frac{MS_e}{r}} \tag{26}$$

where $q_{\alpha}(t, f_e)$ is obtained from table for upper percentage points of the studentised range. The number of treatments to be compared is denoted by t, and f_e is the degree of freedom associated with the mean square error. From the analysis of variance table, the mean square error (MS_e), the number of replications r, and the degree of freedom f_e were obtained. The treatments are arranged in ascending order of magnitude. Treatments t and 1 are significantly different if

$$\left|\overline{Y_t} - \overline{Y_1}\right| > HSD \tag{27}$$

The procedure next tests treatments t-1 and 1. This continues until all paired comparisons are made. Tukey's procedure controls the family-wise error rate to α, which means that an experiment with only one incorrect decision is undesirable as an experiment with many incorrect decisions. This procedure can be used to determine which dispatching rule consistently yields better performance indicators.

The Tukey-Kramer procedure is used for pairwise contrast if the sample size is unequal (Kramer, 1956). It controls the experimentwise error rate by using the interval:

$$Y_{i.} - Y_{j.} \pm q_{\alpha}(t, f_e)\sqrt{MS_e}\sqrt{\frac{1}{2}\left(\frac{1}{n_i} + \frac{1}{n_j}\right)} \tag{28}$$

where $q_{\alpha}(t, f_e)$ is the upper α quantile of a studentised range distribution. If the interval contains zero, the two sample means are not significantly different at α. In the one-way ANOVA situation one would use $q_{\alpha}(t,N-t)$. The parameter t is the total number of means or levels, and N is the total sample size.

If the data is not normal, the nonparametric test is used to analyse the effects of variable controls on the simulation response. Kruskal-Wallis test offers a nonparametric alternative to the usual one-way analysis of variance and is based on the ranks of the data. The test assumes that the data arise as k independent random samples from continuous distributions all having the same shape or from identical distributions with the same median (or mean if the distributions are symmetrical). The null hypothesis of no differences among the k populations is tested against the alternative of at least one difference. The procedure can be summarised as follows

1. Rank the combined data, using tied ranks if necessary.
2. Calculate S_i = sum of the ranks of the ith sample, $i = 1, 2, ..., k$.
3. The test statistic is

$$H = \frac{12}{n(n+1)}\sum_{i=1}^{k}\frac{S_i^2}{n_i} - 3(n+1) \tag{29}$$

where $n = \sum_{i=1}^{k} n_i$, the overall sample size (n_i = size of the ith sample)

4. The rejection region is $H \geq \chi^2_{k-1}$ (χ^2 with k-1 degrees of freedom) if each $n_i \geq 5$.

3.7.7. Comparison test

Suppose Y_{1i} and Y_{2i} be random variables representing a measure of performance taken from two models (Kleijnen, 1995). Suppose that n days are simulated, so i = 1,2, ..., n. Then let μ_{1i} and μ_{2i} be means of those random variables. To compare the effectiveness of the two models, there are are n paired (correlated) differences $d_i = \mu_{1i} - \mu_{2i}$ and $\bar{d}$ be the point estimator of d. If the samples are normally and independently distributed, the Student t test can be used to examine the difference in response $\bar{d}$ whose mean δ and variance s^2_d. Suppose the null hypothesis is H_o: $\delta = 0$. Then the test statistic becomes

$$t_{n-1} = \frac{\bar{d} - \delta}{s_d / \sqrt{n}} \tag{30}$$

where $\bar{d}$ denotes the average of the n d's, δ is the expected value of d, and s_d represents the estimated standard deviation of d.

In the case of a non-significance when the null hypothesis is accepted, the conclusion is that there is no significant differences in the performance between the two models. If the t_{n-1} value is significant, the average performance of the first model per day deviates significantly from the second model.

3.8. Summary

Controlling the flow of material on a shop floor involves executing a master production schedule to meet customer due date requirements while attempting to keep WIP inventory low. Three scenarios have been developed, to test dispatching rules, to test two expediting rules, and to examine the effect of variance reduction on shop performance. A simulation model has been built as a means to implement a DBR environment in a hypothetical job shop. The next chapter deals with the simulation results, including analysis and interpretation.

Chapter 4
Simulation Results and Analyses

4.1 Introduction

This chapter discusses the simulation results and analyses. Before any analysis is conducted, both model verification and validation are outlined. Model verification tests that the simulation model performs as was planned in the conceptual model. In the verification stage, the value of several constant parameters used in the simulation runs are determined. These parameters include the job mix, release rule, machine utilisation, buffer multipliers and due date settings. This kind of model verification is termed validating the input-output transformation (Banks and Carson, 1984). The hypothetical job shop is treated as a function that transforms certain input parameters into output performance measures.

To validate the model, tests are carried out to confirm that the model performs as intended. Most hypothetical job shops are derived on the assumption that certain specified sets of conditions are satisfied. For example, the hypothetical shop used in this study is a non-terminating, steady state simulation. Hence the long-run or steady state behavior of the system should be independent of the initial conditions. The validation stage includes a test for steady state condition, an autocorrelation test, a data sufficiency test, a normality test and a homogeneity variance test. The results of the experiments presented help to select appropriate values for the control variables that effect the shop performance, based on statistical tests on the inventory and due date performance measures.

Section 4.2 presents the analysis of the dispatching rules (scenario 1). Section 4.3 describes a statistical analysis for two expediting rules (scenario 2). Section 4.4 reports the analysis for three levels of variance reductions (scenario 3). Finally, the main points drawn from the analysis are summarised.

4.2. Scenario 1: Analysis of Dispatching rules

This section assesses the impact that First Come First Served (FCFS), Short Processing Time (SPT) and Minimum Slack Time (MINSLK) have on the shop floor performance under a DBR methodology.

4.2.1. Verification and validation

Before conducting final output data analysis, the simulation program has to be verified to be working as planned. For example a small deterministic problem, in which the processing times are constant, was used as an input to the program and the results are compared to those obtained by manual calculation. It was found that the dispatching rule and due date setting performed as planned.

In this verification phase, the experimental parameters were tested. The product mix and the machine utilisation needed to be tested statistically. Product mix is assumed to be uniformly distributed. The machine utilisation was determined based on the level of protective capacity. To do this, a simulation program with FCFS dispatching rule was run over 2,000 simulated days. Table 4.1 shows that the actual product mix for this program with ten percent mean protective capacity (MPC) and sixteen buffer multiplier is statistically similar to the planned product mix at the five percent level of significance.

Table 4.1. Statistical test for Product Mix 10%

Product number	Product produced	Product Mix (%)	Differences (Δx_i) from 10%	Test statistics
1	9728	10.0084	0.0084	Mean = $\Sigma\, \Delta x_i/n$ = 1.78 E-16
2	9767	10.0485	0.0486	Std = 0.122678
3	9886	10.1709	0.1709	$t = \dfrac{\sum_{i=1}^{n} \Delta x_i / n - 0}{s / \sqrt{n}}$
4	9680	9.9591	-0.0409	
5	9509	9.7831	-0.2169	
6	9601	9.8778	-0.1222	
7	9641	9.9189	-0.0810	= 4.58 E-15
8	9887	10.1720	0.1720	
9	9781	10.0629	0.0629	$t_{1-\alpha/2,n-1} = t_{0.975,9} = 2.262$
10	9718	9.9981	-0.0019	

Table 4.2 shows that the resources for the FCFS dispatching rule perform as expected at a five percent level of significance. The results show that the program has been verified and model validation can now be carried out to test the assumptions of the model.

Table 4.2. Statistical test for machine utilisation

Resource number	Planned capacity	Actual capacity	Differences (Δx_i)	Test statistics
1	0.904	0.847	0.057	Mean = $\Sigma\ \Delta x_i/n$ = 0.005333
2	0.904	0.903	0.001	Std = 0.019875
3	0.904	0.900	0.004	$t = \dfrac{\sum_{i=1}^{n} \Delta x_i\ / n - 0}{s / \sqrt{n}}$
4	0.899	0.896	0.003	
5	0.894	0.898	-0.004	
6	0.894	0.894	0.000	
7	0.904	0.906	-0.002	= 0.805047
8	0.894	0.905	-0.011	
9	1.000	1.000	0.000	$t_{1-\alpha/2,n-1} = t_{0.975,8} = 2.306$

Model validation is the process of ensuring that the model represents the system under study. Model validation involves tests of steady state conditions, autocorrelation, data sufficiency, normality, and homogeneity of variances.

The indicator used to study the steady state condition is the average delay time of parts, that is the time difference between theoretical time to arrive at the constraint buffer and the actual arrival time. The statistical fluctuations of the shop are represented by this average delay time. The estimation of the steady state mean of delay time depends on the warm-up period and is calculated when the initial transient period is judged to be over. Thus data is collected after the warmup period to estimate the steady state mean of delay time. Figure 4.1 graphs the test of steady state condition for the FCFS rule, a buffer multiplier of 16, and ten percent mean protective capacity. However, this graph cannot provide any information to determine when the system has reached the steady state condition. For this purpose, Welch's procedure is adopted, as it provides a graphical way of selecting the warmup period length (Law and Kelton, 1991). The graph in Figure 4.2 shows that the warmup period (identified by the arrow) is approximately fifty simulation days. Any fluctuations that occur after fifty days can be assumed to be caused by the random nature of the system.

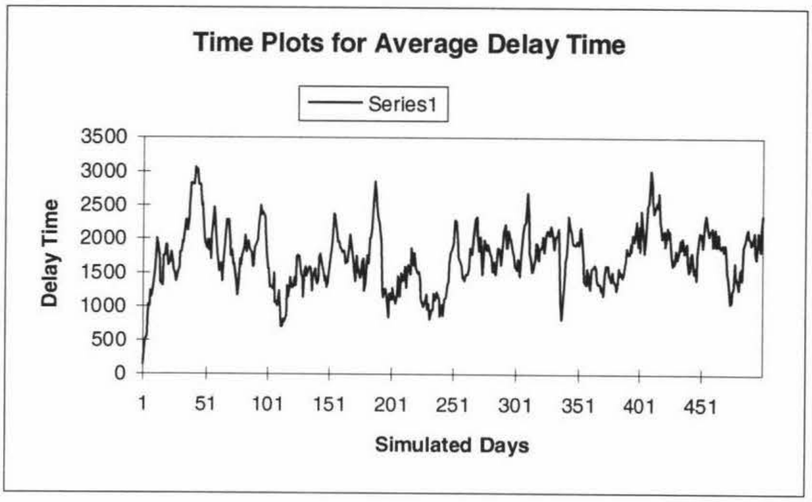


Figure 4.1. Time plots for average delay time

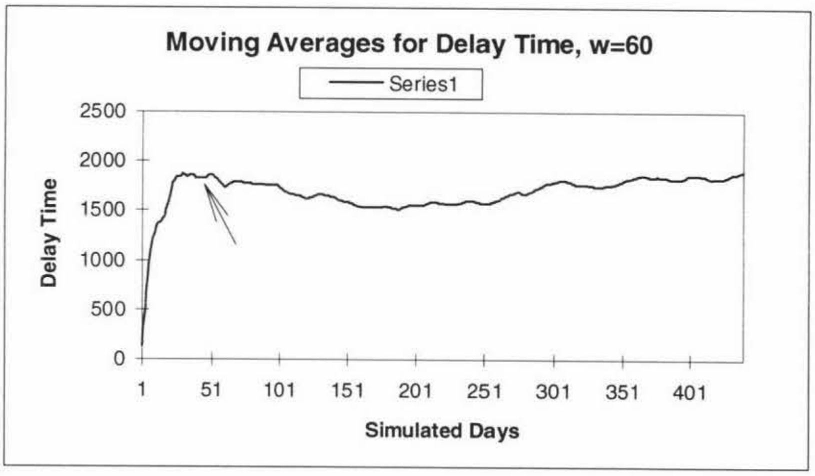


Figure 4.2. Moving averages for delay time with $W = 60$

A statistical test was conducted to show that the model reaches steady state after fifty days. The mean of the first difference between the successive daily average delay time after smoothing in Figure 4.2 should be non-zero before the shop reaches the steady state. This mean should converge to nearly zero after steady state is reached. The test should be performed on the average daily delay time before and after fifty days. The period after the warmup period was chosen as fifty days to give a balanced comparison. The null hypothesis to be tested was, H_o: $\Sigma \Delta y_i/n = 0$ for both periods. The test statistic is based on the Z-test with the standard deviation estimated from the sample data. The one tailed alternative hypothesis is given by $\Sigma\ \Delta y_i/n > 0$. The five percent tabulated value of z is 1.6449. The results of statistical tests in Table 4.3, show that the system reaches steady state after fifty days, because the first difference of these values is not significantly different from zero. This analysis illustrates that fifty days is sufficient length for the transient period and thereby discarded. For the subsequent analysis, data was collected after sixty days. This data will be uncorrelated. The following test for autocorrelation uses the batch means method.

Table 4.3. The steady state statistical test

Indicator variable	Before steady state	After steady state
Daily average delay time to the constraint	$\Sigma\ \Delta y_i/n = 35.121$ St. dev(s) of $\Sigma\ \Delta y_i/n$ $= 52.977$ $z = 4.641$	$\Sigma\ \Delta y_i/n = -0.603$ St. dev(s) of $\Sigma\ \Delta y_i/n$ $= 6.371$ $z = -0.662$

4.2.2. Autocorrelation test

Setting the experimental parameters such as the buffer constant multiplier need to determine before data is collected. Data should comply the autocorrelation test to ensure the independent observation. Several experiments were conducted to decide the buffer constant multipliers. The selection of buffer multiplier for each experiment depends on the percentage of jobs that are tardy. The model should operate on not more than 10% of job tardiness (Gardiner *et al.*, 1993). If the buffer multiplier is too short, many parts will be absent from the constraint's buffer. The constraint resource would work with the available part following the priority of earliest due date. On the other hand, if the buffer multiplier is too large, the lead time will be too long and congestion will occur.

The pilot experiments suggest that 2,000 simulated days is the basis for the autocorrelation test. If each observation is taken daily, a simulation run of 2,000 observations can be divided into 15 contiguous batches, each consisting of 128 daily observations. The 15 batched means can be used in an autocorrelation test. Since the first subrun (128 days) contains the transient period (50 days), this observation was discarded (Fishman, 1978). A similar judgment also applied to other number of batches. An example in Figure 4.3 shows that the FCFS dispatching rule with the 20% MPC gives best results with buffer multiplier of 8. This is because this buffer multiplier gives the best estimate for the stable percent jobs tardy, that is the shop recovers from parts being delayed before they reach the constraint resource. As can be seen in Figure 4.3, the FCFS dispatching rule with buffer multilier of 8 produces the percent jobs tardy which goes down to absorb the fluctuation occured at the 13*th* batch.

The planning factors for buffer multipliers were different depending on the planned mean protective capacity (MPC). The results of pilot experiments proposed that the buffer multipliers for the FCFS dispatching rule at the 10%, 20%, 30% and 40% MPCs were 16, 8, 6, and 5 respectively. Since the performance of FCFS will be the benchmark, SPT and MINSLK dispatching rules will use the same buffer multipliers. It can be seen that the 10% MPC needs almost twice the buffer multiplier of the 20% MPC. However, the constant buffer multipliers for 30% and 40% MPC are not much different. Clearly the model is more sensitive when there is limited protective capacity.

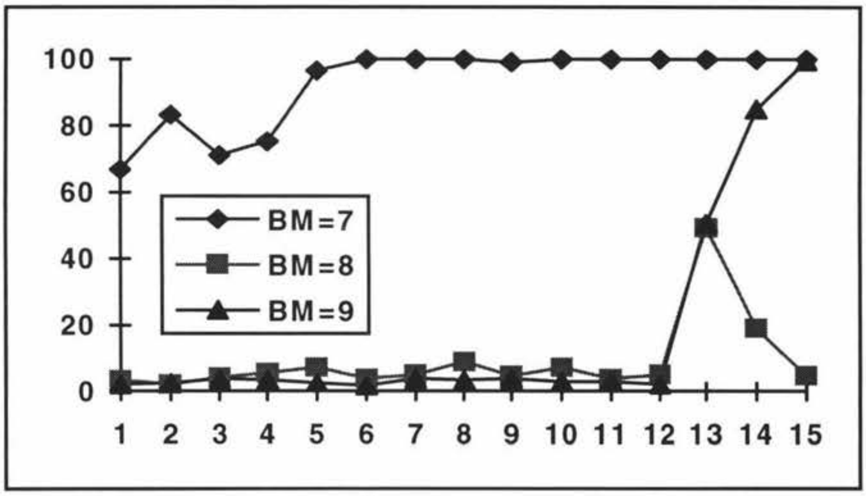


Figure 4.3. Percentage of tardy jobs for FCFS with MPC=20%

In this experiment, the level of significance for the autocorrelation test is set at five percent and H_o is not rejected if the absolute value of Z_k is less than 1.645. Tables 4.4-4.6 give the calculation of the Z_k statistic as a function of dispatching rules and increasing number of batches. The calculations of Z_k values indicate that the performance indicators for the three dispatching rules with 10% MPC were not significantly autocorrelated at the 5% level of significance, after each datum was collected over 128 simulated days.

Table 4.4. Calculation of Z_k statistic for the FCFS dispatching rule

Number of batches (k)	Number of periods of obs. per batch (m)	Z_k value of performance indicators			
		Delay time	Cycle time	Queue length	Percent tardy
1872	1	37.765	0.315	39.529	37.558
936	2	24.522	1.118	25.638	26.135
468	4	14.723	0.380	15.185	17.509
234	8	7.042	1.484	7.193	11.889
117	16	2.905	0.457	2.939	8.309
58	32	0.986	-0.475	0.935	5.317
29	64	0.397	1.282	0.309	2.243
14	128	1.214	0.376	1.268	0.733

Table 4.5. Calculation of Z_k statistic for the SPT dispatching rule

Number of batches (k)	Number of periods of obs. per batch (m)	Z_k value of performance indicators			
		Delay time	Cycle time	Queue length	Percent tardy
1872	1	14.052	7.769	40.862	11.187
936	2	12.832	2.842	27.510	8.264
468	4	9.667	0.951	17.707	6.609
234	8	7.621	0.246	10.664	5.423
117	16	4.856	-0.145	5.806	3.828
58	32	1.841	-1.116	2.066	0.990
29	64	1.404	0.577	1.570	0.100
14	128	1.051	1.562	1.206	0.809

Table 4.6. Calculation of Z_k statistic for the MINSLK dispatching rule

Number of batches (k)	No. of periods of obs. per batch (m)	Z_k value of performance indicators			
		Delay time	Cycle time	Queue length	Percent tardy
1872	1	8.447	10.084	37.514	4.424
936	2	6.0517	4.617	24.116	0.962
468	4	4.414	-1.106	13.708	3.816
234	8	2.239	-5.067	6.786	1.255
117	16	2.991	-4.274	3.331	1.993
58	32	3.411	-3.294	1.309	2.229
29	64	1.415	-0.309	1.354	1.603
14	128	0.672	-0.521	0.178	1.151

Table 4.7 exhibits the autocorrelation test for the simulation outputs with the various MPCs. Three Z_k values of performance indicators for MINSLK dispatching rule with 20% MPC were more than the critical Z value. This unstationary result may be caused by the presence of the initialisation bias. After deleting four initial observations of the 20% MPC, the performance indicators did not show any autocorrelation. The similar situation was also applied for MINSLK dispatching rule with 30% MPC but only deleting two initial observations. The next analysis uses the sample size (k = 10, 12, and 14) obtained from this analysis.

Table 4.7. Calculation of Z_k statistic for the various levels of percent MPCs

Percent MPC	Dispatching Rules	Z_k value of performance indicators (k=14)			
		Delay time	Cycle time	Queue length	Percent tardy
20%	FCFS	0.363	0.540	0.367	0.738
	SPT	-0.865	-0.638	-1.172	0.101
	MINSLK	0.578 1.058 (k=10)	3.464 0.740 (k=10)	2.293 1.115 (k=10)	3.263 -0.021 (k=10)
30%	FCFS	0.934	0.885	0.906	0.872
	SPT	0.319	0.776	0.529	0.811
	MINSLK	0.641 0.715 (k=12)	3.038 0.970 (k=12)	3.050 0.721 (k=12)	3.219 0.364 (k=12)
40%	FCFS	0.467	1.566	0.392	0.873
	SPT	0.537	1.626	0.807	0.865
	MINSLK	-0.064	1.488	1.109	0.669

4.2.3. Data sufficiency test

A data sufficiency test was performed to determine that the available data was adequately represented the behaviour of the shop. To determine the sample size, the following procedure was adopted from Law and Kelton (1991). The performance statistics (average cycle time, delay time, queue length, and percent jobs tardy) were collected following the warmup period. The number of samples depends on the result of the autocorrelation test, which produces the pilot runs or epochs n ($n = k$ = 10, 12, or 14). If the runs are not autocorrelated, then the observations can be treated as if they were *IID* normal random variables.

Since a compact confidence interval is needed, a relatively small value (0.15) for the relative half confidence was selected for analysis. Table 4.8 shows that the data sample size was adequate for all dispatching rules and mean protective capacities. The only exception to this was some of the data of the average percentage of tardy jobs statistic. The extra runs to collect data for the percent of jobs tardy could not be performed. There are two reasons. Firstly, the nature of the fluctuations occur in the shop cause high variability in percent jobs tardy. The second reason is the consideration of simulation costs. Alternatively, the analysis of simulation output for the percent of jobs tardy will use a nonparametric test (Conover, 1980).

Table 4.8. Test for data sufficiency in the selected simulation output

Relative Half Confidence Interval (RHCI) for FCFS

Percent MPCs	The performance statistics (n = 14)			
	Delay time	Cycle time	Queue length	Percent tardy
10%	0.0421	0.0022	0.0432	0.8369
20%	0.0401	0.0025	0.0355	0.7523
30%	0.0257	0.0027	0.0125	1.2057
40%	0.0192	0.0029	0.0069	1.4552

Relative Half Confidence Interval (RHCI) for SPT

Percent MPCs	The performance statistics (n = 14)			
	Delay time	Cycle time	Queue length	Percent tardy
10%	0.0596	0.0069	0.0567	0.0709
20%	0.0276	0.0023	0.0208	0.4793
30%	0.0211	0.0026	0.0093	1.1587
40%	0.0216	0.0028	0.0061	1.4296

Relative Half Confidence Interval (RHCI) for MINSLK

Percent MPCs	The performance statistics (n = 14)			
	Delay time	Cycle time	Queue length	Percent tardy
10%	0.0634	0.0047	0.0591	0.0646
20%*	0.0365	0.0027	0.0231	0.0579
30%**	0.0254	0.0026	0.0118	0.0193
40%	0.0223	0.0029	0.0085	0.8228

* n = 10, ** n = 12

4.2.4. Normality test

For the statistical tests for the performance of dispatching rules to be carried out, the collected data should satisfy normality and homogeneity of variances. The null hypothesis asserts that *F(y)* is a normal distribution function with unspecified mean and variance, whereas the alternative hypothesis is that *F(y)* is non-normal distribution. The Shapiro-Wilk's test is adopted to examine the performance statistics for normality (Conover, 1980).

Table 4.9 shows the calculated values that are greater than the tabulated values for all simulation runs. As a result, it can be assumed that the simulations output from the system that were tested followed the normal distribution.

Table 4.9. Test of normality for the distribution of performance statistics

Percent MPCs	Dispatch-ing rules	Shop performance statistics					
		Delay time		Cycle time		Queue length	
		Shapiro Wilk's W[1]	Critical Value[2]	Shapiro Wilk's W[1]	Critical Value[2]	Shapiro Wilk's W[1]	Critical Value[2]
10%	FCFS	0.9752	0.825	0.9069	0.825	0.9726	0.825
	SPT	0.9255	0.825	0.8724	0.825	0.9068	0.825
	MINSLK	0.9001	0.825	0.9566	0.825	0.8627	0.825
20%	FCFS	0.9644	0.825	0.9547	0.825	0.9568	0.825
	SPT	0.8879	0.825	0.8723	0.825	0.8616	0.825
	MINSLK	0.9759	0.781	0.9047	0.781	0.9861	0.781
30%	FCFS	0.9819	0.825	0.9325	0.825	0.9814	0.825
	SPT	0.9363	0.825	0.9198	0.825	0.9351	0.825
	MINSLK	0.8273	0.805	0.9016	0.805	0.8468	0.805
40%	FCFS	0.9563	0.825	0.9725	0.825	0.9474	0.825
	SPT	0.9631	0.825	0.9710	0.825	0.9359	0.825
	MINSLK	0.9741	0.825	0.9817	0.825	0.9491	0.825

[1] Compared to the critical value (1%) if the Shapiro-Wilk's *W* is larger, it leads to acceptance of the null hypothesis in support of normality.

[2] Obtained from Table A18 Quantiles of the Shapiro-Wilk's Test Statistic, in *Practical Nonparametric Statistics* by W.J. Conover (1980), pp. 468-469.

4.2.5. Test for homogeneity of variances

The results from Bartlett's test in Table 4.10 show that only Bartlett's test for homogeneity of variances for the average cycle time with 10% mean protective capacity was rejected.

Table 4.10. Results of Bartlett's tests for homogeneity of variances

Percent MPCs	Shop performance measures					
	Average delay time		Average cycle time		Average queue length	
	Test statistic's	p-value	Test statistic's	p-value	Test statistic's	p-value
10%	1.337	0.513	13.383	0.001	2.579	0.275
20%	4.252	0.119	0.145	0.930	2.767	0.251
30%	1.603	0.472	0.017	0.991	0.713	0.700
40%	0.603	0.740	0.022	0.989	1.242	0.537

Since Bartlett's test is sensitive to the assumption of normality, the test statistics for the cycle time with the 10% mean protective capacity rejected the hypothesis that the variances are homogeneous at $\alpha = 0.001$. Therefore, Levene's test was carried out to confirm the result of Bartlett's test (Levene, 1960).

Using Levene's procedure, the test statistic L for the average cycle time with the 10% MPC was 2.148 and the p-value was 0.130. There was sufficient evidence to conclude that the data was from a normal distribution at the level of significance 0.01 percent. As a result, the performance statistics that include delay time, cycle time, and queue length are meaningful to be used in an analysis of variance.

4.2.6. One way analysis of variances

The analysis of variance of the simulation results is presented in Tables 4.11-4.14. If the null hypothesis is rejected, it is concluded that at least one of three dispatching rules is significantly different from the others. The statistical techniques that permit further comparisons to be made between treatment means are referred to as multiple comparison methods. Multiple comparisons tests were made using Tukey's Studentized Range Test to compare all pairs of treatment means (Steel and Torrie, 1980).

Table 4.11. Analysis of Variance for Dispatching Rules with the 10% MPC

Analysis of Variance on *Delay Time*					
Source	*DF*	*SS*	*MS*	*F*	*p*
Model	2	941138	470569	20.11	0.000
Error	39	912414	23395		
Total	41	1853552			

Analysis for Comparing Dispatching Rules: Delay Time Results				
Level	*N*	*Mean*	*StDev*	*Tukey Grouping**
FCFS	14	1782.2	129.8	a
SPT	14	1416.8	146.2	c
MINSLK	14	1626.1	178.7	b

* Means with the same letter are not significantly different

Analysis of Variance on *Cycle Time*					
Source	DF	SS	MS	F	p
Model	2	142681	71340	73.86	0.000
Error	39	37671	966		
Total	41	180352			

Analysis for Comparing Dispatching Rules: Cycle Time Results				
Level	N	Mean	StDev	Tukey Grouping*
FCFS	14	3654.0	13.8	a
SPT	14	3529.8	42.5	b
MINSLK	14	3653.0	29.9	a

* Means with the same letter are not significantly different

Analysis of Variance on *Queue Length*					
Source	DF	SS	MS	F	p
Model	2	511.3	255.6	8.16	0.001
Error	39	1222.2	31.3		
Total	41	1733.4			

Analysis for Comparing Dispatching Rules: Queue Length Results				
Level	N	Mean	StDev	Tukey Grouping*
FCFS	14	54.808	4.103	b
SPT	14	63.067	6.191	a
MINSLK	14	60.839	6.233	a

* Means with the same letter are not significantly different

Table 4.12. Analysis of Variance for Dispatching Rules with the 20% MPC

Analysis of Variance on *Delay Time*					
Source	DF	SS	MS	F	p
Model	2	143733	71866	43.65	0.000
Error	35	57623	1646		
Total	37	201356			

Analysis for Comparing Dispatching Rules: Delay Time Results				
Level	N	Mean	StDev	Tukey Grouping*
FCFS	14	746.49	51.79	a
SPT	14	607.25	29.00	b
MINSLK	10	709.84	36.24	a

* Means with the same letter are not significantly different

Analysis of Variance on *Cycle Time*					
Source	DF	SS	MS	F	p
Model	2	153091.5	76545.7	1539.85	0.000
Error	35	1739.8	49.7		
Total	37	154831.3			

Analysis for Comparing Dispatching Rules: Cycle Time Results				
Level	N	Mean	StDev	Tukey Grouping*
FCFS	14	1742.57	7.44	b
SPT	14	1713.23	6.68	c
MINSLK	10	1869.18	6.99	a

* Means with the same letter are not significantly different

Analysis of Variance on *Queue Length*					
Source	DF	SS	MS	F	p
Model	2	235.46	117.73	70.31	0.000
Error	35	58.60	1.67		
Total	37	294.07			

Analysis for Comparing Dispatching Rules: Queue Length Results				
Level	N	Mean	StDev	Tukey Grouping*
FCFS	14	26.147	1.607	c
SPT	14	30.029	1.083	b
MINSLK	10	32.288	1.043	a

* Means with the same letter are not significantly different

Table 4.13. Analysis of Variance for Dispatching Rules with the 30% MPC

Analysis of Variance on *Delay Time*					
Source	DF	SS	MS	F	p
Model	2	27593	13796	66.02	0.000
Error	37	7732	209		
Total	39	35325			

Analysis for Comparing Dispatching Rules: Delay Time Results				
Level	N	Mean	StDev	Tukey Grouping*
FCFS	14	369.02	16.40	a
SPT	14	317.41	11.62	b
MINSLK	12	375.86	15.01	a

* Means with the same letter are not significantly different

Analysis of Variance on *Cycle Time*					
Source	DF	SS	MS	F	p
Model	2	84522.9	42261.5	1439.96	0.000
Error	37	1085.9	29.3		
Total	39	85608.8			

Analysis for Comparing Dispatching Rules: Cycle Time Results				
Level	N	Mean	StDev	Tukey Grouping*
FCFS	14	1188.57	5.53	b
SPT	14	1184.05	5.38	b
MINSLK	12	1286.54	5.33	a

* Means with the same letter are not significantly different

Analysis of Variance on *Queue Length*					
Source	DF	SS	MS	F	p
Model	2	87.551	43.776	236.29	0.000
Error	37	6.855	0.185		
Total	39	94.406			

Analysis for Comparing Dispatching Rules: Queue Length Results				
Level	N	Mean	StDev	Tukey Grouping*
FCFS	14	21.125	0.459	c
SPT	14	22.895	0.371	b
MINSLK	12	24.805	0.460	a

* Means with the same letter are not significantly different

Table 4.14. Analysis of Variance for Dispatching Rules with the 40% MPC

Analysis of Variance on *Delay Time*					
Source	DF	SS	MS	F	p
Model	2	6096.5	3048.2	52.68	0.000
Error	39	2256.6	57.9		
Total	41	8353.0			

Analysis for Comparing Dispatching Rules: Delay Time Results				
Level	N	Mean	StDev	Tukey Grouping*
FCFS	14	210.55	7.00	b
SPT	14	191.78	7.19	c
MINSLK	14	220.89	8.54	a

* Means with the same letter are not significantly different

Analysis of Variance on *Cycle Time*					
Source	DF	SS	MS	F	p
Model	2	27.7	13.9	0.73	0.489
Error	39	741.8	19.0		
Total	41	769.6			

Analysis for Comparing Dispatching Rules: Cycle Time Results				
Level	N	Mean	StDev	Tukey Grouping*
FCFS	14	884.18	4.41	a
SPT	14	883.34	4.26	a
MINSLK	14	885.32	4.42	a

* Means with the same letter are not significantly different

Analysis of Variance on *Queue Length*					
Source	DF	SS	MS	F	p
Model	2	4.4713	2.2357	46.70	0.000
Error	39	1.8671	0.0479		
Total	41	6.3384			

Analysis for Comparing Dispatching Rules: Queue Length Results				
Level	N	Mean	StDev	Tukey Grouping*
FCFS	14	17.127	0.207	b
SPT	14	17.886	0.189	a
MINSLK	14	17.291	0.255	b

* Means with the same letter are not significantly different

The practical interpretation of the Tukey groupings that appear in Tables 4.11-4.14 is as follows. The dispatching rules with the same letter implies that the means of the pairwise comparisons are not significantly different. The error rate of $\alpha = 5\%$ applies to the family of all paired comparisons. In Table 4.11 (MPC = 10%), the three dispatching rules demonstrate different performance in delay time. SPT outperforms FCFS and MINSLK. SPT also gives a lower cycle time compared to FCFS and MINSLK. On the other hand, FCFS yields lower queue length in front of the constraint resource.

In Table 4.12 (MPC = 20%), it can be seen that SPT results in a lower delay time. SPT is also more effective in producing a lower cycle time. In the queue length performance measure, MINSLK outperforms other dispatching rules.

For the 30% MPC in Table 4.13, SPT is significantly different from FCFS and MINSLK at $\alpha = 5\%$ in delay time and cycle time. For the queue length performance measure, FCFS is more effective than SPT and MINSLK at $\alpha = 5\%$.

Table 4.14 shows that SPT still performs better in the delay time performance measure than FCFS and MINSLK. The dispatching rules FCFS, SPT and MINSLK have the same letter for cycle time, so the evidence is that the three populations involved cannot be distinguished. Finally, FCFS and MINSLK result in better queue length performance results.

Since the distribution of percentage of tardy jobs is not normally distributed, the non-parametric test for analysis of variance is adopted. The problem is to compare three different dispatching rules based on the percentage of tardy jobs. The null and alternative hypotheses are H_o: the three dispatching rules are equivalent and H_a: at least one dispatching rule is different from the others. Table 4.15 shows the calculation of Kruskal-Wallis tests for all levels of mean protective capacity. Using $\alpha = 0.05$, the rejection region is $H \geq 5.991$ (from χ^2_2). Therefore H_o is rejected for all cases and it is concluded that the three dispatching rules are not equally effective in yielding the percentage of tardy jobs.

Table 4.15. Kruskal-Wallis tests for dispatching rules on the percent jobs tardy

Kruskal-Wallis Test for the 10% MPC				
LEVEL	NOBS	MEDIAN	AVE. RANK	Multiple Comparisons*
FCFS	14	4.035	11.4	c
SPT	14	8.330	19.6	b
MINSLK	14	11.575	33.5	a
OVERALL	42		21.5	
H = 23.18	d.f. = 2	p = 0.000		

Kruskal-Wallis Test for the 20% MPC				
LEVEL	NOBS	MEDIAN	AVE. RANK	Multiple Comparisons*
FCFS	14	5.100	13.1	b
SPT	14	5.500	15.9	b
MINSLK	10	62.055	33.5	a
OVERALL	38		19.5	
H = 22.00	d.f. = 2	p = 0.000		

Kruskal-Wallis Test for the 30% MPC				
LEVEL	NOBS	MEDIAN	AVE. RANK	Multiple Comparisons*
FCFS	14	2.230	13.9	b
SPT	14	2.200	15.1	b
MINSLK	12	87.270	34.5	a
OVERALL	40		20.5	
H = 24.65	d.f. = 2	p = 0.000		
H = 24.65	d.f. = 2	p = 0.000 (adjusted for ties)		

Kruskal-Wallis Test for the 40% MPC				
LEVEL	NOBS	MEDIAN	AVE. RANK	Multiple Comparisons*
FCFS	14	0.7900	14.9	b
SPT	14	0.9200	18.0	b
MINSLK	14	4.4800	31.6	a
OVERALL	42		21.5	
H = 14.81	d.f. = 2	p = 0.001		
H = 14.82	d.f. = 2	p = 0.001 (adjusted for ties)		

* Medians with the same letter are not significantly different.

In a multiple comparison column in Table 4.15, average ranks followed by the same letter are not significantly different at the 5% level of significance. For 10% MPC, FCFS is significantly different from SPT and MINSLK. However, FCFS and SPT are not different in performance when compared to MINSLK for the remaining MPCs. In other words, the use of FCFS or SPT is favourable to MINSLK in this hypothetical job shop, to improve the due date performance, because FCFS and SPT consistently produce the lower median of percent jobs tardy.

4.2.7. Interpretation of overall results

Tables 4.11-4.14 include the analysis of variance of the inventory performance measures and multiple comparison tests. Only the cycle time for dispatching rules with the 40% MPC are not significantly different at α = 5%. The results of comparative analysis of dispatching rules reveal that SPT is consistently better (lower delay time and lower cycle time), whereas FCFS performs better in providing a lower queue length. Moreover, the proportion of tardy jobs under FCFS and SPT is much lower than under MINSLK (see Table 4.15). The overall results in Table 4.16 appear that SPT and FCFS procedures are more favorable than MINSLK. The reason is attributable to the fact that in this hypothetical job shop there are the uniform load distribution for the nonconstraint resources and the nonconstraint resources have large processing time variations. In this situation, SPT performs better at minimising the flow time of parts. However, the shorter flow time leads to build up of more queue parts in front of the constraint resource. This result is consistent with the prior research (Conway *et al.*, 1967; Blackstone *et al.*, 1982).

Table 4.16. Summarised analysis results for dispatching rules

Percent MPCs	Shop performance measure			
	Delay Time	Cycle Time	Queue Length	Percent Tardy
10%	0.000	0.000	0.001	0.000
20%	0.000	0.000	0.000	0.000
30%	0.000	0.000	0.000	0.000
40%	0.000	0.489*	0.000	0.001
Favourable dispatching rule	SPT	SPT	FCFS	SPT and FCFS

* Shading area indicates not significant p-values at the 5% experimentwise error rate.

4.3. Scenario 2: Analysis of Expediting Rules

This section presents an analysis of the expediting rules. The question to be answered by this analysis can be stated as "is there a difference in the performance of expediting rules?". There are six performance measures used in this experiment: delay time to the constraint, cycle time, queue length in front of the constraint, percentage of tardy jobs and frequency of expediting. Frequency of expediting is the total number of parts requiring special priority in order to arrive at the buffer. To answer the research question, the paired comparison tests will be used to test the effects of two expediting rules on shop performance (Kleijnen, 1995). Before the data was collected, the simulation model was tested for steady state conditions. Initial simulation runs were used to collect data to be used in the statistical tests.

Verification tests for simulation models were conducted. These included program debugging, test for resource utilisation and a test for expediting rule mechanism. The test results indicated the program conformed to the desired behaviour. The models can thus be accepted as a realistic presentation of the hypothetical job shop.

Since the models are of a discrete event, non-terminating system, a test for steady state condition must be carried out. A modification of Welch's procedure is used to delete the transient values before the steady state conditions (Law and Kelton, 1991). To determine where the transient phase ends, delay time is used as an indicator of steady state. The behaviour of the model is well represented by considering the fluctuation in delay time. For example, averaged delay times of the shop with the dynamic expediting rule and FCFS dispatching rule at the buffer multiplier of 16 are recorded and plotted against the daily observation number (see Figure 4.4). Using Welch's procedure, several arbitrary values of moving average window, w, were used to smooth out the high-frequency oscillations in averaged delay times. A moving average with $w = 60$ provides a satisfactory graph to determine when the system reaches the steady state. From the graph in Figure 4.5, it can be estimated that the transient phase ends after 30 simulated days. The t test was also used to confirm this warm-up period length.

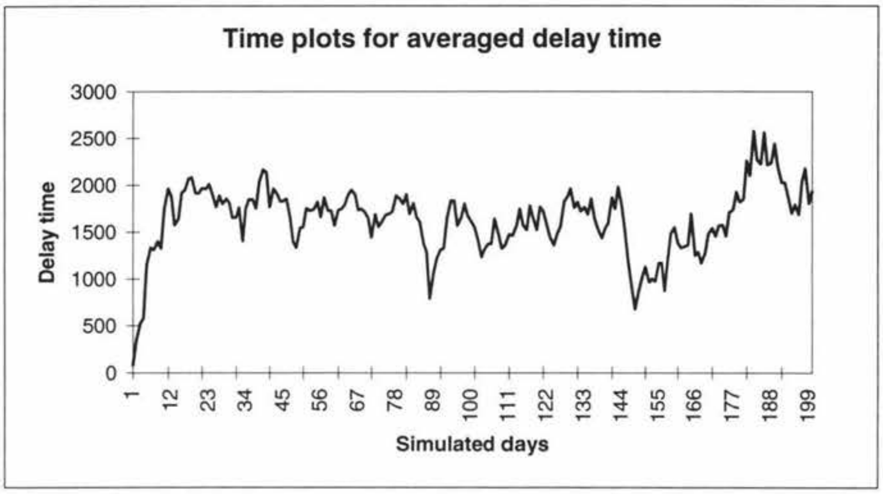


Figure 4.4. Time plots for averaged delay time

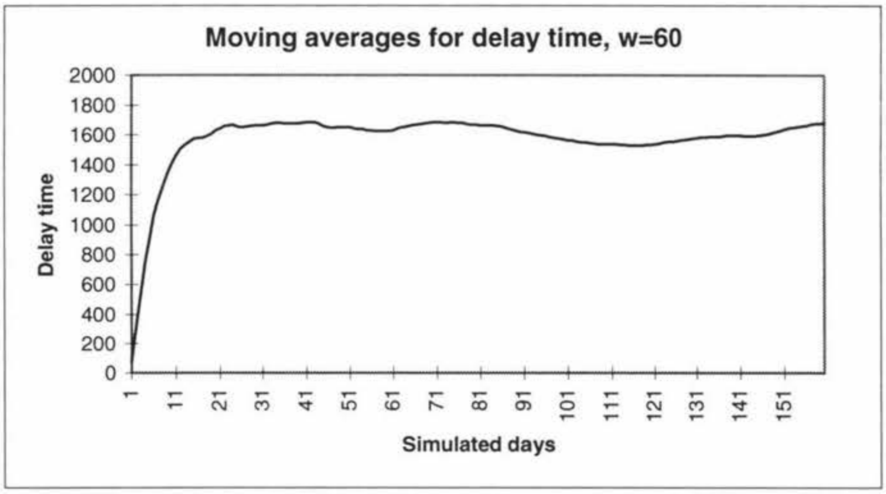


Figure 4.5. Moving averages for delay time with $W = 60$

The five percent tabulated value of t for the one-tailed alternative hypothesis mean $\Sigma\ \Delta y_i/n > 0$ with $(n\text{-}1) = 28$ df is 1.70113. The t values of the difference between the successive daily moving averaged delay time before steady state and after steady state is given in Table 4.17. The test results show that the transient period is approximately 30 simulated days at the five percent level of significance, because the first difference of the values after steady state is significantly different from zero. Therefore, the required data for the comparison tests will be recorded after 30 simulated days.

Table 4.17. The t test for steady state conditions

Indicator variable	*Before steady state (n=29)*	*After steady state (n = 29)*
Moving average delay time to the resource constraint	Mean Δy_i = 54.819 St. dev (s) of Δy_i = 74.335 t = 3.9713	Mean Δy_i = -1.3347 St. dev (s) of Δy_i = 4.9713 t = -1.4452

A series of simulation runs were made to choose suitable values for the buffer multipliers. The buffer multiplier should be large enough to minimise the percentage of tardy jobs while keeping the cycle time as low as possible. Data on percentage of tardy jobs for the simulation model using the FCFS dispatching rules and MPC = 10% was collected after steady state conditions. A test for data sufficiency for the buffer multiplier equal to 15 provides the suitable number of data to allow the effect of buffer multipliers on percent jobs tardy to be observed. The average of delay time is 4.275 and the standard deviation is 4.7034. Using Kleijnen's formula with $c = 0.2\bar{x}$, the required sample size is 87.36352 or 88 observations.

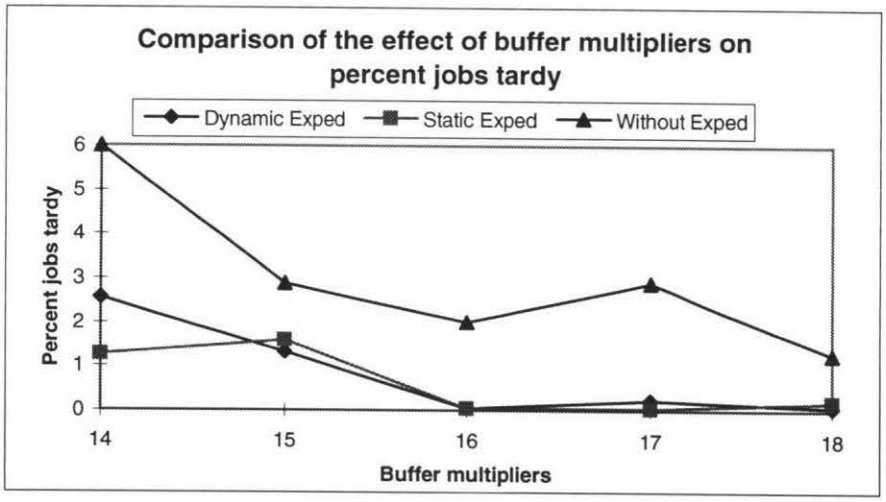


Figure 4.6. Comparison of the effect of buffer multipliers on percent jobs tardy

Figure 4.6 compares the effect of buffer multipliers on percent jobs tardy for three simulation models: dynamic expediting, static expediting and without expediting. The buffer multiplier of 16 gives a relatively lower percentage of tardy jobs for the three models. It is unnecessary to increase buffer multipliers to more than 16 because of increasing the cycle time. Hence the comparison for MPC = 10% will use the buffer multiplier of 16 as the benchmark. This procedure was also employed to determine the value of the buffer multipliers for the remaining values of MPC. The MPCs of 20%, 30% and 40% have buffer multipliers of 8, 5 and 4 respectively.

The simulation models operate under a combination of selected control variables such as MPCs, buffer multipliers, dispatching rules and expediting rules. The data was collected to enable inferences to be drawn about the relationship between the control variables and shop performance. The following sub-sections provide an analysis of the simulation results.

4.3.1. Comparison of the dynamic and static expediting rules under FCFS with various levels of MPCs

To test the effectiveness of dynamic and static expediting rules on shop performance, a number of runs are required to obtain a specified level of precision. Kleijnen (1987) suggests a data sufficiency formula to determine the number of runs required for comparison tests. The constant half-width of confidence interval c is specified as a fraction of the quantity to be estimated. In the comparison test, the constant c is a fraction of statistic $\overline{d}$, that is the average of n paired differences d_i. A larger value of c requires fewer runs to estimate the statistics of interest. Kleijnen (1987) recommends the constant c equal to $0.20\overline{d}$. The value of $c = 0.50\overline{d}$ was required in this experiment, to give a reasonable number of additional runs.

Let w_i and v_i denote the performance measure on simulated day i under dynamic expediting and static expediting respectively. Let there are n paired differences $d_i = w_i - v_i$, which are normally distributed. If δ is the expected value of d, the null hypothesis is H_o: $\delta = 0$. If H_o is accepted, it means that the performance of the two expediting styles are the same. The performance of the expediting styles is statistically different if $| t_{n-1}| > t_{1-\alpha;n-1}$ because the simulation models have different performance values each day. The following comparison tests are subsequently given according to the MPCs of the shops.

Table 4.18 shows the data sufficiency tests for the simulation models of two expediting rules with MPC = 10%, using FCFS. The reasonable sample size required to test the hypothesis is 184 observations. For this comparison test, the simulation models are run for more 154 additional simulated days.

Table 4.18. Tests for the number of required simulation runs

Measures	*Average*	$c = 0.2\overline{d}$	s_o	n_o	n	$n - n_o$
Delay time	-324.065	-64.8129	204.4921	30	29 (28.739)	-1
Cycle time	-3.23067	-0.64613	14.06735	30	1369 (1368.41)	1339
Queue length	11.3	2.226	5.829685	30	20 (19.801)	-10
Percent tardy	-3.14733	-0.62947	5.02034	30	184 (183.636)	154
Exped number	-19.9667	-3.99333	27.32876	30	136 (135.209)	106

The comparison tests for the static expediting rule and the dynamic expediting rule with MPC = 20% are as follows.

(1) Delay time to the buffer.

H_o: There is no significant difference between the two expediting styles.

H_a: There is a significant difference between the two expediting styles.

The average $\overline{d}$ is -41.055. The estimated standard deviation s_o is 247.569. The sample size n is 184. The calculated t statistic is -2.249. The tabulated $t_{0.95;183}$ value is 1.6532. Since the calculated $|t_{n-1}|$ statistic is more than $t_{1-\alpha;n-1}$, the null hypothesis is rejected at the 5% level of significance. This result concludes that the model with the static expediting rule has a lower average delay time to the buffer by approximately 41.0554 minutes per simulated day compared to the model with the dynamic expediting rule.

(2) Cycle time

H_o: There is no significant difference between the two expediting styles.

H_a: There is a significant difference between the two expediting styles.

The average $\overline{d}$ is -0.370. The estimated standard deviation s_o is 19.764. The sample size n is 184. The calculated t statistic is -0.254. The tabulated $t_{0.95;183}$ value is 1.6532. Since the calculated $|t_{n-1}|$ statistic is less than $t_{1-\alpha;n-1}$, the null hypothesis is accepted at the 5% level of significance. This result concludes that there is no significant difference in cycle time between the two expediting styles.

(3) Queue length in front of the constraint

H_o: There is no significant difference between the two expediting styles.

H_a: There is a significant difference between the two expediting styles.

The average $\bar{d}$ is 1.409. The estimated standard deviation s_o is 7.873. The sample size n is 184. The calculated t statistic is 2.429. The tabulated $t_{0.95;183}$ value is 1.6532. Since the calculated $|t_{n-1}|$ statistic is more than $t_{1-\alpha;n-1}$, the null hypothesis is rejected at the 5% level of significance. This result concludes that the model with dynamic expediting rule has a lower average queue length by 2 parts compared with the model with static expediting rule.

(4) Percentage of tardy jobs

H_o: There is no significant difference between the two expediting styles.

H_a: There is a significant difference between the two expediting styles.

The average $\bar{d}$ is -0.406. The estimated standard deviation s_o is 247.569. The sample size n is 184. The calculated t statistic is -1.968. The tabulated $t_{0.95;183}$ value is 1.6532. Since the calculated $|t_{n-1}|$ statistic is more than $t_{1-\alpha;n-1}$, the null hypothesis is rejected at the 5% level of significance. This result concludes that the model with static expediting rule has lower average percentage jobs tardy by 0.406% per simulated day compared with the model with dynamic expediting rule.

(5) Number of expediting actions taken

H_o: There is no significant difference between the two expediting styles.

H_a: There is a significant difference between the two expediting styles.

The average $\bar{d}$ is 5.908. The estimated standard deviation s_o is 33.298. The sample size n is 184. The calculated t statistic is 2.407. The tabulated $t_{0.95;183}$ value is 1.6532. Since the calculated $|t_{n-1}|$ statistic is more than $t_{1-\alpha;n-1}$, the null hypothesis is rejected at the 5% level of significance. This result concludes that the model with dynamic expediting rule has higher average expediting numbers by 5.9 parts per simulated day compared with the static expediting rule.

The DBR models with MPC = 20% require 57 simulated days to estimate the average of paired differences. Using $c = 0.5\bar{d}$, the models with MPC = 30% and MPC = 40% need 210 and 240 simulated days respectively to be used in the comparison tests. Table 4.19 shows the overall results of comparison tests between the models with dynamic expediting and static expediting under various MPCs and the FCFS dispatching rule.

For the low mean protective capacity (MPC = 10%) under FCFS, the dynamic expediting rule outperforms the static expediting rule in terms of delay time and percentage jobs tardy. Dynamic expediting results in the least percentage of daily tardy jobs. The delay time for dynamic expediting is also less than for the static expediting rule. However, dynamic expediting produces higher queue lengths and more expediting. The difference in queue length is only 1.4 parts per day. The dynamic expediting rule expedites more frequently. This indicates that the DBR model with the dynamic expediting rule identifies the potentially late parts and expedites successfully. Both buffering styles have the same performance in cycle time.

At the 20% MPC, the model with dynamic expediting results in greater queue lengths and more expediting. These two buffering styles significantly have no differences in delay time, cycle time or percent jobs tardy at 20% of MPC.

The model with 30% of MPC gives a slightly different result. The static expediting rule produces lower delay time and higher queue lengths. The DBR model with dynamic expediting rule needs more time to bring delayed parts to the buffer. The static expediting rule decreases delay time, but at the expense of increased queue length by 0.7 parts per day. The dynamic expediting rule still has higher expediting by 25.48 parts per day.

At the 40% MPC level, the model with the dynamic expediting rule performs better in terms of percent jobs tardy and number of parts needing expediting. There are no differences in delay time, cycle time and queue length for both the expediting rules.

In addition to the above analysis, both buffering styles produce the same performance in cycle time. For companies that emphasise the cycle time, the static expditing rule is preferable to the dynamic expediting rule because it demands less expediting. Dynamic buffering style is favourable in the environment which emphasises delivery performance such as on time delivery.

Table 4.19. Comparison tests for the dynamic and static expediting rules with FCFS

The DBR models with MPC = 10%						
Measures	*Average*	s_o	n	t_{n-1}	$t_{0.95;183}$	*Sig.?*
Delay time	-41.055	247.569	184	-2.249	1.6532	yes
Cycle time	-0.370	19.764	184	-0.254	1.6532	no
Queue length	1.409	7.872	184	2.429	1.6532	yes
Percent jobs tardy	-0.406	2.797	184	-1.969	1.6532	yes
Exped. number	5.908	33.298	184	2.407	1.6532	yes

The DBR models with MPC = 20%						
Measures	*Average*	s_o	n	t_{n-1}	$t_{0.95;56}$	*Sig.?*
Delay time	-19.034	88.274	57	-1.627	1.6725	no
Cycle time	-0.1207	6.002	57	-0.152	1.6725	no
Queue length	0.682	2.751	57	1.871	1.6725	yes
Percent jobs tardy	-0.328	2.236	57	-1.107	1.6725	no
Exped. number	15.579	23.710	57	4.961	1.6725	yes

The DBR models with MPC = 30%						
Measures	*Average*	s_o	n	t_{n-1}	$t_{0.95;209}$	*Sig.?*
Delay time	21.807	89.577	210	3.528	1.6522	yes
Cycle time	-0.045	2.943	210	-0.220	1.6522	no
Queue length	-0.731	2.948	210	-3.592	1.6522	yes
Percent jobs tardy	-0.043	2.516	210	-0.249	1.6522	no
Exped. number	25.481	32.507	210	11.359	1.6522	yes

Table 4.19. Comparison tests for the dynamic and static expediting rules with FCFS (continued)

The DBR models with MPC = 40%						
Measures	*Average*	s_o	n	t_{n-1}	$t_{0.95;241}$	*Sig.?*
Delay time	-5.239	55.650	240	-1.458	1.6512	no
Cycle time	-0.074	2.015	240	-0.569	1.6512	no
Queue length	0.147	1.928	240	1.181	1.6512	no
Percent jobs tardy	-0.239	1.653	240	-2.246	1.6512	yes
Exped. number	15.733	26.121	240	9.331	1.6512	yes

4.3.2. Comparison of the dynamic and static expediting rules under SPT with various levels of MPCs

The performance comparison of two expediting rules under SPT dispatching rule at the 10% MPC uses 186 runs so that the amount of relative error is less than the planning value of $0.2\overline{d}$ as shown in Table 4.20.

Table 4.20. Tests for data sufficiency

Measures	*Average*	$c = 0.2\overline{d}$	s_o	n_o	n	$n - n_o$
Delay time	590.211	118.042	946.875	30	186	156
Cycle time	-92.154	-18.431	709.758	30	4282	5252
Queue length	-22.455	-4.491	6.502	30	7	-23
Percent tardy	7.001	1.400	9.873	30	144	114
Exped number	15.067	3.013	22.582	30	163	133

The same common random numbers were assigned in the simulation models to compare the dynamic expediting rule and the static expediting rule. The analysis will begin with the lowest MPC of 10%.

(1) Delay time to the buffer

The result concludes that there is a significant difference in average delay time between the model with dynamic expediting rule and the model with static expediting rule. The model with the dynamic expediting rule has a higher average amount of delay time by 590.2 minutes per day.

(2) Cycle time

The results show that there is no significance difference in the cycle time for the two models.

(3) Queue length at the constraint

There is a significance difference in average queue length. The model with dynamic expediting rule has a lower average queue length by 22.45 parts per day.

(4) Percent jobs tardy

It is concluded that there is a significant difference in average percentage of tardy jobs. The model using the dynamic expediting rule has a higher percent of tardy jobs by 2.0 percent jobs per day.

(5) Expediting number

It is concluded that there is a significant difference in average number of expediting actions taken. The model with dynamic expediting rule has more expediting by 10.796 parts per day.

Table 4.21 shows the completed analysis of comparison tests for expediting rules under SPT. To calculate the additional observations for the 20%, 30% and 40% MPCs, the models use the planned relative error by $0.5\overline{d}$. The reasonable sample sizes for the 20%, 30% and 40% MPCs are 53, 161 and 82 respectively.

The overall results for the models under SPT are consistent with the model under FCFS. Both buffering styles have no difference in cycle time. The dynamic expediting rule has a higher number of expediting for all MPCs. However, the dynamic expediting rule produces longer delay time at the 10% MPC and higher percent of jobs tardy at the 10% and 20% MPCs. With respect to queue length, dynamic expediting is the best choice for the 10% and 20% MPCs.

Under higher MPCs such as 30% and 40%, the two buffering styles have no difference in performance in delay time, cycle time, queue length and percentage of tardy jobs. At a mean protective capacity of 30% and 40%, the non-constraint resources may have a lot of spare time to catch up with the schedule, even if a high priority index is given to the potentially late part. These results suggest that under the SPT dispatching rule, the static expediting rule is more favourable than the dynamic expediting.

Table 4.21. Comparison tests for the dynamic and static expediting rules under SPT

The DBR models with MPC = 10%						
Measures	*Average*	s_o	n	t_{n-1}	$t_{0.95;185}$	*Sig.?*
Delay time	213.419	888.314	186	3.277	1.6531	yes
Cycle time	-20.371	591.611	186	-0.469	1.6531	no
Queue length	-8.062	16.757	186	-6.561	1.6531	yes
Percent jobs tardy	1.965	10.449	186	2.566	1.6531	yes
Exped. number	10.796	25.911	186	5.682	1.6531	yes

The DBR models with MPC = 20%						
Measures	*Average*	s_o	n	t_{n-1}	$t_{0.95;52}$	*Sig.?*
Delay time	49.909	259.806	53	1.399	1.6747	no
Cycle time	-1.968	125.415	53	-0.114	1.6747	no
Queue length	-1.767	7.153	53	-1.799	1.6747	yes
Percent jobs tardy	1.884	7.712	53	1.779	1.6747	yes
Exped. number	14.094	30.757	53	3.336	1.6747	yes

Table 4.21. Comparison tests for the dynamic and static expediting rules under SPT (continued)

The DBR models with MPC = 30%						
Measures	*Average*	s_o	n	t_{n-1}	$t_{0.95;160}$	*Sig.?*
Delay time	-5.266	96.038	161	-0.696	1.6544	no
Cycle time	0.563	33.636	161	0.212	1.6544	no
Queue length	0.170	2.854	161	0.757	1.6544	no
Percent jobs tardy	-0.167	3.149	161	-0.673	1.6544	no
Exped. number	12.130	21.484	161	7.164	1.6544	yes

The DBR models with MPC = 40%						
Measures	*Average*	s_o	n	t_{n-1}	$t_{0.95;81}$	*Sig.?*
Delay time	-0.737	48.581	82	-0.137	1.6639	no
Cycle time	0.058	2.688	82	0.196	1.6639	no
Queue length	-0.009	1.530	82	-0.051	1.6639	no
Percent jobs tardy	-0.147	1.107	82	-1.203	1.6639	no
Exped. number	13.146	18.075	82	6.586	1.6639	yes

4.3.3. Comparison of the dynamic and static expediting rules under the FCFS and SPT dispatching rules

This section compares the dynamic and static expediting rules under the FCFS and SPT rules. Since the detailed analyses for the static and dynamic expediting rules for each dispatching rule have been done in the previous section, the analysis focuses on the difference between the SPT and FCFS rules. The analysis starts with the highest mean protective capacity and works through to the lowest mean protective capacity.

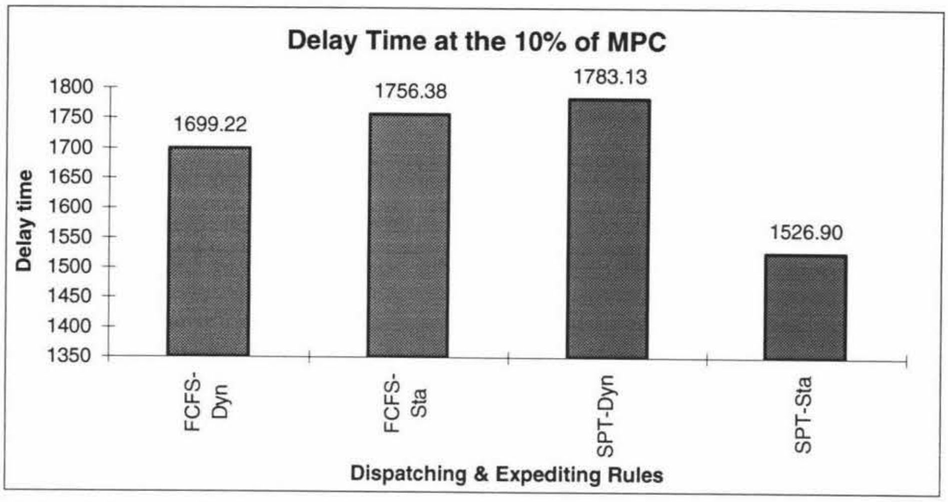


Figure 4.7. The effect of the dispatching and expediting rules on delay time at MPC of 10%

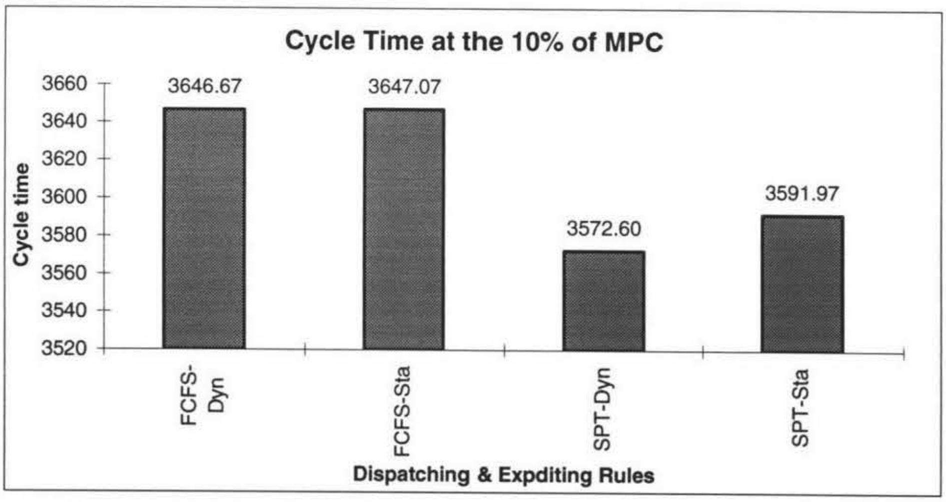


Figure 4.8. The effect of the dispatching and expediting rules on cycle time at MPC of 10%

Figures 4.7-4.10 show shop performance graphs as a function of the dispatching and expediting rules for the 10% level of MPC. The reasonable sample size for this comparison is 173, based on the required samples from the DBR system with FCFS and the static expediting rule. The model with dynamic expediting and FCFS produces a lower delay time as shown in Figure 4.7. This fashion is opposite to the DBR with SPT in which delay time for the static expediting rule is lower than the dynamic expediting rule. It indicates that the shop under FCFS with the static expediting rules and the shop under SPT with the dynamic expediting rule have more late parts in the buffer. The dynamic expediting rule only reduces late parts in the buffer under FCFS. On the other hand, Figure 4.8 indicates that the shop with the SPT dispatching rule produce lower cycle times. The queue length in the shop with FCFS are lower than the shop with SPT (see Figure 4.9).

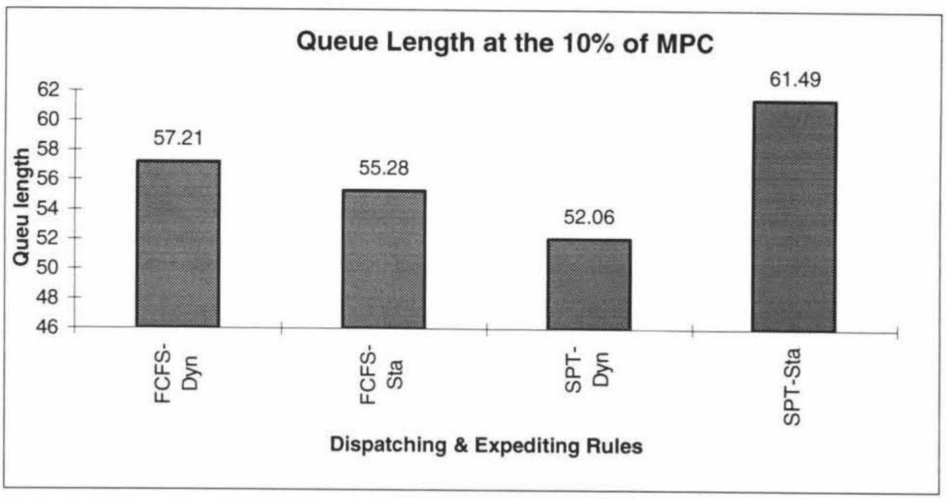


Figure 4.9. The effect of the dispatching and expediting rules on queue length at MPC of 10%

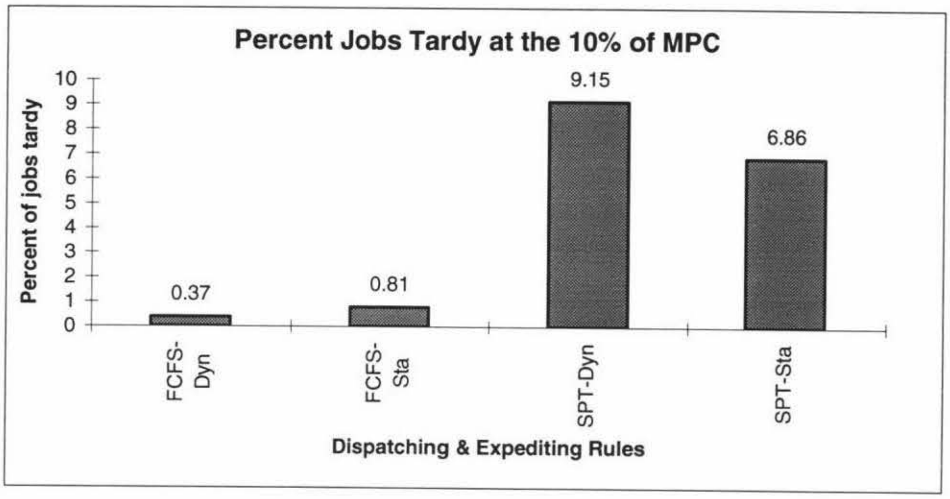


Figure 4.10. The effect of the dispatching and expediting rules on percent jobs tardy at MPC of 10%

Meanwhile, the FCFS rule is more effective in reducing the percentage of tardy jobs under the dynamic expediting rule, as shown in Figure 4.10. The dynamic expediting rule in the SPT shop fails to reduce the percentage of tardy jobs. These results confirm that the dynamic expediting rule works well in the FCFS environment and cannot perform as expected in the SPT environment. This may be explained by two reasons. Firstly, the model with the SPT rule needs a special action to prioritise the expediting part from the queue in front of the non-constraint resources. Secondly, there may be a clash among parts that need expediting at the same resource at the same time. A modified expediting priority could be developed based on this situation. This issue will be raised in the chapter on future research.

The performances in cycle time and percent jobs tardy for the shop at 20% MPC is shown in Figures 4.11 and 4.12. The reasonable samples are 148 observations based on the SPT shop with the dynamic expediting rule. Figure 4.11 also indicates that the SPT shop performs well in reducing cycle time. Again, the FCFS rule provides better delivery performance in the percentage of tardy jobs as shown in Figure 4.12.

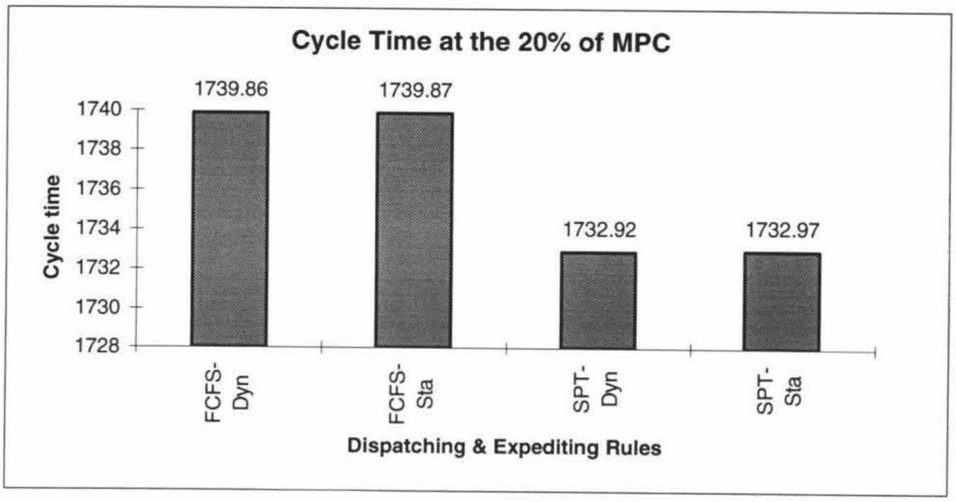


Figure 4.11. The effect of the dispatching and expediting rules on cycle time at MPC of 20%

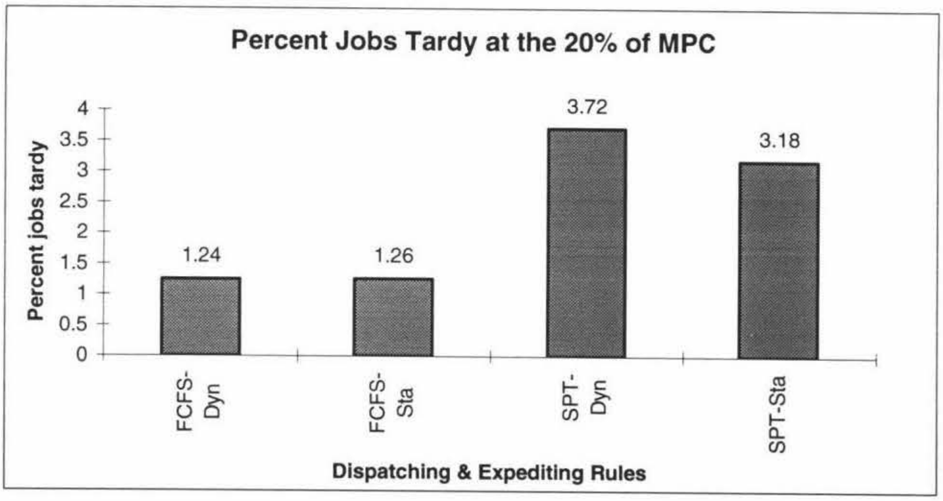


Figure 4.12. The effect of the dispatching and expediting rules on percent jobs tardy at MPC of 20%

The sample size used with an MPC of 30% is 168 based on the SPT rule with static expediting. There is no difference in cycle time under the SPT and FCFS shops as shown in Figure 4.13. Using dynamic expediting under SPT and FCFS has the effect of reducing the percentage of tardy jobs as shown in Figure 4.14.

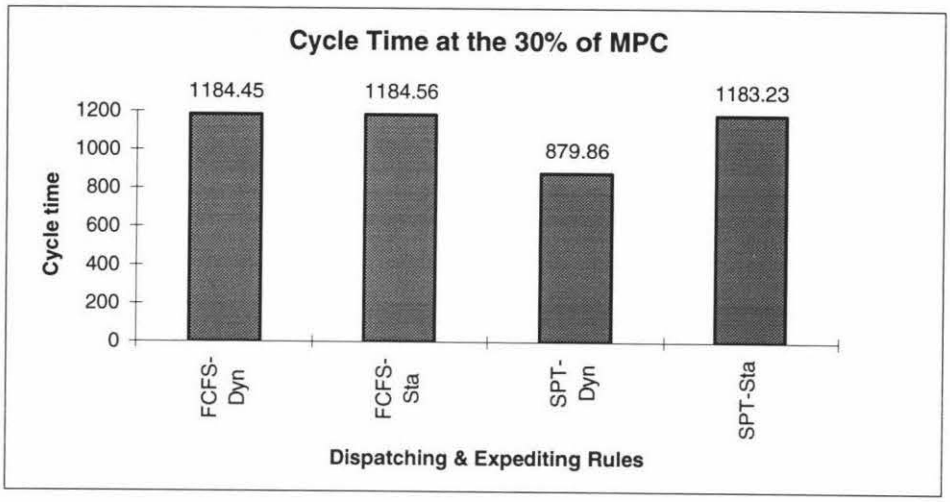


Figure 4.13. The effect of the dispatching and expediting rules on cycle time at MPC of 30%

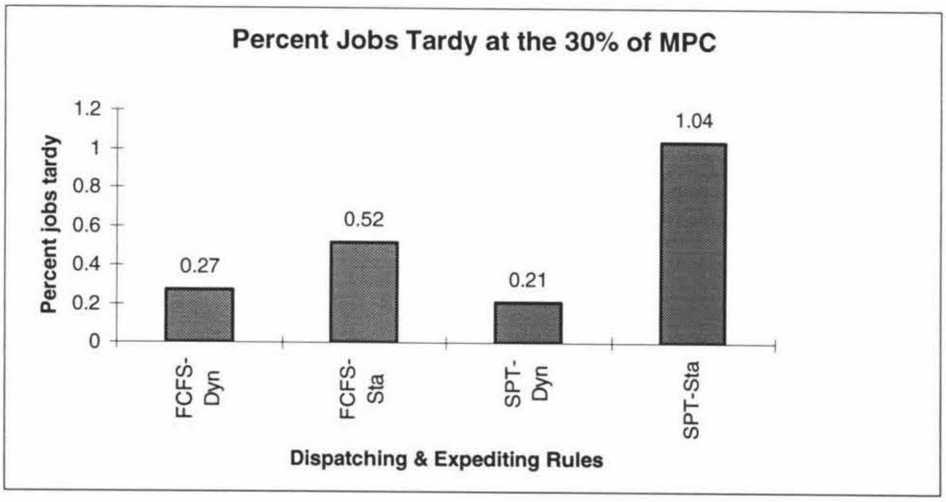


Figure 4.14. The effect of the dispatching and expediting rules on percent jobs tardy at MPC of 30%

The shop at 40% MPC requires 174 observations based on the SPT rule with static expediting. Figures 4.15 and 4.16 show the performance measures for the shops loaded at 40% MPC. The SPT shops have shorter cycle times as shown in Figure 4.15. Again, the dynamic expediting rule reduces the percentage of delayed jobs under the FCFS rule as shown in Figure 4.16.

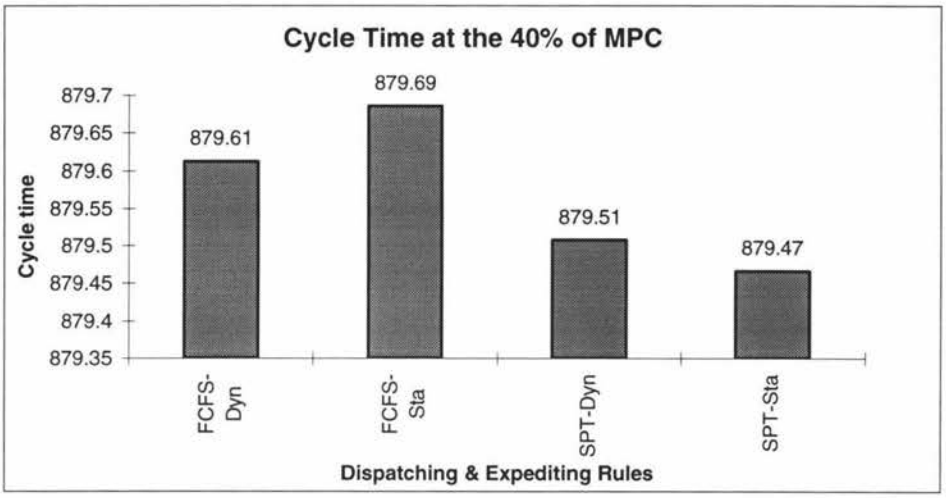


Figure 4.15. The effect of the dispatching and expediting rules on cycle time at MPC of 40%

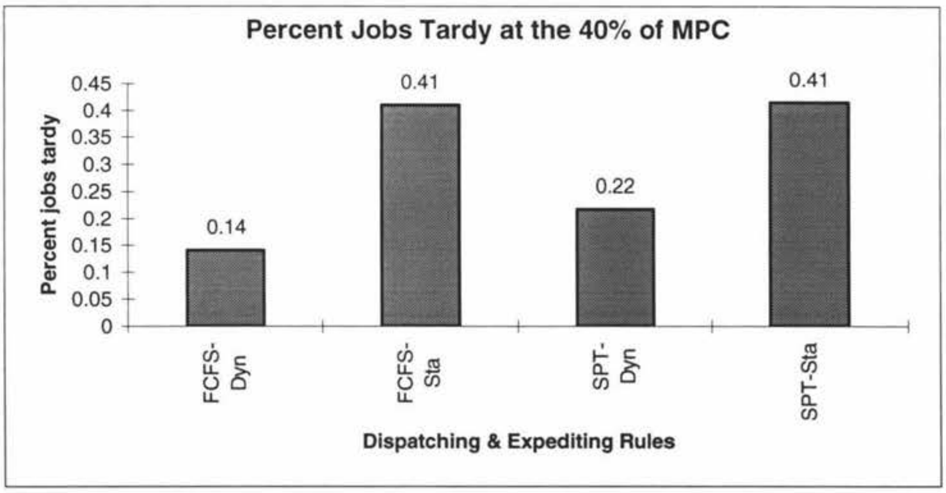


Figure 4.16. The effect of the dispatching and expediting rules on percent jobs tardy at MPC of 40%

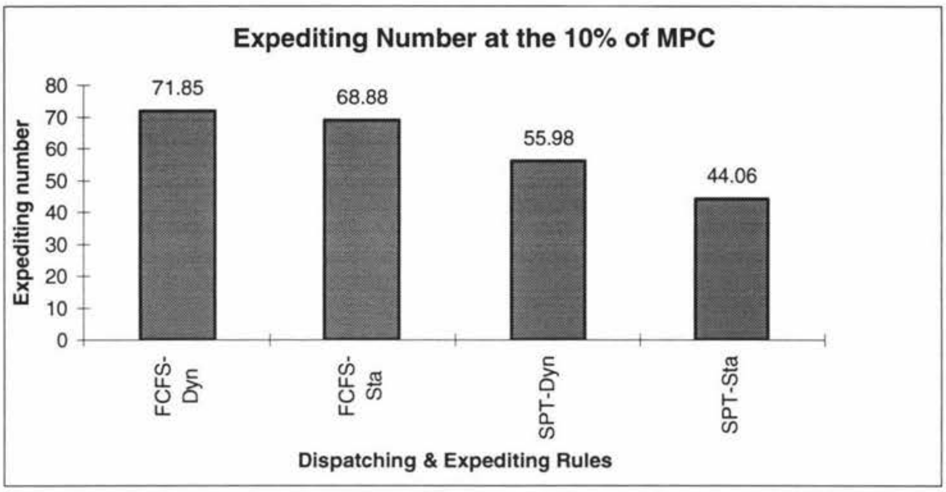


Figure 4.17. The effect of the dispatching and expediting rules on expediting number at MPC of 10%

The final comment is given regarding the number of parts requiring expediting. For example, Figure 4.17 shows that the number of expediting actions for the shop with the dynamic expediting rule is higher than with the static expediting rule at 10% MPC. Results from the above discussion confirm that the dynamic expediting rule needs more expediting for all MPC levels. This indicates that the dynamic expediting is faster at identifying when to expedite. However, the choice of expediting should consider the shop load. The FCFS shops with lower MPC requires the dynamic expediting rule since there is only a small amount of spare capacity to catch up the schedule. When the shops have higher MPC, the static expediting rule is sufficient. The dynamic expediting rule seems to give unsatisfactory performance in the SPT shops.

The other factors in deciding when to employ the expediting rules are the length of buffer multipliers and the processing time variability. The following section explores these two factors. Since practitioners mainly use FCFS highly loaded shops, the investigation into the reduction in processing time variability uses the FCFS rule at 10% of MPC.

4.4. Scenario 3: Analysis of Reduction in Processing Time Variability

This section presents the analysis of variance reduction of processing times. The experiments are based on the FCFS shop with two buffering styles and two coefficients of variability (CV) of 100% and 50%. The experimental parameter for the model that needs to be determine is the buffer multiplier. The buffer multiplier is chosen in such a way that it provides the optimal effect on the percentage of tardy jobs. Figure 4.18 shows the effects of buffer multipliers on percent jobs tardy with a CV of 100%.

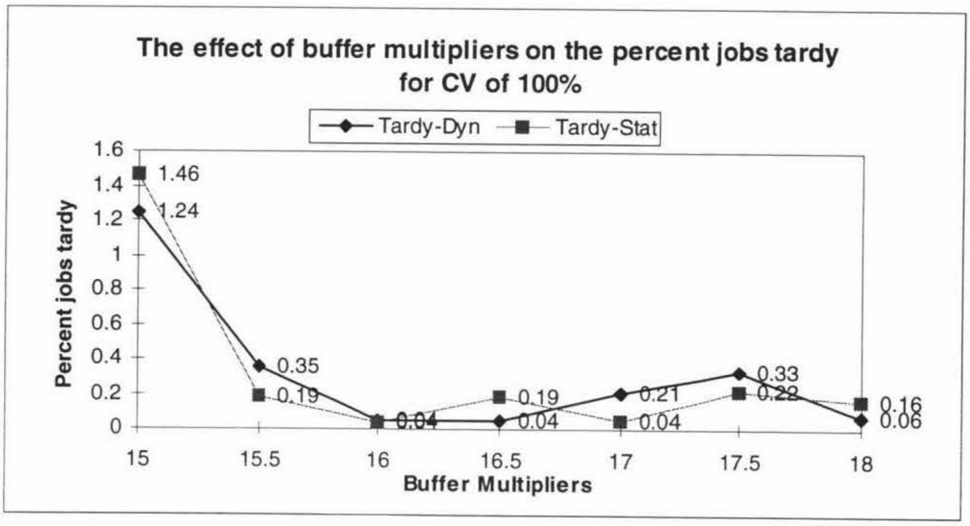


Figure 4.18. Comparison of the effect of buffer multipliers on percent jobs tardy for CV of 100%

The buffer multipliers range from 15 to 18 with a half interval (0.5 units). From the pilot simulation, the reasonable sample size is 99 observations according to the calculation obtained from the FCFS shop with the dynamic expediting rule at a buffer multiplier of 15. The dynamic expediting rule outperforms the static expediting rule in reducing the percentage of tardy jobs with buffer multipliers of 15, 16, 16.5 and 18. The static expediting rule performs better with buffer multipliers of 15.5, 17 and 17.5. In terms of percent jobs tardy, the two expediting rules work to reduce the late orders, as the buffer multipliers are set larger. However, different buffer multipliers have different effects on the delivery performance of the expediting rules. This may occur due to the clash when the delayed parts are expedited on a particular resource. The resource only processes the expedited part based on the FCFS priority rule. If more than one expedited part is waiting at a resource, they will only be dealt with on a FCFS basis. As a result, more orders will be late because of this waiting time. This situation requires a modification in the dynamic expediting when there is a clash among the expedited parts. This issue will be covered in the future research chapter.

Figure 4.19 shows the effects of buffer multipliers on the percentage of tardy jobs when the processing times variability is reduced to 50%. The reasonable sample size is approximately 205 observations due to percentage of tardy jobs for the shop with the static expediting rule and a buffer multiplier of 6. Hence, the observations for all the buffer multipliers are collected as many as this sample size. The selected buffer multiplier ranges between 6 and 9, with the optimum buffer multiplier being 7. This number is less than half the buffer multiplier of the shop with a CV of 100%. The static expediting rule seems preferable to the dynamic expediting rule because it provides reduced percent jobs tardy at lower levels of the buffer multiplier.

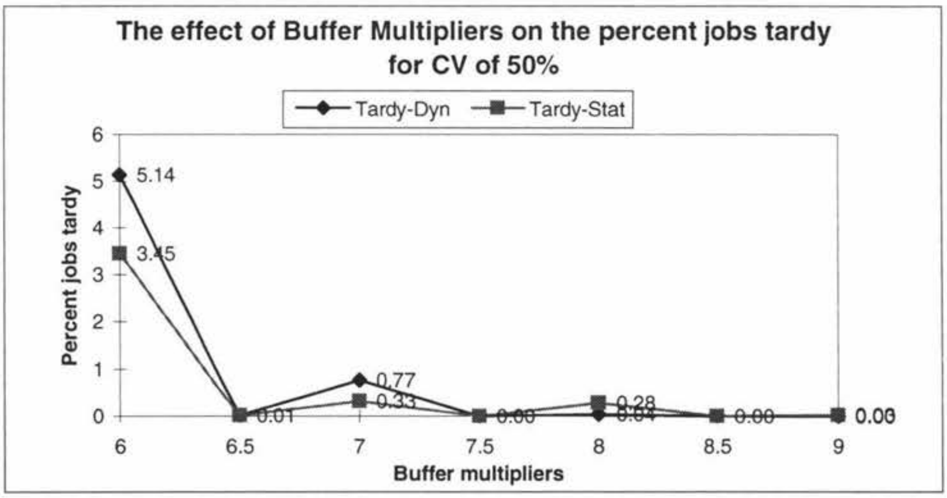


Figure 4.19. Comparison of the effect of buffer multipliers on percent jobs tardy for CV of 50%

The performance of the two expediting rules is the same when the buffer multiplier is greater than 7. This result suggests that the improved quality introduced by reducing the processing times variability causes the shop to only require the simpler static expediting rule. The same random number streams were used to verify this statement by employing comparison tests. The sample size is 73 observations based on the expediting number so that the relative error is less than the planning value of $0.2\bar{d}$. The results are as follows.

(1) Delay time to the buffer

H_o: There is no significant difference between the two expediting rules.

H_a: There is a significant difference between the two expediting rules.

The average $\bar{d}$ is 21.38. The estimated standard deviation s_o is 123.87. The calculated t statistic is 1.47. The tabulated $t_{0.95;72}$ value is 1.66. As a result, the null hypothesis is accepted at the 5% level of significance. It is possible to conclude that there is no significance difference in delay time to the buffer between the two expediting rules.

(2) Cycle time

H_o: There is no significant difference between the two expediting rules.

H_a: There is a significance difference between the two expediting rules.

The average $\bar{d}$ is 0.51. The estimated standard deviation s_o is 8.75. The calculated t statistic is 0.50. The tabulated $t_{0.95;72}$ value is 1.66. As a result, the null hypothesis is accepted at the 5% level of significance. It is possible to conclude that there is no significance difference in cycle time between the two expediting rules.

(3) Queue length in front of the constraint

H_o: There is no significant difference between the two expediting rules.

H_a: There is a significance difference between the two expediting rules.

The average $\bar{d}$ is -0.65. The estimated standard deviation s_o is 4.05. The calculated t statistic is -1.38. The tabulated $t_{0.95;72}$ value is 1.66. As a result, the null hypothesis is accepted at the 5% level of significance. It is possible to conclude that there is no significance difference in queue length in front of the constraint between the two expediting rules.

(4) Percent jobs tardy

H_o: There is no significant difference between the two expediting rules.

H_a: There is a significance difference between the two expediting rules.

The average $\overline{d}$ is 0.74. The estimated standard deviation s_o is 2.76. The calculated t statistic is 2.76. The tabulated $t_{0.95;72}$ value is 1.66. As a result, the null hypothesis is rejected at the 5% level of significance. It is possible to conclude that the shop with the static expediting has a lower percent jobs tardy by approximately 0.74 percent per simulated day compare to the shop with the dynamic expediting rule.

(5) Number of expediting

H_o: There is no significant difference between the two expediting rules.

H_a: There is a significance difference between the two expediting rules.

The average $\overline{d}$ is 40.39. The estimated standard deviation s_o is 39.79. The calculated t statistic is 8.67. The tabulated $t_{0.95;72}$ value is 1.66. As a result, the null hypothesis is rejected at the 5% level of significance. It is possible to conclude that the shop with the static expediting rule has a lower number of expediting by 40.39 parts per simulated day compared with the shop with the dynamic expediting rule.

The results listed above confirm that the static expediting rule is favourable compared with the dynamic expediting rule. The results show that the static expediting results in lower late parts and less expediting. There are no differences in the other performance measures.

4.5. Summary

The analysis for the three experimental scenarios have been outlined. The results of scenario one can be summarised as follows. The FCFS dispatching rule outperforms other dispatching rules in terms of percent jobs tardy. The SPT dispatching rule can confidently be employed in the shop to reduce cycle time.

The shop using the expediting rule provides reduced percent jobs tardy than without an expediting mechanism. Nevertheless, the results of scenario two advocate that the dynamic expediting rule is valuable to use in a shop with low mean protective capacity of 10% and with the FCFS dispatching rule. The shop under the SPT dispatching rule performs better using the static expediting rule to reduce the late orders as well as cycle times. Scenario three enables the shop to reduce the buffer multiplier from 16 to 7 at 10% of MPC. The static expediting rule is preferable to use given lower variability of processing times. Some ideas for future work have been identified in this chapter and will be elaborated on in Chapter 5.

Chapter 5
Future Research

5.1. Introduction

The present study has certain limitations related to the choice of experimental conditions of the model. A simulation study of a make to order company was used in the study because of its ability to reveal the features of Buffer Management when using various dispatching and expediting rules. Besides a make-to-stock strategy, there are several product position strategies, such as make-to-stock and make-to-assembly, that can be used to formulate further study into the effectiveness of Buffer Management (Fogarty *et al.*, 1991). A basis for selecting the adequate strategy in terms of advantages and disadvantages of each strategy will provide future work. The next section describes some areas that this study has identified as having high potential for future research in Buffer Management.

5.2. Avenues for Future Study

The simulation models developed in this study have suggested some valuable areas for future research. In total there are eight future studies that have been identified during the analysis: determination of buffer length, plant types, shop performance, conflict in expediting priority, effects of disruptions, process and transfer batches, interactive constraints and buffer information system.

5.2.1. Determination of buffer length

This study would vary the level of mean protective capacity and selecting the optimal buffer multiplier to provide a better experimental parameter. The indicator variable used is the percentage of tardy jobs. The resource constraint works on the orders that are available in the buffer. Job tardiness only refers to the swapping time between two jobs at the resource constraint. A general method is required in the DBR shop where the constraints include the assembly and shipping areas. There are three alternatives to improve the method to determine the buffer length. Firstly, a modified performance measure is required to determine the optimum value for the buffer multipliers. Emshoff and Sisson (1970) have proposed a method termed the specific search procedure. The buffer multiplier serves as a controllable factor. Response surface methodology then can be used to sequentially vary the buffer multiplier until it is no longer possible to vary any other factors individually and improve the response in the indicator variable.

A second suggestion is to use real time scheduling. The idea behind this is to quantify the ability of the shop to fully protect the constraint's schedule with the buffer status measured in terms of the number of missing parts. The user defined program with "if-then" function can be developed to authorise the shop either to increase or to decrease the buffer multipliers, based on the percentage of missing parts. If the missing part is found in the red zone, the buffer multiplier needs to relax. The optimal choice of the red zone is a good setting to start this approach.

The third method is by the use of dynamic buffering after Goldratt (1990). The two expediting rules were tested within the static buffering environment in which there was no evaluation of the buffer length resulting from the load of the non-constraint resources. Buffer length is assumed to be static and protects the constraint from disruptions and non-instant availability (Goldratt, 1990). On the other hand, dynamic buffering considers the load of the non-constraint resources. It attempts to eliminate the effect of non-instant availability. By checking the load of the non-constraint resource capacity in a given bucket time, a job may be released earlier, hence the buffer length is altered. The dynamic buffering environment may be considered to test the expediting rules in future research.

5.2.2. Plant types

The plant type used in this study is a modification of the A type plant where the constraint is found at the end of the process of the shop (Umble and Srikanth, 1990). It is not known whether the dynamic expediting rule can perform well in the other three plant types, T-plants, V-plants and combination plants. An area of future research is to investigate different plant types to determine the effect of the dynamic expediting rule on shop performance.

5.2.3. Shop performance

Buffer Management is aimed at controlling the shop to achieve the global goal, that is to deliver the product to the customer and thereby make money. Shop performance indicates the impact of operational actions to pursue the global goal. There is a need for determining the shop performance before assessing the effectiveness of dispatching rules. Goldratt (1990) suggests two local performance measures in Buffer Management: throughput-dollar-days (TDD) and inventory-dollar-days (IDD). The equivalent measure suggested by Schragenheim and Ronen (1995) are late-order-days (LOD). LOD counts the sum of days behind the schedule for a late job. Therefore, investigating dispatching rules with due-date criteria are more preferable in a DBR environment, including mean tardiness, percent tardy, maximum tardiness, root mean square of tardiness and mean absolute lateness (Fry *et al.*, 1988). Future research is required to examine the effects on these dispatching rules on the local performance measures in TDD, IDD and LOD if dispatching rules are employed.

5.2.4. Conflict in expediting priority

This study finds that the dynamic expediting rule is faster in deciding when to expedite. However, more expedited jobs identified lead to more load being put on the non-constraint resources. As a result, the expediting parts have long waiting times to process at non-constraints. The conflict arises due to the determination of priority to authorise the non-constraint resource to work on the expedited parts, so that the expediting time provided is still sufficient. If the number of conflicts is large, the expediting parts are very likely to arrive at the constraint late. Schragenheim and Ronen (1995) have outlined two schemes to cope with this situation. Firstly, the non-constraint resource is authorised to process the expedited parts following the FCFS priority rule. The resource immediately stops its current job and sets up for the expediting part.

Secondly, the non-constraint resource is authorised to process the expediting parts only when at least one piece is available for the expediting job. The idea of the second scheme is to avoid wasted capacity, whereas the first scheme wants to push the expediting parts as fast as possible. The dynamic expediting rule is thought to cooperate with these two schemes. Future research can explore the effectiveness of the dynamic expediting in the conflicting environment.

5.2.5. Effects of disruptions

This study excludes the effects of disruptions on shop performance. However, disruptions usually occur in the real shop, as part of statistical fluctuations. Future research may involve some sources of disruptions such as machine down times, set-up times, alternative routings, rush orders, transfer and process batches, and overtime.

5.2.6. Process and transfer batches

This study assumes that batch size is a single entity. Once demand is assigned to a particular product type, the resource processes them as one batch and also transfers them in one batch to the next resource. In this sense, it does not differentiate between the transfer and process batches. Umble and Srikanth (1990, p. 115) define process batch as "the quantity of product processed at a resource before that resource changes over to produce a different product". The amounts of products that are moved from one resource to the next one is referred to as the transfer batch. Goldratt and Cox (1992) contend that the process batch should not equal the transfer batch to maintain the smooth flow in a manufacturing line. Choices of the process and transfer batches are interesting topics for future research in Buffer Management.

5.2.7. Interactive constraints

As a consequence of the complex product mix, there are often two or more constraints in a well-established job shop. Interactive constraints occur when one constraint directly or indirectly feeds the other constraints to produce at least one product (Goldratt, 1990). In the shop with one constraint, interactive constraints only occur if this constraint feeds itself. Interactive constraints will make the due date performance deteriorate faster. In this situation, the fast response expediting rule can be used to advise when to expedite and hence better control the flow of parts without endangering due date performance. This study reveals that the dynamic expediting rule has a fast response to the need to identify parts requiring expediting. Future work can be developed to tackle interactive constraints with the dynamic expediting rule.

5.2.8. Buffer information system

As well as a warning system to begin expediting, buffer status may be used to capture information to initiate ongoing improvement. Examples are to find the sources of disruptions and to reveal the existence of interactive constraints. By monitoring the buffer status, the effectiveness of the non-constraint resources can be found, for instance the level of protective capacity at a particular time. Umble and Srikanth (1990) suggest that disruption factors can be used to prioritise the maintenance on the particular resource. In addition, visualising the buffer status has the advantage of showing the trade-off between the buffer time needed to absorb disruptions. Future research into the monitoring procedure and displaying the buffer status in a useful fashion could include using expediting rules to provide a timely basis of information for ongoing improvement. Jaikumar (1996) describes the importance of a visual buffer information system to tackle common disruptions.

5.3. Summary

This chapter has given some areas for future research in Buffer Management as a control means in production management. There are eight suggestions for future research which will enhance the knowledge of Buffer Management in practice. The next chapter presents some conclusions derived from this research.

Chapter 6
Conclusions

6.1. Introduction

This simulation study has attempted to resolve three problems associated with Buffer Management: the effects on shop performance of dispatching rules, expediting rules, and variance reduction. This chapter outlines some of the conclusions that have been drawn from this study.

6.2. Dispatching Rules

Dispatching rules provide a means by which a particular resource prioritises parts in a queue. The FCFS, SPT and MINSLK dispatching rules were examined with various levels of mean protective capacity (MPC). From the simulation results, the shop with 10% mean protective capacity has the largest buffer multiplier of 16. For other shops considered with mean protective capacities of 20%, 30% and 40%, the buffer multiplier was reduced to 8, 6 and 5 respectively. These findings assert that a heavily loaded shop needs a longer buffer time to attain the same percentage of tardy jobs. The heavily loaded shop experienced high congestion and fully utilised resources. On the other hand, shops with higher MPC can maintain the same level of tardy jobs using a shorter buffer multiplier.

The simulation results indicate that the FCFS dispatching rule achieves the best due date performance, measured by the percent jobs tardy. The SPT dispatching rule consistently outperforms the FCFS and MINSLK dispatching rules in terms of cycle time and delay time at the 10%, 20% and 30% mean protective capacity. The shop with the highest mean protective capacity (40%) provides a similar cycle time regardless of which dispatching rule is used.

Given increasing time-based competition, a facility needs to reduce cycle times as well as the percentage of tardy jobs. Even though the shop using the SPT dispatching rule results in reduced average cycle time, it cannot satisfy its customers in terms of delivery reliability. This research recommends that practitioners use the FCFS dispatching rule which may result in longer cycle times but does lead to less tardy jobs.

In terms of queue length, FCFS is the dispatching rule of choice. A facility with a lower queue length helps practitioners to easily track and solve the quality problems.

6.3. Expediting Rules

Expediting rules are mechanisms for deciding when to expedite parts that are potentially late arrivals at the buffer. This study tested both static and dynamic expediting rules, to determine their effectiveness given variability in processing times, under different levels of MPC and various dispatching rules.

The shop using FCFS performs best in conjunction with the dynamic expediting rule, assisting Buffer Management to reduce job tardiness especially when the MPC is low. At the higher levels of MPC, the static and dynamic expediting rules produce the same performance in delay time, cycle time and percent jobs tardy, but less expediting is required with the static expediting rule. These results suggest practitioners should use the dynamic expediting rule when mean protective capacity is low and to consider the static expediting rule at higher levels of MPC.

The dynamic expediting rule does not perform as expected in reducing the percentage of tardy jobs when used in conjunction with the SPT dispatching rule. The shop using the SPT dispatching rule should use the static expediting rule which results in less expediting and a lower cycle time. However, this research finds that the FCFS dispatching rules more consistently produces a lower percentage of tardy jobs when combined with either the static or dynamic expediting rules as compared to the SPT dispatching rule.

6.4. Reduction in Processing Time Variability

The shop with 10% mean protective capacity under FCFS demonstrates improved performance in all criteria when the coefficient of variability is reduced from 100% to 50%. The shop with high variability has a range of buffer multipliers between 15 and 18. After reducing the processing time variability, the range of the buffer multiplier reduces to between 6 and 9. As a result, the optimal buffer multiplier reduces by more than 50%, from 16 to 7.

The static expediting rule is preferable for the lower processing time variability because it has less expediting and lower late parts as contrasted to the dynamic expediting rule. This research suggests that there are a lot of opportunities to improve the facility using the FCFS dispatching rule that can provide a profound impact on shop performance. A consistent improvement in the manufacturing line will keep and win customer loyalty.

6.5. Summary

This research has investigated the effects of dispatching and expediting rules on facility performance operating under the DBR methodology. The analysis also considers the reduction of processing time variability, which reflects the effort of quality improvement on the shop floor.

This conclusion chapter has presented the main points obtained from analysis and discussion. The simulation results confirm that the FCFS dispatching rule is able to maintain reliable delivery performance, that is, fulfilling customer expectations. The dynamic expediting rule works better in the shop using the FCFS dispatching rule and does not perform as expected in the shop using the SPT dispatching rule. While the improvement process results in reduced cycle time, practitioners need to consider the use of the static expediting rule to avoid the extra cost of expediting.

There is a lot of work required to improve Buffer Management as a control mechanism for the shop floor. This research has outlined eight ideas for future research that can be used to initiate this process of improvement. The aim of future research on Buffer Management should be to create and disseminate knowledge on effective methods for shop floor control. Practitioners can then reap the benefits of research to better understand their own business processes.

References

Ashton, J.E., Cook, F.X., "Time to Reform Job Shop Manufacturing", *Harvard Business Review*, Vol. , No. 2, 1989, pp. 106-111.

Ashcroft, S.H., "Applying the Principles of Optimized Production Technology in a Small Manufacturing Company", *Engineering Costs and Production Economics*, Vol. 17, No. 1, 1989, pp. 79-88.

Atwater, J.B., Chakravorty, S.S., "Does Protective Capacity Assist Managers in Competing Along Time-Based Dimensions?", *Production and Inventory Management Journal*, Vol. 35, No. 3, 1994, pp. 53-59.

Baker, K.R., *Introduction to Sequencing and Scheduling*, Wiley, New York, 1974.

Baker, K.R., "Sequencing Rules and Due-Date Assignments in A Job Shop", *Management Science*, Vol. 30, No. 9, 1984, pp. 1093-1104.

Banks, J., Carson, J.S., *Discrete Event System Simulation*, Prentice-Hall, Englewood Cliffs, NJ, 1984.

Blackstone, J.H., Phillips, D.T., Hogg, G.L., "A State-of-the-art Survey of Dispatching Rules for Manufacturing Job Shop Operations", *International Journal of Production Research*, Vol. 20, No. 1, 1982, pp. 27-45.

Carter, P.L., Melnyk, S.A., Handfield, R.B., "Identifying the Basic Process Strategies for Time-Based Competition", *Production and Inventory Management Journal*, Vol. 36, No. 1, 1995, pp. 65-70.

Chakravorty, S.S., Atwater, J.B., "A Comparative Study of Line Design Approaches for Serial Production Systems", *International Journal of Operations and Production Management*, Vol. 16, No. 6, 1996, pp. 91-108.

Chakravorty, S.S., Sessum, J.L., "Developing Effective Strategies to Prioritize Set-up Reduction in a Multi-Machine Production System: A Throughput Approach", *International Journal of Operations and Production Management*, Vol. 15, No. 10, 1995, pp. 103-111.

Colvenaer, D.D., Maes, J., Gelders, L. "Application of TOC/OPT Rules in a Medium Sized Shop", *Production Planning and Control*, Vol. 3, No. 4, 1992, pp. 413-421.

Cook, D.P., "A Simulation Comparison of Traditional, JIT, and TOC Manufacturing Systems in A Flow Shop with Bottlenecks", *Production and Inventory Management Journal*, Vol. 35, No. 1, pp. 73-78, 1994.

Conover, W.J., *Practical Nonparametric Statistics*, 2nd ed., Wiley, New York, 1980.

Conway, R.W., Maxwell, W.L., McClain, J.O., Thomas, L.J., "The Role of Work-In-Process Inventory in Serial Production Lines", *Operations Research*, Vol. 36, No. 2, 1988, pp. 229-241.

Conway, R.W., Maxwell, W.L., Miller, L.W., *Theory of Scheduling*, Addison-Wesley, Reading, MA, 1967.

Crandall, R.E., Burwell, T.H., "The Effect of Work-In-Process Inventory Levels on Throughput and Lead Times", *Production and Inventory Management*, Vol. 34, No. 1, 1993, pp. 6-11.

Emshoff, J.R., Sisson, R.L., *Design and Use of Computer Simulation Models*, MacMillan, London, 1970.

Fishman, G.S., *Principles of Discrete Event Simulation*, Wiley, New York, 1978, pp.237-247.

Fogarty, D.W., Blackstone, J.H., Hoffmann, T.R., *Production & Inventory Management*, 2nd ed., South-Western Publishing, Cincinnati, OH, 1991, pp. 648-687.

Fry, T.D., Karwan, K.R., Steele, D.C., "Implementing Drum-Buffer-Rope to Control Manufacturing Lead Time", *International Journal of Logistics Management*, Vol. 2, No. 1, 1991, pp. 12-18.

Fry, T.D., Philipoom, P.R., Leong, G.K., Smith, A.E., "An Investigation of Bottleneck Position in a Multi-Stage Job Shop", *International Journal of Operations and Production Management*, Vol. 7, No. 6, 1987, pp. 55-63.

Fry, T.D., Philipoom, P.R., Markland, R.E., "Dispatching in a Multistage Job Shop where Machine Capacities are Unbalanced", *International Journal of Production Research*, Vol. 26, No. 7, 1988, pp. 1193-1223.

Gardiner, S.C., Blackstone, J.H., Gardiner, L.R., "Drum-Buffer-Rope and Buffer Management", *International Journal of Operations and Production Management*, Vol. 13, No. 6, 1993, pp. 68-78.

Goldratt, E.M., "Computerized Shop Floor Scheduling", *International Journal of Production Research*, Vol. 26, No. 3, 1988, pp. 443-455.

Goldratt, E.M., *The Haystack Syndrome: Shifting Information Out of the Data Ocean*, North River Press, Croton-on-Hudson, NY, 1990.

Goldratt, E.M., Cox, J., *The Goal*, North River Press, Croton-on-Hudson, NY, 1984.

Goldratt, E.M., Cox, J., *The Goal: A Process of Ongoing Improvement*, rev. 2nd ed., North River Press, Croton-on-Hudson, NY, 1992.

Goldratt, E.M., Fox, R.E., *The Race: For A Competitive Edge*, North River Press, Croton-on-Hudson, NY, 1986.

Graves, R.J., Konopka, J.M., Milne, R.J., "Literature Review of Material Flow Control Mechanisms", *Production Planning and Control*, Vol. 6, No. 5, 1995, pp. 395-403.

Guide, V.D.R., "A Simulation Model of Drum-Buffer-Rope for Production Planning and Control at a Naval Aviation Depot", *Simulation*, Vol. 65, No. 3, 1995, pp. 147-168.

Guide, V.D.R., Ghiselli, "Implementation of Drum-Buffer-Rope at a Military Rework Depot Engine Works", *Production and Inventory Management Journal*, Vol. 36, No. 3, 1995, pp. 79-82.

Hicks, C.R., *Fundamental Concepts in the Design of Experiments*, 3rd ed., Holt, Rinehart, and Winston, New York, 1982.

Hurley, S.F., "A Practical Heuristic for Effective Buffer Management", *International Journal of Operations and Production Management*, Vol. 16, No. 10, 1996, pp. 89-101.

Huq, Z., Bernardo, J.J., "The Sensitivity of Job Mix and Load Capacity Bottlenecks on Inventory and Due Date Performance in A Manufacturing System", *Production Planning and Control*, Vol. 6, No. 6, 1995, pp. 516-529.

Jacobs, R.R., Bragg, D.J., "Repetitive Lots: Flow Time Reductions Through Sequencing and Dynamic Batch Sizing", *Decision Sciences*, Vol. 19, No. 1, 1988, pp. 281-294.

Jaikumar, R., "When More is Less: A Minimalist Architecture for Manufacturing", *European Management Journal*, Vol. 14, No. 5, 1996, pp. 433-441.

Kleijnen, J.P.C., *Statistical Tools for Simulation Practitioners*, Marcel Dekker, New York, 1987.

Kleijnen, J.P.C., "Verification and Validation of Simulation Models", *European Journal of Operational Research*, Vol. 82, No. 1, 1995, pp. 145-162.

Kramer, C.Y., "Extensions of Multiple Range Tests to Group Means with Unequal Numbers of Replications", *Biometrics*, Vol. 12, 1956, pp. 307-310.

Law, A.M., Kelton, W.D., *Simulation Modeling and Analysis*, 2nd ed., McGraw-Hill, New York, 1991.

Levene, H., "Robust Tests for Equality of Variances", *Contributions to Probability and Statistics: Essays in Honor of H. Hotelling*, ed. I. Olkin, Stanford University Press, 1960.

Lockamy, A., Cox, J.F., *Reengineering Performance Measurement: How to Align Systems to Improve Processes, Products, and Profits*, Irwin, Burr Ridge, Illinois, 1994.

Melnyk, S.A., Denzler, D.R., Fredendall, L., "Variance Control vs. Dispatching Efficiency", *Production and Inventory Management Journal*, Vol. 33, No. 3, 1992, pp. 6-13.

Melnyk, S.A., Ragatz, G.L., "Order Review/Release: Research Issues and Perspectives", *International Journal of Production Research*, Vol. 27, No. 7, 1989, pp. 1081-1096.

Miller, D.M., Fink, R.L., "A Study of Expediting Practices in Manufacturing Firms", *Production Planning and Control*, Vol. 1, No. 3, 1990, pp. 170-178.

Morgan, G., "Paradigms, Methapors, and Puzzle Solving in Organizational Theory", *Administrative Science Quarterly*, Vol. 25, No. 4, 1980, pp. 605-621.

Noreen, E., Smith, D., Mackey, J.T., *The Theory of Constraints and Its Implications for Management Accounting*, North River Press, Great Barrington, MA, 1995.

Philipoom, P.R., Malhotra, M.K., Jensen, J.B., "An Evaluation of Capacity Sensitive Order Review and Release Procedures in Job Shops", *Decision Sciences*, Vol. 24, No. 6, 1993, pp. 1109-1133.

Radovilsky, Z.D., "Estimating the Size of the Time Buffer in the Theory of Constraints: Implications for Management", *International Journal of Management*, Vol. 11, No. 3, 1994, pp. 839-847.

Ragatz, G.L., Mabert, V.A., "An Evaluation of Order Release Mechanisms in A Job-Shop Environment", *Decision Sciences*, Vol. 19, No. 1, 1988, pp. 167-189.

Ramsay, M.L., Brown, S., Tabidzadeh, K., "Push, Pull, and Squeeze Shop Floor Control with Computer Simulation", *Industrial Engineering*, Vol. 22, No. 2, 1990, pp. 39-45.

Reimer, G., "Material Requirements Planning and Theory of Constraints: Can They Coexist? A Case Study", *Production and Inventory Management Journal*, Vol. 32, No. 4, 1991, pp. 48-52.

Russell, E.C., *Building Simulation Models with SIMSCRIPT II.5*, CACI, Los Angeles, CA, 1989.

Schragenheim, E., Cox, J.F., Ronen, B., "Process Flow Industry - Scheduling and Control Using Theory of Constraints", *International Journal of Production Research*, Vol. 32, No. 8, 1994, pp. 1867-1877.

Schragenheim, E., Ronen, B., "The Drum-buffer-rope Shop Floor Control", *Production and Inventory Management Journal*, Vol. 31, No. 3, 1990, pp. 18-23.

Schragenheim, E., Ronen, B., "Buffer Management: A Diagnostic Tool For Production Control", *Production and Inventory Management Journal*, Vol. 32, No. 2, 1991, pp. 74-79.

Schragenheim, E., Ronen, B., "Expediting as a Two-Edge Sword", *Journal of Systems Improvement*, Vol. 1, No. 1, 1995, pp. 33-38.

Spencer, M.S., "Using 'The Goal' in an MRP System", *Production and Inventory Management Journal*, Vol. 32, No. 4, 1991, pp. 22-28.

Spencer, M.S., Cox, J.F., "Master Production Scheduling Development in A Theory of Constraints Environment", *Production and Inventory Management Journal*, Vol. 36, No. 1, 1995a, pp. 8-14.

Spencer, M.S., Cox, J.F., "Optimum Production Technology (OPT) and the Theory of Constraints (TOC): Analysis and Genealogy", *International Journal of Production Research*, Vol. 33, No. 6, 1995b, pp. 1495-1504.

Stalk, G., "Time - The Next Source of Competitive Advantage", *Harvard Business Review*, July-August, 1988, pp. 41-51.

Steel, R.G.D., Torrie, J.H., *Principles and Procedures of Statistics: A Biometrical Approach*, 2nd ed., McGraw-Hill, Auckland, 1980.

Tersine, R.J., Hummingbird, E.A., "Lead-Time Reduction: The Search for Competitive Advantage", *International Journal of Operations and Production Management*, Vol. 15, No. 2, 1995, pp. 8-18.

Umble, M.M., Srikanth, M.L., *Synchronous Manufacturing: Principles for World Class Excellence*, South-Western Publishing, Cincinnati, OH, 1990.

Wisner, J.D., "A Review of the Order Release Policy Research", *International Journal of Operations and Production Management*, Vol. 15, No. 6, 1995, pp. 25-40.

Wu, S.Y., Morris, J.S., Gordon, T.M., "A Simulation Analysis of the Effectiveness of Drum-Buffer-Rope Scheduling in Furniture Manufacturing", *Computers and Industrial Engineering*, Vol. 26, No. 4, 1994, pp. 757-764.

Appendix I
Product Routings and Processing Times Data

Product Type Code	Task Resource Identity	Task Processing Time (min.)			
		10% MPC	20% MPC	30% MPC	40% MPC
1	5	3.0	3.0	2.4	1.7
	6	2.3	1.8	1.5	1.6
	2	2.0	1.7	1.4	1.2
	8	1.7	1.7	1.2	1.3
	1	2.7	2.3	2.0	1.5
	9	1.8	1.8	1.8	1.8
2	3	2.3	1.7	1.6	1.5
	4	2.0	1.8	1.5	1.3
	2	1.9	1.4	1.3	1.2
	7	2.2	1.8	1.7	1.6
	8	1.7	1.7	1.2	1.3
	6	2.2	1.7	1.6	1.5
	9	1.6	1.6	1.6	1.6
3	2	1.8	1.6	1.4	1.1
	5	3.1	2.3	2.1	1.6
	6	2.3	1.8	1.7	1.4
	1	2.5	2.4	2.1	1.6
	3	2.1	2.0	1.9	1.3
	7	2.3	1.9	1.8	1.5
	4	1.8	1.6	1.3	1.4
	8	1.9	2.0	1.5	1.3
	9	2.2	2.2	2.2	2.2
4	6	2.3	2.1	2.0	1.6
	3	2.4	1.9	1.8	1.4
	2	1.6	1.6	1.5	1.2
	4	2.1	2.1	1.7	1.6
	5	2.8	2.1	1.8	1.8
	7	2.1	2.0	1.8	1.7
	8	2.0	1.5	1.2	1.3
	9	1.6	1.6	1.6	1.6
5	1	2.7	2.2	2.1	1.7
	2	1.9	1.7	1.3	1.0
	7	2.2	1.8	1.7	1.6
	8	2.1	1.5	1.4	1.5
	5	3.0	2.0	1.9	1.7
	3	2.0	2.1	2.0	1.5
	4	1.9	1.8	1.5	1.6
	9	1.6	1.6	1.6	1.6

Appendix I

Product Routings and Processing Times Data

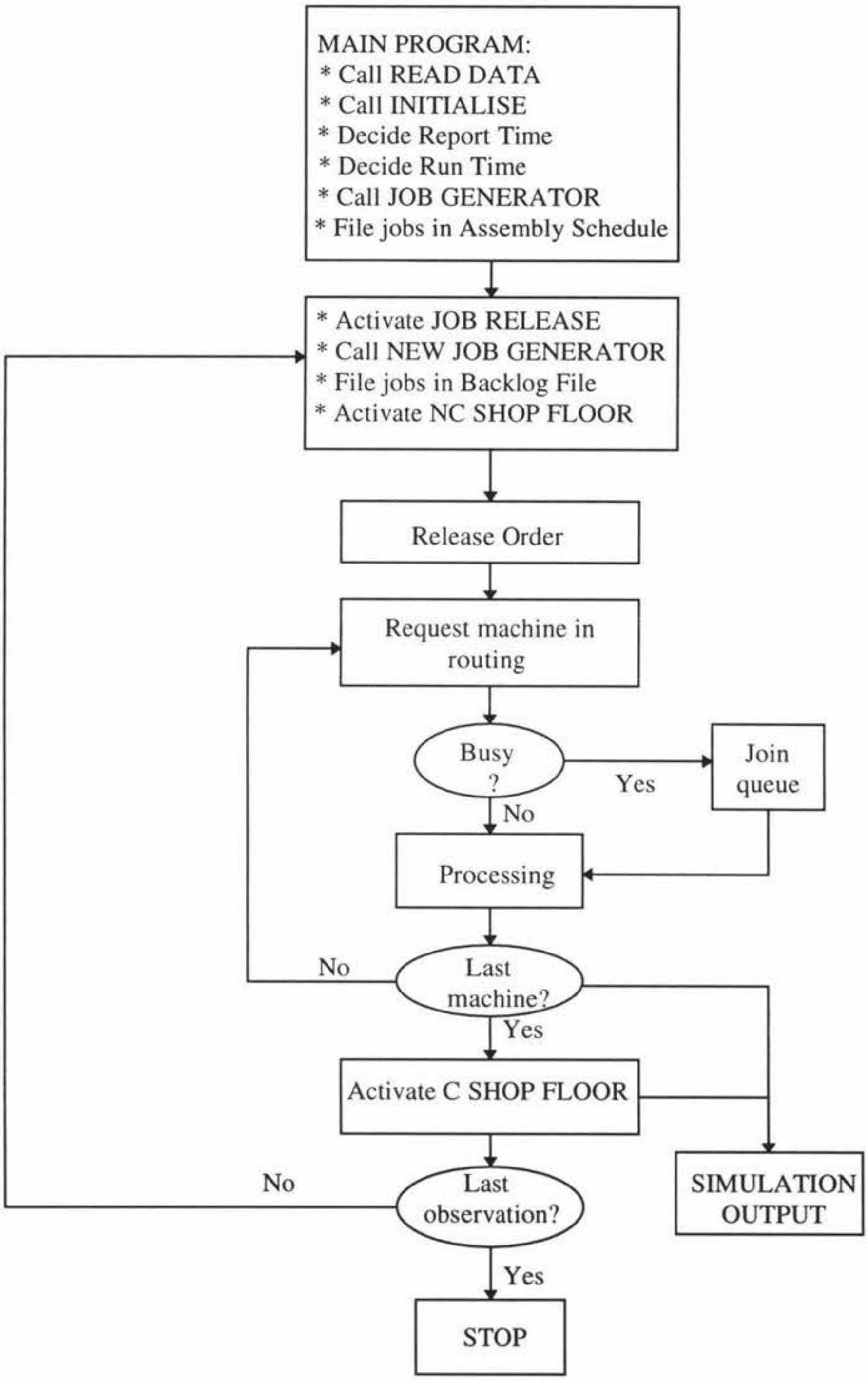
(Continued)

Product Type Code	Task Resource Identity	Task Processing Time (min.)			
		10% MPC	20% MPC	30% MPC	40% MPC
6	3	2.5	2.0	1.9	1.5
	4	2.0	1.9	1.2	1.5
	1	1.6	1.6	1.4	1.1
	6	2.2	2.3	1.9	1.5
	8	2.2	1.8	1.5	1.4
	2	1.6	1.6	1.4	1.1
	9	2.0	2.0	2.0	2.0
7	8	2.3	2.0	1.6	1.5
	7	2.4	2.0	1.9	1.7
	4	1.9	1.8	1.7	1.3
	3	2.3	1.7	1.6	1.3
	2	2.0	1.7	1.3	1.2
	9	2.2	2.2	2.2	2.0
8	7	2.2	2.1	2.0	1.9
	2	1.9	1.8	1.5	1.1
	6	2.0	2.3	2.0	1.4
	3	2.1	2.1	2.0	1.4
	1	2.4	2.2	2.0	1.7
	5	2.8	2.5	2.4	1.7
	4	2.1	2.0	1.4	1.4
	9	2.4	2.4	2.4	2.4
9	2	1.5	1.7	1.3	1.2
	5	3.0	2.9	2.3	1.8
	6	2.3	2.2	1.8	1.4
	3	2.2	2.2	2.1	1.5
	7	2.2	1.8	1.6	1.6
	8	2.0	2.2	1.7	1.3
	4	1.9	1.8	1.5	1.3
	1	2.3	2.0	1.9	1.5
	9	2.4	2.4	2.4	2.4
10	8	2.0	1.8	1.5	1.5
	1	2.7	2.0	1.7	1.6
	6	2.1	2.0	1.9	1.5
	7	2.3	1.9	1.8	1.8
	4	2.1	1.9	1.6	1.4
	2	1.7	1.8	1.5	1.2
	9	2.0	2.0	2.0	2.0

Appendix II
Planned Protective Capacity

Resources	Load Distribution or Mean Protective Capacity (MPC)							
	MPC = 10%		MPC = 20%		MPC = 30%		MPC = 40%	
	Percent PC	Utilisation	Percent PC	Utilisation	Percent PC	Utilisation	Percent PC	Utilisation
1	9.60	90.40	22.22	77.78	29.29	70.71	43.43	56.57
2	9.60	90.40	16.16	83.84	29.80	70.20	41.92	58.08
3	9.60	90.40	20.71	79.29	24.75	75.25	42.42	57.58
4	10.10	89.90	15.66	84.34	32.32	67.68	35.35	64.65
5	10.61	89.39	25.25	74.75	34.85	65.15	47.98	52.02
6	10.61	89.39	18.18	81.82	27.27	72.73	39.90	60.10
7	9.60	90.40	22.73	77.27	27.78	72.22	32.32	67.68
8	10.61	89.39	19.19	80.81	34.34	65.66	36.87	63.13
Mean	10.04	89.96	20.01	79.99	30.05	69.95	40.03	59.97

Appendix III
Simulation Program Flow Chart



Appendix IV
Program Listing for the Shop using the FCFS Dispatching Rule

```
Preamble
   Normally mode is undefined

   Processes include CALL.REPORT, STOP.SIMUL, JOB.RELEASE
   Every C.SHOP.FLOOR has
      a C.PRIORITY
      Define C.PRIORITY as a real variable
   Every NC.SHOP.FLOOR has
      a NC.ORDER.TYPE and owns
      a NC.JOB.ORDER
      Define NC.ORDER.TYPE as an integer variable

   Priority order is JOB.RELEASE, NC.SHOP.FLOOR, C.SHOP.FLOOR,
      CALL.REPORT and STOP.SIMUL

   Resources include CONSTRAINT
      Every NON_CONSTRAINT has
      a NC.ID
      Define NC.ID as an integer variable

   Permanent entities
      Every PRODUCT.TYPE has
         a NUM.TASKS,              '' number of tasks
         a TIME.TO.CONSTRAINT,     '' lead time to the constraint
         a TIME.BUFFER,            '' similar to buffer length
         a CONSTRAINT.TIME,        '' constraint processing time
         a DELAY.TIME.MEASURE,     '' delay time to the constraint
         a LATENESS.PRODUCT,       '' lateness for this product
         a CYCLE.TIME.MEASURE      '' a measure of cycle time
         and owns a TASK.SEQUENCE
      Define NUM.TASKS as an integer variable
      Define TIME.TO.CONSTRAINT, TIME.BUFFER, CONSTRAINT.TIME,
         DELAY.TIME.MEASURE, LATENESS.PRODUCT and
         CYCLE.TIME.MEASURE as real variables
      Define TASK.SEQUENCE as a fifo set

   Temporary entities
      Every TASK has
         a PROCESSING.TIME,
         a NON_CONSTRAINT.ID and
         can belong to a TASK.SEQUENCE
      Define PROCESSING.TIME as a real variable
      Define NON_CONSTRAINT.ID as an integer variable

      Every JOB has
         a JOB.NAME,               '' the list of order number
         a BATCH.SIZE,             '' number of orders
         a SEQUENCE.FACTOR,        '' time before a job consumed at CCR
         a ROPE.LENGTH,            '' processing time plus time buffer
         a RELEASE.TIME,           '' at the gating point
         a DUE.DATE,               '' release time + rope + constraint time
         a ENTER.BUFFER.TIME,      '' arriving job at the buffer
         a DELAY.TIME,             '' actual coming minus entering time
         a CONSUMING.TIME,         '' time the constraint start consuming
         a COMPLETION.TIME,        '' time a job is completed
         a LATENESS,               '' completion time minus due date
         a CYCLE.TIME,             '' completion time minus release time
```

```
      a PROD.TYPE and          '' product type
      can belong to
      a NC.JOB.ORDER,
      a BACKLOG.FILE,
      a BUFFER.FILE and
      an ASSEMBLY.SCHEDULE

   Define SEQUENCE.FACTOR, ROPE.LENGTH, RELEASE.TIME,
      DUE.DATE, ENTER.BUFFER.TIME, DELAY.TIME, CONSUMING.TIME,
      COMPLETION.TIME, LATENESS, CYCLE.TIME, NON.CONSTRAINT.TIME
      as real variables
   Define JOB.NAME, BATCH.SIZE, RESOURCE.ID and PROD.TYPE
      as integer variables

The system owns
   a BACKLOG.FILE,
   a BUFFER.FILE and
   an ASSEMBLY.SCHEDULE

   Define BACKLOG.FILE as a set ranked by low RELEASE.TIME
   Define BUFFER.FILE as a set ranked by low CONSUMING.TIME
   Define ASSEMBLY.SCHEDULE as a FIFO set

Define S.JOB as an integer variable
Define DATA.LIST as real, 1-dimensional array
Define N.JOB as an integer variable
Define S.SEQUENCE.FACTOR as a real variable
Define S.JOB.NAME, MAX.BATCH.SIZE, MIN.BATCH.SIZE
   and NO.OF.COMPLETED.JOBS as integer variables

Define NO.OF.JOB.CHANGES as an integer variable
Define NC.JOB.ORDER as a FIFO set

Define REPORT.TIME and RUN.TIME as real variables
Define REPORT.NUMBER as an integer variable

Define BUFFER.MULTIPLIER as an integer variable

Define START.MASTER.SCHEDULE as a real variable
Define TARDY.IN.BUFFER as a real variable
Define MAKE.SPAN as a real variable
Define TARDINESS.MEASURE as a real variable
Define LATENESS.MEASURE as a real variable
Define NO.OF.TARDY.JOBS as an integer variable

Accumulate UTILISATION.C as the average of N.X.CONSTRAINT
Accumulate AVE.QUEUE.LENGTH.C as the average and
   MAX.QUEUE.LENGTH.C as the maximum of N.Q.CONSTRAINT
Accumulate MIN.QUEUE.LENGTH.C as the minimum of N.Q.CONSTRAINT

Accumulate UTILISATION.NC as the average of N.X.NON_CONSTRAINT
Accumulate AVE.QUEUE.LENGTH.NC as the average and
   MAX.QUEUE.LENGTH.NC as the maximum of N.Q.NON_CONSTRAINT
Accumulate MIN.QUEUE.LENGTH.NC as the minimum of
   N.Q.NON_CONSTRAINT

Tally NUM.DELAY.TIME as the num,
   MEAN.DELAY.TIME as the average, MAX.DELAY.TIME as the
   maximum and MIN.DELAY.TIME as the minimum of DELAY.TIME.MEASURE
Tally NUM.LATENESS as the num,
   MEAN.LATENESS as the average, MAX.LATENESS as the
   maximum and MIN.LATENESS as the minimum of LATENESS.PRODUCT
Tally NUM.CYCLE.TIME as the num,
   MEAN.CYCLE.TIME as the average,
   MAX.CYCLE.TIME as the maximum and MIN.CYCLE.TIME as
   the minimum of CYCLE.TIME.MEASURE
Tally NUM.TARDY.IN.BUFFER as the num,
   MEAN.TARDY.IN.BUFFER as the mean,
   MAX.TARDY.IN.BUFFER as the maximum, MIN.TARDY.IN.BUFFER
   as the minimum, VARIATION.TARDY.IN.BUFFER as the std.dev
   of TARDY.IN.BUFFER
Tally NUM.JOBS.TARDY as the num, MEAN.JOBS.TARDY as the mean,
```

```
      MAX.JOBS.TARDY as the maximum, MIN.JOBS.TARDY
      as the minimum, VARIATION.JOBS.TARDY as the std.dev
      of TARDINESS.MEASURE
   Tally MEAN.MAKE.SPAN as the mean of MAKE.SPAN
   Tally MEAN.LATENESS.MEASURE as the mean of LATENESS.MEASURE

End '' Preamble


Main
   Define i as an integer variable
   Call READ.DATA

   Let REPORT.TIME = 1

   Print 1 line thus
      Input: Run time for FCFS (2010 days)
      Read RUN.TIME

   Call INITIALIZE

   Let N.JOB = 100

   For i=1 to N.JOB,
   Do
      Call JOB.GENERATOR
   Loop

   Call RELEASE.TIME.RULES

   Activate a CALL.REPORT in REPORT.TIME days

   Activate a STOP.SIMUL in RUN.TIME days

   Start SIMULATION

End '' Main


Process C.SHOP.FLOOR

   Define C.PROCESSING.TIME as a real variable
   Define PRODUCT.TYPE as an integer variable
   Define MIN.CONSUMING.TIME as a real variable
   Define JOB as a pointer variable
   Define JOB.TO.BE.PROCESSED as an integer variable

   For each JOB in the BUFFER.FILE,
      compute MIN.CONSUMING.TIME as the minimum of
      CONSUMING.TIME(JOB)

   For each JOB in the BUFFER.FILE,
      with CONSUMING.TIME(JOB) = MIN.CONSUMING.TIME
      Find JOB.TO.BE.PROCESSED = the first JOB
      Remove this JOB from the BUFFER.FILE
      Let PRODUCT.TYPE = PROD.TYPE(JOB.TO.BE.PROCESSED)
      Let C.PROCESSING.TIME = BATCH.SIZE(JOB.TO.BE.PROCESSED)*
         CONSTRAINT.TIME(PRODUCT.TYPE)

   Let C.PRIORITY(C.SHOP.FLOOR) = 0 - CONSUMING.TIME(JOB.TO.BE.PROCESSED)

   Request 1 unit of CONSTRAINT with priority C.PRIORITY(C.SHOP.FLOOR)
   Work C.PROCESSING.TIME minutes
   Relinquish 1 unit of CONSTRAINT

   Let COMPLETION.TIME(JOB.TO.BE.PROCESSED) = time.v*24*60

   Let LATENESS(JOB.TO.BE.PROCESSED) =
      COMPLETION.TIME(JOB.TO.BE.PROCESSED) -
      DUE.DATE(JOB.TO.BE.PROCESSED)
   Let LATENESS.PRODUCT(PRODUCT.TYPE) = LATENESS(JOB.TO.BE.PROCESSED)
   Let LATENESS.MEASURE = LATENESS(JOB.TO.BE.PROCESSED)
```

```
    If DUE.DATE(JOB.TO.BE.PROCESSED) < COMPLETION.TIME(JOB.TO.BE.PROCESSED)
       Let TARDINESS.MEASURE = COMPLETION.TIME(JOB.TO.BE.PROCESSED) -
          DUE.DATE(JOB.TO.BE.PROCESSED)
       If TARDINESS.MEASURE >= 10
          Add 1 to NO.OF.TARDY.JOBS
       Always
    Always

    Let CYCLE.TIME(JOB.TO.BE.PROCESSED) =
       COMPLETION.TIME(JOB.TO.BE.PROCESSED) -
       RELEASE.TIME(JOB.TO.BE.PROCESSED)
    Let CYCLE.TIME.MEASURE(PRODUCT.TYPE) = CYCLE.TIME(JOB.TO.BE.PROCESSED)
    Let MAKE.SPAN = CYCLE.TIME(JOB.TO.BE.PROCESSED)

    Let S.JOB = S.JOB + 1
    If S.JOB <> JOB.NAME(JOB.TO.BE.PROCESSED),
       Add 1 to NO.OF.JOB.CHANGES
    Always

    Destroy this JOB called JOB
    Add 1 to NO.OF.COMPLETED.JOBS

End '' Process C.SHOP.FLOOR


Process CALL.REPORT
   Open 2 for output, name is "aout4e.txt"
   Use unit 2 for output
   Until time.v >= RUN.TIME
   Do
       Let REPORT.NUMBER = REPORT.NUMBER + 1
       Call PERIODICAL.REPORT
       Wait 1 days
   Loop

End '' Process CALL.REPORT


Routine INITIALIZE

   Let TIME.V = 0

   For each NON_CONSTRAINT,
   Do
       Let N.X.NON_CONSTRAINT(NON_CONSTRAINT) = 0
       Let N.Q.NON_CONSTRAINT(NON_CONSTRAINT) = 0
   Loop

   Let START.MASTER.SCHEDULE = 0
   Let S.SEQUENCE.FACTOR = 0
   Let N.X.CONSTRAINT(CONSTRAINT) = 0
   Let N.Q.CONSTRAINT(CONSTRAINT) = 0

   Let REPORT.NUMBER = 0

   For each PRODUCT.TYPE,
   Do
      For each TASK in TASK.SEQUENCE(PRODUCT.TYPE)
      Do
         For each NON_CONSTRAINT,
            with NC.ID(NON_CONSTRAINT) = NON_CONSTRAINT.ID(TASK)
         Find the first case
         If found
            Let NON_CONSTRAINT.ID(TASK) = NON_CONSTRAINT
         Always
      Loop
   Loop

   Until the ASSEMBLY.SCHEDULE is empty
   Do
       Remove the first JOB from the ASSEMBLY.SCHEDULE
```

```
        Destroy this JOB called JOB
    Loop

    Until the BACKLOG.FILE is empty
    Do
        Remove the first JOB from the BACKLOG.FILE
        Destroy this JOB called JOB
    Loop

    Until the BUFFER.FILE is empty
    Do
        Remove the first JOB from the BUFFER.FILE
        Destroy this JOB called JOB
    Loop

End '' Routine INITIALIZE


Routine JOB.GENERATOR
   Create a JOB
      Let JOB.NAME(JOB) = S.JOB.NAME + 1
      Let S.JOB.NAME = JOB.NAME(JOB)
      Let PROD.TYPE(JOB) = randi.f(1,N.PRODUCT.TYPE,5)
      Let BATCH.SIZE(JOB) = randi.f(MIN.BATCH.SIZE, MAX.BATCH.SIZE,7)
      Add (CONSTRAINT.TIME(PROD.TYPE(JOB))*BATCH.SIZE(JOB))
         to S.SEQUENCE.FACTOR
      Let SEQUENCE.FACTOR(JOB) = S.SEQUENCE.FACTOR -
         (CONSTRAINT.TIME(PROD.TYPE(JOB))*BATCH.SIZE(JOB))
      Let ROPE.LENGTH(JOB) = BATCH.SIZE(JOB)*
         (1+BUFFER.MULTIPLIER)*TIME.TO.CONSTRAINT(PROD.TYPE(JOB))
      File JOB in the ASSEMBLY.SCHEDULE

End '' Routine JOB.GENERATOR


Process JOB.RELEASE
   Define MIN.RELEASE.TIME as a real variable
   Define WAIT.TO.RELEASE as a real variable
   Define JOB.TO.BE.RELEASED as an integer variable

   Call NEW.JOB.GENERATOR

   For each JOB in the BACKLOG.FILE,
      compute MIN.RELEASE.TIME as the minimum of RELEASE.TIME(JOB)

   For each JOB in the BACKLOG.FILE,
      with RELEASE.TIME(JOB) = MIN.RELEASE.TIME
      Find JOB.TO.BE.RELEASED = the first JOB
      Remove this JOB from the BACKLOG.FILE
      Create a NC.SHOP.FLOOR
      File JOB.TO.BE.RELEASED in NC.JOB.ORDER(NC.SHOP.FLOOR)
      Let WAIT.TO.RELEASE = 0
      For each JOB.TO.BE.RELEASED in the NC.JOB.ORDER(NC.SHOP.FLOOR),
      Do
         Let NC.ORDER.TYPE(NC.SHOP.FLOOR) =
            PROD.TYPE(JOB.TO.BE.RELEASED)
         Let WAIT.TO.RELEASE = RELEASE.TIME(JOB.TO.BE.RELEASED)
      Loop
      If WAIT.TO.RELEASE <= RUN.TIME*60*24,
      Activate this NC.SHOP.FLOOR in WAIT.TO.RELEASE minutes
      Always

End '' JOB.RELEASE


Process NC.SHOP.FLOOR

   Define BUFFER.ARRIVAL as a real variable
   Define PRODUCT.TYPE as an integer variable
   Define NC.PROCESSING.TIME as a real variable
   Define TASK as an integer variable
```

```
    Define SIZE as an integer variable

    For each JOB in NC.JOB.ORDER(NC.SHOP.FLOOR),
       Let SIZE = BATCH.SIZE(JOB)

    Let PRODUCT.TYPE = NC.ORDER.TYPE(NC.SHOP.FLOOR)

    For each TASK in TASK.SEQUENCE(PRODUCT.TYPE),
    Do
       Let NC.PROCESSING.TIME = SIZE*PROCESSING.TIME(TASK)
       Request 1 unit of NON_CONSTRAINT(NON_CONSTRAINT.ID(TASK))
       Work Exponential.f(NC.PROCESSING.TIME, 5) minutes
       Relinquish 1 unit of NON_CONSTRAINT(NON_CONSTRAINT.ID(TASK))
    Loop

    Let BUFFER.ARRIVAL = 0
    Let BUFFER.ARRIVAL = time.v*24*60

    For each JOB in NC.JOB.ORDER(NC.SHOP.FLOOR),
    Do
       Let DELAY.TIME(JOB) = BUFFER.ARRIVAL - ENTER.BUFFER.TIME(JOB)
       If time.v*24*60 > ENTER.BUFFER.TIME(JOB),
          Let TARDY.IN.BUFFER = DELAY.TIME(JOB)
       Always
    Loop

    For each JOB in NC.JOB.ORDER(NC.SHOP.FLOOR),
    Do
       Let DELAY.TIME.MEASURE(PRODUCT.TYPE) = DELAY.TIME(JOB)
    Loop

    For each JOB in the NC.JOB.ORDER(NC.SHOP.FLOOR)
    Do
     Remove this JOB from NC.JOB.ORDER(NC.SHOP.FLOOR)
     File this JOB in the BUFFER.FILE
    Loop

    If time.v*24*60 <= START.MASTER.SCHEDULE,
       Wait START.MASTER.SCHEDULE - time.v*24*60 minutes
    Always

    Create a C.SHOP.FLOOR

    Activate a C.SHOP.FLOOR now

End '' Process NC.SHOP.FLOOR


Routine NEW.JOB.GENERATOR
   Create a JOB
      Let JOB.NAME(JOB) = S.JOB.NAME + 1
      Let S.JOB.NAME = JOB.NAME(JOB)
      Let PROD.TYPE(JOB) = randi.f(1,N.PRODUCT.TYPE,5)
      Let BATCH.SIZE(JOB) = randi.f(MIN.BATCH.SIZE, MAX.BATCH.SIZE,7)
      Add (CONSTRAINT.TIME(PROD.TYPE(JOB))*BATCH.SIZE(JOB))
         to S.SEQUENCE.FACTOR
      Let SEQUENCE.FACTOR(JOB) = S.SEQUENCE.FACTOR -
         (CONSTRAINT.TIME(PROD.TYPE(JOB))*BATCH.SIZE(JOB))
      Let ROPE.LENGTH(JOB) = BATCH.SIZE(JOB)*
         (1+BUFFER.MULTIPLIER)*TIME.TO.CONSTRAINT(PROD.TYPE(JOB))
      Let RELEASE.TIME(JOB) = START.MASTER.SCHEDULE -
         ROPE.LENGTH(JOB) + SEQUENCE.FACTOR(JOB)
      Let ENTER.BUFFER.TIME(JOB) = RELEASE.TIME(JOB) +
         (BATCH.SIZE(JOB)*TIME.TO.CONSTRAINT(PROD.TYPE(JOB)))
      Let CONSUMING.TIME(JOB) = RELEASE.TIME(JOB) +
         ROPE.LENGTH(JOB)
      Let DUE.DATE(JOB) = RELEASE.TIME(JOB) + ROPE.LENGTH(JOB) +
         (BATCH.SIZE(JOB)*CONSTRAINT.TIME(PROD.TYPE(JOB)))

   File JOB in the BACKLOG.FILE
   Return
```

```
End '' Routine NEW.JOB.GENERATOR


Routine PERIODICAL.REPORT

   Define PROPORTION.TARDY as a real variable

   Let PROPORTION.TARDY = (NO.OF.TARDY.JOBS/NO.OF.COMPLETED.JOBS)*100

   Print 1 line with mean.tardy.in.buffer, mean.make.span,
     ave.queue.length.c, proportion.tardy thus
****.** ****.** **.** ***.**

  Reset totals of TARDY.IN.BUFFER, TARDINESS.MEASURE, N.X.CONSTRAINT,
   N.Q.CONSTRAINT, MAKE.SPAN, LATENESS.MEASURE

   Let NO.OF.JOB.CHANGES = 0
   Let NO.OF.COMPLETED.JOBS = 0
   Let NO.OF.TARDY.JOBS = 0

End ''Routine PERIODICAL.REPORT


Routine READ.DATA
   Define INDEX, i and j as integer variables

   N.NON_CONSTRAINT = 8
      Create every NON_CONSTRAINT
         For j=1 to N.NON_CONSTRAINT
            Let u.NON_CONSTRAINT(j) = 1

   For each NON_CONSTRAINT,
   Do
      NC.ID(NON_CONSTRAINT) = NC.ID(NON_CONSTRAINT) + 1
   Loop

   N.CONSTRAINT = 1
      Create every CONSTRAINT
         Let u.CONSTRAINT = 1

   N.PRODUCT.TYPE = 10
      Create every PRODUCT.TYPE

   Reserve DATA.LIST(*) as 160

   Open unit 1 for input, name is "INPUT4.txt"
   Use unit 1 for input
   Let EOF.V = 0
   For INDEX = 1 to 153, Read DATA.LIST(INDEX)
   Let EOF.V = 1

   Let INDEX = 0
   Let INDEX = INDEX + 1
      Let MIN.BATCH.SIZE = DATA.LIST(INDEX)
   Let INDEX = INDEX + 1
      Let MAX.BATCH.SIZE = DATA.LIST(INDEX)
   LET INDEX = INDEX + 1
      Let BUFFER.MULTIPLIER = DATA.LIST(INDEX)

   For each PRODUCT.TYPE,
   Do
      Let INDEX = INDEX + 1
         Let NUM.TASKS(PRODUCT.TYPE) = DATA.LIST(INDEX)
      Let INDEX = INDEX + 1
         CONSTRAINT.TIME(PRODUCT.TYPE) = DATA.LIST(INDEX)

      For i = 1 to NUM.TASKS(PRODUCT.TYPE)
      Do
         Create a TASK
         Let INDEX = INDEX + 1
            Let NON_CONSTRAINT.ID(TASK) = DATA.LIST(INDEX)
         Let INDEX = INDEX + 1
```

```
            Let PROCESSING.TIME(TASK) = DATA.LIST(INDEX)
         File TASK in TASK.SEQUENCE(PRODUCT.TYPE)
         Add PROCESSING.TIME(TASK) to
            TIME.TO.CONSTRAINT(PRODUCT.TYPE)
      Loop

      Let TIME.BUFFER(PRODUCT.TYPE) = BUFFER.MULTIPLIER*
         TIME.TO.CONSTRAINT(PRODUCT.TYPE)
   Loop

   Close unit 1

   Print 3 lines thus
           ----------------------------------------
           DRUM-BUFFER-ROPE SIMULATION OF A JOB-SHOP
           ----------------------------------------
   Skip 1 line
   Print 2 lines thus
           THE EXPERIMENTAL CONDITIONS FOR THIS RUN
           ---------------------------------------
   Print 4 lines with N.PRODUCT.TYPE, BUFFER.MULTIPLIER,
      MIN.BATCH.SIZE, MAX.BATCH.SIZE thus
         Number of Product Types          **  [units]
         Buffer Time Multiplier            *  [constant]
         Minimum Batch Size               **  [units]
         Maximum Batch Size               **  [units]
   Skip 2 lines

   Return

End '' Routine READ.DATA


Routine RELEASE.TIME.RULES
   Define i as an integer variable

   For each JOB in the ASSEMBLY.SCHEDULE,
      Compute START.MASTER.SCHEDULE as the maximum of (ROPE.LENGTH(JOB) -
         SEQUENCE.FACTOR(JOB))

   For each JOB in the ASSEMBLY.SCHEDULE,
   Do
      Let RELEASE.TIME(JOB) = START.MASTER.SCHEDULE -
         ROPE.LENGTH(JOB) + SEQUENCE.FACTOR(JOB)
      If RELEASE.TIME(JOB) < 0,
         Let RELEASE.TIME(JOB) = 0
      Always
      Let ENTER.BUFFER.TIME(JOB) = RELEASE.TIME(JOB) +
         (BATCH.SIZE(JOB)*TIME.TO.CONSTRAINT(PROD.TYPE(JOB)))
      Let CONSUMING.TIME(JOB) = RELEASE.TIME(JOB) +
         ROPE.LENGTH(JOB)
      Let DUE.DATE(JOB) = RELEASE.TIME(JOB) + ROPE.LENGTH(JOB) +
         (BATCH.SIZE(JOB)*CONSTRAINT.TIME(PROD.TYPE(JOB)))
      Remove this JOB from the ASSEMBLY.SCHEDULE
      File this JOB in the BACKLOG.FILE
   Loop

   For i=1 to 102500
   Do
      Activate a JOB.RELEASE now
   Loop

End '' Routine RELEASE.TIME.RULES


Process STOP.SIMUL
   Define THROUGHPUT as a real variable
   Define INDEX as an integer variable

   Open unit 3 for output, name is "Aefi4e.TXT"
   Use unit 3 for output
```

```
Print 3 lines thus
        ----------------------------------------
        DRUM-BUFFER-ROPE SIMULATION OF A JOB-SHOP
        ----------------------------------------
Skip 1 line
Print 2 lines thus
        THE EXPERIMENTAL CONDITIONS FOR THIS RUN
        ---------------------------------------
Print 4 lines with N.PRODUCT.TYPE, BUFFER.MULTIPLIER,
   MIN.BATCH.SIZE, MAX.BATCH.SIZE thus
      Number of Product Types          **  [units]
      Buffer Time Multiplier            *  [constant]
      Minimum Batch Size               **  [units]
      Maximum Batch Size               **  [units]
Skip 2 lines
Print 2 line thus
               PRODUCT TYPE CHARACTERISTICS
               ----------------------------
Skip 1 line
For each PRODUCT.TYPE,
Do
   Print 1 line with PRODUCT.TYPE thus
   PRODUCT TYPE: *
   Print 4 lines thus
   TASK ROUTING:
   ------------------------------------------------
   Task Machine Identity       Task Processing Time
   ------------------------------------------------
   For each TASK in TASK.SEQUENCE(PRODUCT.TYPE),
   Do
      Print 1 line with NON_CONSTRAINT.ID(TASK) and
         PROCESSING.TIME(TASK) thus
           *                          **.**  minutes
   Loop
   Print 1 line thus
   ------------------------------------------------
   Skip 1 line
   Print 3 line with TIME.TO.CONSTRAINT(PRODUCT.TYPE),
      CONSTRAINT.TIME(PRODUCT.TYPE) and
      TIME.BUFFER(PRODUCT.TYPE) thus
   Lead time to the constraint        **.** minutes
   Processing time at the constraint  **.** minutes
   Time buffer for this product       **.** minutes
   Skip 2 lines
Loop

Let THROUGHPUT = NO.OF.COMPLETED.JOBS/RUN.TIME

Start new page
Print 3 lines thus
            -------------------------------------
            FINAL SIMULATION EXPERIMENTAL RESULTS
            -------------------------------------
Skip 2 lines
Print 2 lines thus
GENERAL RESULTS
---------------
Print 1 line with NO.OF.COMPLETED.JOBS thus
Number of completed jobs      *****
Print 1 line with THROUGHPUT thus
Throughput rate             ****.**
Print 1 line with START.MASTER.SCHEDULE thus
Master Schedule starts at   ****.**
Print 1 line with NO.OF.JOB.CHANGES thus
Number of job changes is      *****
Skip 2 lines

Print 2 lines thus
PRODUCT TYPE RESULTS
--------------------
For INDEX = 1 to N.PRODUCT.TYPE,
Do
```

```
      Print 1 line with INDEX thus
   Product type *
   Print 3 lines thus
   ---------------------------------------------------------
   VARIABLE   MEAN-TIME      MAX-TIME    MIN-TIME    NUMBER
   ---------------------------------------------------------
   Print 3 lines with MEAN.DELAY.TIME(INDEX),
         MAX.DELAY.TIME(INDEX), MIN.DELAY.TIME(INDEX),
         NUM.DELAY.TIME(INDEX),
         MEAN.LATENESS(INDEX), MAX.LATENESS(INDEX),
         MIN.LATENESS(INDEX), NUM.LATENESS(INDEX),
         MEAN.CYCLE.TIME(INDEX),MAX.CYCLE.TIME(INDEX),
         MIN.CYCLE.TIME(INDEX) and NUM.CYCLE.TIME(INDEX) thus
   DELAY TIME  *****.***    *****.***    *****.***    *****
   LATENESS    *****.***    *****.***    *****.***    *****
   CYCLE TIME  *****.***    *****.***    *****.***    *****
   Print 1 line thus
   ---------------------------------------------------------
   Loop
   Skip 3 lines
   Print 3 lines thus
              WORK STATION PERFORMANCE INFORMATION
              ====================================
                   (SIX WORK STATION JOB SHOP)
   Skip 2 lines
   Print 3 lines thus
   ---------------------------------------------------------------
   RESOURSE  UTILIZATION  MEAN-QUEUE  MAXIMUM-QUEUE  MINIMUM-QUEUE
   ---------------------------------------------------------------
   Print 1 line with UTILISATION.C, AVE.QUEUE.LENGTH.C,
      MAX.QUEUE.LENGTH.C and MIN.QUEUE.LENGTH.C thus
   Constraint  ***.***     ****.***      ****.***      ****.***
   Print 1 line thus
   Non-constraint
   For INDEX=1 to N.NON_CONSTRAINT,
   Do
      Print 1 line with INDEX, UTILISATION.NC(INDEX),
         AVE.QUEUE.LENGTH.NC(INDEX), MAX.QUEUE.LENGTH.NC(INDEX)
         and MIN.QUEUE.LENGTH.NC(INDEX) thus
      *        ***.***     ****.***      ****.***      ****.***
   Loop
   Print 1 line thus
   ----------------------------------------------------------------
   Skip 4 lines
   STOP

End ''Process STOP.SIMUL
```

Appendix V
Program Listing for the FCFS Shop using the Dynamic Expediting Rule

```
Preamble
   Normally mode is undefined

   Processes include CALL.REPORT, STOP.SIMUL, JOB.RELEASE
   Every C.SHOP.FLOOR has
      a C.PRIORITY
      Define C.PRIORITY as a real variable
   Every NC.SHOP.FLOOR has
      a NC.ORDER.TYPE,
      a NC.PRIORITY,
      a REMAINING.PROCESSING and owns
      a NC.JOB.ORDER
      Define NC.ORDER.TYPE as an integer variable
      Define NC.PRIORITY, REMAINING.PROCESSING as real variables

   Priority order is JOB.RELEASE, NC.SHOP.FLOOR, C.SHOP.FLOOR,
      CALL.REPORT and STOP.SIMUL

   Resources include CONSTRAINT
      Every NON_CONSTRAINT has
      a NC.ID
      Define NC.ID as an integer variable

   Permanent entities
      Every PRODUCT.TYPE has
         a NUM.TASKS,            '' number of tasks
         a TIME.TO.CONSTRAINT,   '' lead time to the constraint
         a TIME.BUFFER,          '' similar to buffer length
         a CONSTRAINT.TIME,      '' constraint processing time
         a DELAY.TIME.MEASURE,   '' delay time to the constraint
         a LATENESS.PRODUCT,     '' lateness for this product
         a CYCLE.TIME.MEASURE    '' a measure of cycle time
         and owns a TASK.SEQUENCE
      Define NUM.TASKS as an integer variable
      Define TIME.TO.CONSTRAINT, TIME.BUFFER, CONSTRAINT.TIME,
         DELAY.TIME.MEASURE, LATENESS.PRODUCT and
         CYCLE.TIME.MEASURE as real variables
      Define TASK.SEQUENCE as a fifo set

   Temporary entities
      Every TASK has
         a PROCESSING.TIME,
         a NON_CONSTRAINT.ID and
         can belong to a TASK.SEQUENCE
      Define PROCESSING.TIME as a real variable
      Define NON_CONSTRAINT.ID as an integer variable

      Every JOB has
         a JOB.NAME,             '' the list of order number
         a BATCH.SIZE,           '' number of orders
         a SEQUENCE.FACTOR,      '' time before a job consumed at CCR
         a ROPE.LENGTH,          '' processing time plus time buffer
         a RELEASE.TIME,         '' at the gating point
         a CHECKING.TIME,        '' release time + proc.time + 1/2 buffer
         a DUE.DATE,             '' release time + rope + constraint time
         a ENTER.BUFFER.TIME,    '' arriving job at the buffer
         a DELAY.TIME,           '' actual coming minus entering time
```

```
      a CONSUMING.TIME,        '' time the constraint start consuming
      a COMPLETION.TIME,       '' time a job is completed
      a LATENESS,              '' completion time minus due date
      a CYCLE.TIME,            '' completion time minus release time
      a PROD.TYPE and          '' product type
      can belong to
      a NC.JOB.ORDER,
      a BACKLOG.FILE,
      a BUFFER.FILE and
      an ASSEMBLY.SCHEDULE

   Define SEQUENCE.FACTOR, ROPE.LENGTH, RELEASE.TIME, CHECKING.TIME,
      DUE.DATE, ENTER.BUFFER.TIME, DELAY.TIME, CONSUMING.TIME,
      COMPLETION.TIME, LATENESS, CYCLE.TIME, NON.CONSTRAINT.TIME
      as real variables
   Define JOB.NAME, BATCH.SIZE, RESOURCE.ID and PROD.TYPE
      as integer variables

The system owns
   a BACKLOG.FILE,
   a BUFFER.FILE and
   an ASSEMBLY.SCHEDULE

   Define BACKLOG.FILE as a set ranked by low RELEASE.TIME
   Define BUFFER.FILE as a set ranked by low CONSUMING.TIME
   Define ASSEMBLY.SCHEDULE as a FIFO set

Define S.JOB as an integer variable
Define DATA.LIST as real, 1-dimensional array
Define N.JOB as an integer variable
Define S.SEQUENCE.FACTOR as a real variable
Define S.JOB.NAME, MAX.BATCH.SIZE, MIN.BATCH.SIZE
   and NO.OF.COMPLETED.JOBS as integer variables

Define NO.OF.JOB.CHANGES as an integer variable
Define NO.OF.EXPEDITING as an integer variable
Define NC.JOB.ORDER as a FIFO set

Define REPORT.TIME and RUN.TIME as real variables
Define REPORT.NUMBER as an integer variable

Define BUFFER.MULTIPLIER as an integer variable

Define START.MASTER.SCHEDULE as a real variable
Define TARDY.IN.BUFFER as a real variable
Define MAKE.SPAN as a real variable
Define TARDINESS.MEASURE as a real variable
Define LATENESS.MEASURE as a real variable
Define NO.OF.TARDY.JOBS as an integer variable

Accumulate UTILISATION.C as the average of N.X.CONSTRAINT
Accumulate AVE.QUEUE.LENGTH.C as the average and
   MAX.QUEUE.LENGTH.C as the maximum of N.Q.CONSTRAINT
Accumulate MIN.QUEUE.LENGTH.C as the minimum of N.Q.CONSTRAINT

Accumulate UTILISATION.NC as the average of N.X.NON_CONSTRAINT
Accumulate AVE.QUEUE.LENGTH.NC as the average and
   MAX.QUEUE.LENGTH.NC as the maximum of N.Q.NON_CONSTRAINT
Accumulate MIN.QUEUE.LENGTH.NC as the minimum of
   N.Q.NON_CONSTRAINT

Tally NUM.DELAY.TIME as the num,
   MEAN.DELAY.TIME as the average, MAX.DELAY.TIME as the
   maximum and MIN.DELAY.TIME as the minimum of DELAY.TIME.MEASURE
Tally NUM.LATENESS as the num,
   MEAN.LATENESS as the average, MAX.LATENESS as the
   maximum and MIN.LATENESS as the minimum of LATENESS.PRODUCT
Tally NUM.CYCLE.TIME as the num,
   MEAN.CYCLE.TIME as the average,
   MAX.CYCLE.TIME as the maximum and MIN.CYCLE.TIME as
   the minimum of CYCLE.TIME.MEASURE
Tally NUM.TARDY.IN.BUFFER as the num,
```

```
      MEAN.TARDY.IN.BUFFER as the mean,
      MAX.TARDY.IN.BUFFER as the maximum, MIN.TARDY.IN.BUFFER
      as the minimum, VARIATION.TARDY.IN.BUFFER as the std.dev
      of TARDY.IN.BUFFER
   Tally NUM.JOBS.TARDY as the num, MEAN.JOBS.TARDY as the mean,
      MAX.JOBS.TARDY as the maximum, MIN.JOBS.TARDY
      as the minimum, VARIATION.JOBS.TARDY as the std.dev
      of TARDINESS.MEASURE
   Tally MEAN.MAKE.SPAN as the mean of MAKE.SPAN
   Tally MEAN.LATENESS.MEASURE as the mean of LATENESS.MEASURE

End '' Preamble


Main
   Define i as an integer variable
   Call READ.DATA

   Let REPORT.TIME = 1

   Print 1 line thus
      Input: Run time for FCFS [301 days]
      Read RUN.TIME

   Call INITIALIZE

   Let N.JOB = 100

   For i=1 to N.JOB,
   Do
      Call JOB.GENERATOR
   Loop

   Call RELEASE.TIME.RULES

   Activate a CALL.REPORT in REPORT.TIME days

   Activate a STOP.SIMUL in RUN.TIME days

   Start SIMULATION

End '' Main


Process C.SHOP.FLOOR
   Define C.PROCESSING.TIME as a real variable
   Define PRODUCT.TYPE as an integer variable
   Define MIN.CONSUMING.TIME as a real variable
   Define JOB as a pointer variable
   Define JOB.TO.BE.PROCESSED as an integer variable

   For each JOB in the BUFFER.FILE,
      compute MIN.CONSUMING.TIME as the minimum of
      CONSUMING.TIME(JOB)

   For each JOB in the BUFFER.FILE,
      with CONSUMING.TIME(JOB) = MIN.CONSUMING.TIME
      Find JOB.TO.BE.PROCESSED = the first JOB
      Remove this JOB from the BUFFER.FILE
      Let PRODUCT.TYPE = PROD.TYPE(JOB.TO.BE.PROCESSED)
      Let C.PROCESSING.TIME = BATCH.SIZE(JOB.TO.BE.PROCESSED)*
         CONSTRAINT.TIME(PRODUCT.TYPE)

   Let C.PRIORITY(C.SHOP.FLOOR) = (0 - CONSUMING.TIME(JOB.TO.BE.PROCESSED))

   Request 1 unit of CONSTRAINT with priority C.PRIORITY(C.SHOP.FLOOR)
   Work C.PROCESSING.TIME minutes
   Relinquish 1 unit of CONSTRAINT

   Let COMPLETION.TIME(JOB.TO.BE.PROCESSED) = time.v*24*60

   Let LATENESS(JOB.TO.BE.PROCESSED) =
```

```
      COMPLETION.TIME(JOB.TO.BE.PROCESSED) -
      DUE.DATE(JOB.TO.BE.PROCESSED)
   Let LATENESS.PRODUCT(PRODUCT.TYPE) = LATENESS(JOB.TO.BE.PROCESSED)
   Let LATENESS.MEASURE = LATENESS(JOB.TO.BE.PROCESSED)

   If DUE.DATE(JOB.TO.BE.PROCESSED) < COMPLETION.TIME(JOB.TO.BE.PROCESSED)
      Let TARDINESS.MEASURE = COMPLETION.TIME(JOB.TO.BE.PROCESSED) -
         DUE.DATE(JOB.TO.BE.PROCESSED)
      If TARDINESS.MEASURE >= 10
         Add 1 to NO.OF.TARDY.JOBS
      Always
   Always

   Let CYCLE.TIME(JOB.TO.BE.PROCESSED) =
      COMPLETION.TIME(JOB.TO.BE.PROCESSED) -
      RELEASE.TIME(JOB.TO.BE.PROCESSED)
   Let CYCLE.TIME.MEASURE(PRODUCT.TYPE) = CYCLE.TIME(JOB.TO.BE.PROCESSED)
   Let MAKE.SPAN = CYCLE.TIME(JOB.TO.BE.PROCESSED)

   Let S.JOB = S.JOB + 1
   If S.JOB <> JOB.NAME(JOB.TO.BE.PROCESSED),
      Add 1 to NO.OF.JOB.CHANGES
   Always

   Destroy this JOB called JOB
   Add 1 to NO.OF.COMPLETED.JOBS

End '' Process C.SHOP.FLOOR


Process CALL.REPORT
   Open 2 for output, name is "Exped.txt"
   Use unit 2 for output
   Until time.v >= RUN.TIME
   Do
       Let REPORT.NUMBER = REPORT.NUMBER + 1
       Call PERIODICAL.REPORT
       Wait 1 days
   Loop

End '' Process CALL.REPORT


Routine INITIALIZE
   Let TIME.V = 0

   For each NON_CONSTRAINT,
   Do
       Let N.X.NON_CONSTRAINT(NON_CONSTRAINT) = 0
       Let N.Q.NON_CONSTRAINT(NON_CONSTRAINT) = 0
   Loop

   Let START.MASTER.SCHEDULE = 0
   Let S.SEQUENCE.FACTOR = 0
   Let N.X.CONSTRAINT(CONSTRAINT) = 0
   Let N.Q.CONSTRAINT(CONSTRAINT) = 0

   Let REPORT.NUMBER = 0

   For each PRODUCT.TYPE,
   Do
      For each TASK in TASK.SEQUENCE(PRODUCT.TYPE)
      Do
         For each NON_CONSTRAINT,
            with NC.ID(NON_CONSTRAINT) = NON_CONSTRAINT.ID(TASK)
         Find the first case
         If found
            Let NON_CONSTRAINT.ID(TASK) = NON_CONSTRAINT
         Always
      Loop
   Loop
```

```
   Until the ASSEMBLY.SCHEDULE is empty
   Do
       Remove the first JOB from the ASSEMBLY.SCHEDULE
       Destroy this JOB called JOB
   Loop

   Until the BACKLOG.FILE is empty
   Do
       Remove the first JOB from the BACKLOG.FILE
       Destroy this JOB called JOB
   Loop

   Until the BUFFER.FILE is empty
   Do
       Remove the first JOB from the BUFFER.FILE
       Destroy this JOB called JOB
   Loop

End '' Routine INITIALIZE


Routine JOB.GENERATOR
   Create a JOB
      Let JOB.NAME(JOB) = S.JOB.NAME + 1
      Let S.JOB.NAME = JOB.NAME(JOB)
      Let PROD.TYPE(JOB) = randi.f(1,N.PRODUCT.TYPE,5)
      Let BATCH.SIZE(JOB) = randi.f(MIN.BATCH.SIZE, MAX.BATCH.SIZE,7)
      Add (CONSTRAINT.TIME(PROD.TYPE(JOB))*BATCH.SIZE(JOB))
         to S.SEQUENCE.FACTOR
      Let SEQUENCE.FACTOR(JOB) = S.SEQUENCE.FACTOR -
         (CONSTRAINT.TIME(PROD.TYPE(JOB))*BATCH.SIZE(JOB))
      Let ROPE.LENGTH(JOB) = BATCH.SIZE(JOB)*
         (1+BUFFER.MULTIPLIER)*TIME.TO.CONSTRAINT(PROD.TYPE(JOB))
      File JOB in the ASSEMBLY.SCHEDULE

End '' Routine JOB.GENERATOR


Process JOB.RELEASE
   Define MIN.RELEASE.TIME as a real variable
   Define WAIT.TO.RELEASE as a real variable
   Define JOB.TO.BE.RELEASED as an integer variable

   Call NEW.JOB.GENERATOR

   For each JOB in the BACKLOG.FILE,
      compute MIN.RELEASE.TIME as the minimum of RELEASE.TIME(JOB)

   For each JOB in the BACKLOG.FILE,
      with RELEASE.TIME(JOB) = MIN.RELEASE.TIME
      Find JOB.TO.BE.RELEASED = the first JOB
      Remove this JOB from the BACKLOG.FILE
      Create a NC.SHOP.FLOOR
      File JOB.TO.BE.RELEASED in NC.JOB.ORDER(NC.SHOP.FLOOR)
      Let WAIT.TO.RELEASE = 0

      For each JOB.TO.BE.RELEASED in the NC.JOB.ORDER(NC.SHOP.FLOOR),
      Do
         Let NC.ORDER.TYPE(NC.SHOP.FLOOR) =
            PROD.TYPE(JOB.TO.BE.RELEASED)
         Let WAIT.TO.RELEASE = RELEASE.TIME(JOB.TO.BE.RELEASED)
         Let PRODUCT.TYPE = NC.ORDER.TYPE(NC.SHOP.FLOOR)
         Let REMAINING.PROCESSING(NC.SHOP.FLOOR) =
            TIME.TO.CONSTRAINT(PROD.TYPE(JOB.TO.BE.RELEASED))*
            BATCH.SIZE(JOB.TO.BE.RELEASED)
         Let NC.PRIORITY(NC.SHOP.FLOOR) =
            (CHECKING.TIME(JOB.TO.BE.RELEASED) -
            RELEASE.TIME(JOB.TO.BE.RELEASED)) -
            REMAINING.PROCESSING(NC.SHOP.FLOOR)
      Loop

      If WAIT.TO.RELEASE <= RUN.TIME*60*24,
```

```
      Activate this NC.SHOP.FLOOR in WAIT.TO.RELEASE minutes
      Always

End '' JOB.RELEASE


Process NC.SHOP.FLOOR
   Define BUFFER.ARRIVAL, NC.PROCESSING.TIME, CHK.TIME,
      JOB.PRIORITY as real variables
   Define TASK, PRODUCT.TYPE, SIZE as integer variables
   Define delta as a real variable

   For each JOB in NC.JOB.ORDER(NC.SHOP.FLOOR),
   Do
      Let SIZE = BATCH.SIZE(JOB)
      Let CHK.TIME = CHECKING.TIME(JOB)
   Loop

   Let PRODUCT.TYPE = NC.ORDER.TYPE(NC.SHOP.FLOOR)

   For each TASK in TASK.SEQUENCE(PRODUCT.TYPE),
   Do
      Let NC.PROCESSING.TIME = SIZE*PROCESSING.TIME(TASK)
      Let Delta = (CHK.TIME - remaining.processing(nc.shop.floor))

      If time.v*24*60 < delta
         Let JOB.PRIORITY = 0
      Always

      If time.v*24*60 >= delta
         Let JOB.PRIORITY = (3500 - NC.PRIORITY(NC.SHOP.FLOOR))
         Add 1 to NO.OF.EXPEDITING
      Always

      Request 1 unit of NON_CONSTRAINT(NON_CONSTRAINT.ID(TASK))
         with priority JOB.PRIORITY
      Work Exponential.f(NC.PROCESSING.TIME, 5) minutes
      Relinquish 1 unit of NON_CONSTRAINT(NON_CONSTRAINT.ID(TASK))

      Let REMAINING.PROCESSING(NC.SHOP.FLOOR) =
         (REMAINING.PROCESSING(NC.SHOP.FLOOR) - NC.PROCESSING.TIME)

      Let NC.PRIORITY(NC.SHOP.FLOOR) = (CHK.TIME - time.v*24*60 -
          REMAINING.PROCESSING(NC.SHOP.FLOOR))

   Loop

   Let BUFFER.ARRIVAL = 0
   Let BUFFER.ARRIVAL = time.v*24*60

   For each JOB in NC.JOB.ORDER(NC.SHOP.FLOOR),
   Do
      Let DELAY.TIME(JOB) = BUFFER.ARRIVAL - ENTER.BUFFER.TIME(JOB)
      If time.v*24*60 > ENTER.BUFFER.TIME(JOB),
         Let TARDY.IN.BUFFER = DELAY.TIME(JOB)
      Always
   Loop

   For each JOB in NC.JOB.ORDER(NC.SHOP.FLOOR),
   Do
      Let DELAY.TIME.MEASURE(PRODUCT.TYPE) = DELAY.TIME(JOB)
   Loop

   For each JOB in the NC.JOB.ORDER(NC.SHOP.FLOOR)
   Do
    Remove this JOB from NC.JOB.ORDER(NC.SHOP.FLOOR)
    File this JOB in the BUFFER.FILE
   Loop

   If time.v*24*60 <= START.MASTER.SCHEDULE,
      Wait START.MASTER.SCHEDULE - time.v*24*60 minutes
   Always
```

```
   Create a C.SHOP.FLOOR

   Activate a C.SHOP.FLOOR now

End '' Process NC.SHOP.FLOOR


Routine NEW.JOB.GENERATOR
   Create a JOB
      Let JOB.NAME(JOB) = S.JOB.NAME + 1
      Let S.JOB.NAME = JOB.NAME(JOB)
      Let PROD.TYPE(JOB) = randi.f(1,N.PRODUCT.TYPE,5)
      Let BATCH.SIZE(JOB) = randi.f(MIN.BATCH.SIZE, MAX.BATCH.SIZE,7)
      Add (CONSTRAINT.TIME(PROD.TYPE(JOB))*BATCH.SIZE(JOB))
         to S.SEQUENCE.FACTOR
      Let SEQUENCE.FACTOR(JOB) = S.SEQUENCE.FACTOR -
         (CONSTRAINT.TIME(PROD.TYPE(JOB))*BATCH.SIZE(JOB))
      Let ROPE.LENGTH(JOB) = BATCH.SIZE(JOB)*
         (1+BUFFER.MULTIPLIER)*TIME.TO.CONSTRAINT(PROD.TYPE(JOB))
      Let RELEASE.TIME(JOB) = START.MASTER.SCHEDULE -
         ROPE.LENGTH(JOB) + SEQUENCE.FACTOR(JOB)
      Let ENTER.BUFFER.TIME(JOB) = RELEASE.TIME(JOB) +
         (BATCH.SIZE(JOB)*TIME.TO.CONSTRAINT(PROD.TYPE(JOB)))
      Let CONSUMING.TIME(JOB) = RELEASE.TIME(JOB) +
         ROPE.LENGTH(JOB)
      Let CHECKING.TIME(JOB) = RELEASE.TIME(JOB) +
         0.5*(ROPE.LENGTH(JOB) - (BATCH.SIZE(JOB)*
            TIME.TO.CONSTRAINT(PROD.TYPE(JOB))))
      Let DUE.DATE(JOB) = RELEASE.TIME(JOB) + ROPE.LENGTH(JOB) +
         (BATCH.SIZE(JOB)*CONSTRAINT.TIME(PROD.TYPE(JOB)))

   File JOB in the BACKLOG.FILE
   Return

End '' Routine NEW.JOB.GENERATOR


Routine PERIODICAL.REPORT
   Define PROPORTION.TARDY as a real variable

   Let PROPORTION.TARDY = (NO.OF.TARDY.JOBS/NO.OF.COMPLETED.JOBS)*100

   Print 1 line with MEAN.TARDY.IN.BUFFER, MEAN.MAKE.SPAN,
      AVE.QUEUE.LENGTH.C, PROPORTION.TARDY, NO.OF.EXPEDITING thus
   ****.** ****.** **.** ***.** *****

   Reset totals of TARDY.IN.BUFFER, TARDINESS.MEASURE, N.X.CONSTRAINT,
      N.Q.CONSTRAINT, MAKE.SPAN, LATENESS.MEASURE

   Let NO.OF.JOB.CHANGES = 0
   Let NO.OF.COMPLETED.JOBS = 0
   Let NO.OF.TARDY.JOBS = 0
   Let NO.OF.EXPEDITING = 0

End ''Routine PERIODICAL.REPORT


Routine READ.DATA
   Define INDEX, i and j as integer variables

   N.NON_CONSTRAINT = 8
      Create every NON_CONSTRAINT
         For j=1 to N.NON_CONSTRAINT
            Let u.NON_CONSTRAINT(j) = 1

   For each NON_CONSTRAINT,
   Do
      NC.ID(NON_CONSTRAINT) = NC.ID(NON_CONSTRAINT) + 1
   Loop

   N.CONSTRAINT = 1
```

```
      Create every CONSTRAINT
         Let u.CONSTRAINT = 1

   N.PRODUCT.TYPE = 10
      Create every PRODUCT.TYPE

   Reserve DATA.LIST(*) as 160

   Open unit 1 for input, name is "INPUT1.txt"
   Use unit 1 for input
   Let EOF.V = 0
   For INDEX = 1 to 153, Read DATA.LIST(INDEX)
   Let EOF.V = 1

   Let INDEX = 0
   Let INDEX = INDEX + 1
      Let MIN.BATCH.SIZE = DATA.LIST(INDEX)
   Let INDEX = INDEX + 1
      Let MAX.BATCH.SIZE = DATA.LIST(INDEX)
   LET INDEX = INDEX + 1
      Let BUFFER.MULTIPLIER = DATA.LIST(INDEX)

   For each PRODUCT.TYPE,
   Do
      Let INDEX = INDEX + 1
         Let NUM.TASKS(PRODUCT.TYPE) = DATA.LIST(INDEX)
      Let INDEX = INDEX + 1
         CONSTRAINT.TIME(PRODUCT.TYPE) = DATA.LIST(INDEX)

      For i = 1 to NUM.TASKS(PRODUCT.TYPE)
      Do
         Create a TASK
         Let INDEX = INDEX + 1
            Let NON_CONSTRAINT.ID(TASK) = DATA.LIST(INDEX)
         Let INDEX = INDEX + 1
            Let PROCESSING.TIME(TASK) = DATA.LIST(INDEX)
         File TASK in TASK.SEQUENCE(PRODUCT.TYPE)
         Add PROCESSING.TIME(TASK) to
            TIME.TO.CONSTRAINT(PRODUCT.TYPE)
      Loop

      Let TIME.BUFFER(PRODUCT.TYPE) = BUFFER.MULTIPLIER*
         TIME.TO.CONSTRAINT(PRODUCT.TYPE)
   Loop

   Close unit 1

   Print 3 lines thus
           ----------------------------------------
           DRUM-BUFFER-ROPE SIMULATION OF A JOB-SHOP
           ----------------------------------------
   Skip 1 line
   Print 2 lines thus
           THE EXPERIMENTAL CONDITIONS FOR THIS RUN
           ----------------------------------------
   Print 4 lines with N.PRODUCT.TYPE, BUFFER.MULTIPLIER,
      MIN.BATCH.SIZE, MAX.BATCH.SIZE thus
         Number of Product Types          **  [units]
         Buffer Time Multiplier            *  [constant]
         Minimum Batch Size               **  [units]
         Maximum Batch Size               **  [units]
   Skip 2 lines

   Return

End '' Routine READ.DATA


Routine RELEASE.TIME.RULES
   Define i as an integer variable

   For each JOB in the ASSEMBLY.SCHEDULE,
```

```
      Compute START.MASTER.SCHEDULE as the maximum of (ROPE.LENGTH(JOB) -
         SEQUENCE.FACTOR(JOB))

   For each JOB in the ASSEMBLY.SCHEDULE,
   Do
      Let RELEASE.TIME(JOB) = START.MASTER.SCHEDULE -
         ROPE.LENGTH(JOB) + SEQUENCE.FACTOR(JOB)
      If RELEASE.TIME(JOB) < 0,
         Let RELEASE.TIME(JOB) = 0
      Always
      Let ENTER.BUFFER.TIME(JOB) = RELEASE.TIME(JOB) +
         (BATCH.SIZE(JOB)*TIME.TO.CONSTRAINT(PROD.TYPE(JOB)))
      Let CONSUMING.TIME(JOB) = RELEASE.TIME(JOB) +
         ROPE.LENGTH(JOB)
      Let CHECKING.TIME(JOB) = RELEASE.TIME(JOB) +
         0.5*(ROPE.LENGTH(JOB) - (BATCH.SIZE(JOB)*
            TIME.TO.CONSTRAINT(PROD.TYPE(JOB))))
      Let DUE.DATE(JOB) = RELEASE.TIME(JOB) + ROPE.LENGTH(JOB) +
         (BATCH.SIZE(JOB)*CONSTRAINT.TIME(PROD.TYPE(JOB)))
      Remove this JOB from the ASSEMBLY.SCHEDULE
      File this JOB in the BACKLOG.FILE
   Loop

   For i=1 to 15000
   Do
      Activate a JOB.RELEASE now
   Loop

End '' Routine RELEASE.TIME.RULES


Process STOP.SIMUL
   Define THROUGHPUT as a real variable
   Define INDEX as an integer variable

   Open unit 3 for output, name is "Final.TXT"
   Use unit 3 for output

   Print 3 lines thus
           ----------------------------------------
           DRUM-BUFFER-ROPE SIMULATION OF A JOB-SHOP
           ----------------------------------------
   Skip 1 line
   Print 2 lines thus
           THE EXPERIMENTAL CONDITIONS FOR THIS RUN
           ----------------------------------------
   Print 4 lines with N.PRODUCT.TYPE, BUFFER.MULTIPLIER,
      MIN.BATCH.SIZE, MAX.BATCH.SIZE thus
         Number of Product Types          **  [units]
         Buffer Time Multiplier            *  [constant]
         Minimum Batch Size               **  [units]
         Maximum Batch Size               **  [units]
   Skip 2 lines
   Print 2 line thus
                  PRODUCT TYPE CHARACTERISTICS
                  ----------------------------
   Skip 1 line
   For each PRODUCT.TYPE,
   Do
      Print 1 line with PRODUCT.TYPE thus
      PRODUCT TYPE: *
      Print 4 lines thus
      TASK ROUTING:
      ----------------------------------------------
      Task Machine Identity       Task Processing Time
      ----------------------------------------------
      For each TASK in TASK.SEQUENCE(PRODUCT.TYPE),
      Do
         Print 1 line with NON_CONSTRAINT.ID(TASK) and
            PROCESSING.TIME(TASK) thus
               *                          **.**  minutes
      Loop
```

```
   Print 1 line thus
   ------------------------------------------------
   Skip 1 line
   Print 3 line with TIME.TO.CONSTRAINT(PRODUCT.TYPE),
      CONSTRAINT.TIME(PRODUCT.TYPE) and
      TIME.BUFFER(PRODUCT.TYPE) thus
   Lead time to the constraint        **.** minutes
   Processing time at the constraint  **.** minutes
   Time buffer for this product       **.** minutes
   Skip 2 lines
Loop

Let THROUGHPUT = NO.OF.COMPLETED.JOBS/RUN.TIME

Start new page
Print 3 lines thus
            -------------------------------------
            FINAL SIMULATION EXPERIMENTAL RESULTS
            -------------------------------------
Skip 2 lines
Print 2 lines thus
GENERAL RESULTS
---------------
Print 1 line with NO.OF.COMPLETED.JOBS thus
Number of completed jobs      *****
Print 1 line with THROUGHPUT thus
Throughput rate             ****.**
Print 1 line with START.MASTER.SCHEDULE thus
Master Schedule starts at   ****.**
Print 1 line with NO.OF.JOB.CHANGES thus
Number of job changes is      *****
Skip 2 lines

Print 2 lines thus
PRODUCT TYPE RESULTS
--------------------
For INDEX = 1 to N.PRODUCT.TYPE,
Do
   Print 1 line with INDEX thus
Product type *
Print 3 lines thus
---------------------------------------------------------
VARIABLE   MEAN-TIME      MAX-TIME    MIN-TIME    NUMBER
---------------------------------------------------------
Print 3 lines with MEAN.DELAY.TIME(INDEX),
      MAX.DELAY.TIME(INDEX), MIN.DELAY.TIME(INDEX),
      NUM.DELAY.TIME(INDEX),
      MEAN.LATENESS(INDEX), MAX.LATENESS(INDEX),
      MIN.LATENESS(INDEX), NUM.LATENESS(INDEX),
      MEAN.CYCLE.TIME(INDEX),MAX.CYCLE.TIME(INDEX),
      MIN.CYCLE.TIME(INDEX) and NUM.CYCLE.TIME(INDEX) thus
DELAY TIME  *****.***    *****.***    *****.***    *****
LATENESS    *****.***    *****.***    *****.***    *****
CYCLE TIME  *****.***    *****.***    *****.***    *****
Print 1 line thus
---------------------------------------------------------
Loop
Skip 3 lines
Print 3 lines thus
           WORK STATION PERFORMANCE INFORMATION
           ====================================
               (NINE WORK STATION JOB SHOP)
Skip 2 lines
Print 3 lines thus
---------------------------------------------------------------
RESOURSE  UTILIZATION  MEAN-QUEUE  MAXIMUM-QUEUE  MINIMUM-QUEUE
---------------------------------------------------------------
Print 1 line with UTILISATION.C, AVE.QUEUE.LENGTH.C,
   MAX.QUEUE.LENGTH.C and MIN.QUEUE.LENGTH.C thus
Constraint  ***.***     ****.***      ****.***      ****.***
Print 1 line thus
Non-constraint
```

```
   For INDEX=1 to N.NON_CONSTRAINT,
   Do
      Print 1 line with INDEX, UTILISATION.NC(INDEX),
         AVE.QUEUE.LENGTH.NC(INDEX), MAX.QUEUE.LENGTH.NC(INDEX)
         and MIN.QUEUE.LENGTH.NC(INDEX) thus
      *        ***.***     ****.***      ****.***      ****.***
   Loop
   Print 1 line thus
   ----------------------------------------------------------------
   Skip 4 lines
   STOP

End ''Process STOP.SIMUL
```