PRACTICAL BUFFER SIZING TECHNIQUES UNDER DRUM-BUFFER-ROPE: DEVELOPMENT OF A MODEL AND FUZZY LOGIC IMPLEMENTATION

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

Jeffrey Lawrence Foote

1996

In Memory of Louis Borman

(1913 - 1994)

Acknowledgements

I would like to sincerely thank Dr. Simon Hurley for his expert know-how, patience and support during the last two years. His infectious enthusiasm for constraints and belief that research is about solving real problems has helped in many ways to see this project to completion.

I would also like to sincerely thank Dr. Adrian Evans for his technical expertise and many insightful and helpful comments, not to mention his sense of humour that often been appreciated.

Without their help, encouragement and excellent supervision, this research would have not been possible.

I would also like to thank Chatu Lokuge, for his expert programming skills and Togar Simatupang for his cheerful optimism.

Thanks are also due to Massey University for their financial support; the Department of Production Technology for its excellent study facilities; Prof. Bob Hodgson for his support that enabled me to travel to the University of Houston, Texas; and fellow postgraduate students for conversation and friendship.

For flatmates who put up with me during the final stages of this project and friends for their interest, never-ending support and encouragement.

Finally, I would to thank my parents for their on-going support during my time as a student.

Abstract

A production buffer is a queue of work waiting in front of a manufacturing work-station for processing. The buffer protects the work-station utilisation from variability in the flow of work from feeding work-stations. Effective buffer management is critical to the smooth flow of work and the maintenance of a predictable output rate.

An effective buffer management policy must address three questions that characterise the buffer management problem (BMP):

1. What objective function to use?
2. Where to locate buffers?
3. What is the appropriate buffer sizes?

Despite being simple to describe, to date few practical heuristics for buffer management have been developed by researchers. The approach of researchers is to place a buffer of work in front of every work-station, whatever the objective function is being used. The answer to the third issue is then typically found by applying a combinational optimisation technique. The practical benefits of locating buffers throughout a manufacturing facility and the use of complex combinatorial optimisation methods to solve over-stylised problems are questionable. As a consequence of this "academic" approach, research results are rarely used by practitioners who still rely on intuition to solve the BMP.

The production application of the Theory of Constraints, Drum-Buffer-Rope (DBR), provides exact answers to the first and second questions. Throughput (or output rate) is adopted as the objective function. Buffers locations are limited in front of the constraint work-station, in assembly areas using constraint processed parts and in the shipping area.

Buffer size is a open issue in a DBR implementation and directly influences the time-based competitiveness of a manufacturing facility. Too small a buffer can result in the constraint

work-station being starved and due date promises missed; too large a buffer can result in a longer than necessary lead times.

Buffer sizing advice is vague and non-specific and relies heavily on managerial understanding and experience. This can reduce the effectiveness of DBR implementations and greatly increases the implementation lead time as intuition rarely guarantees the best possible outcome for a given set of circumstances.

In today's competitive and increasing globalised economy, a structured approach that sizes buffers in an effective and implementable manner is likely to yield significant benefits over a traditional DBR implementation. This thesis explores the subject of practical buffer sizing in a DBR environment.

A fuzzy logic approach is proposed and used to size buffers in a simulated DBR environment. The effectiveness of the technique is assessed and contrasted with a simple and commonly used buffer sizing heuristic.

Simulation results demonstrate that fuzzy logic effectively sizes buffers and is likely to provide a satisfactory answer to the third question of the BMP: what is the appropriate buffer size.

Table of Contents

List of Figures

List of Tables

Chapter One

Introduction

1.1 Research Background

Markets for manufactured products are described as becoming more "globalised". Schermerhorn (1996) defines globalisation in terms of the global economy:

> "one based on worldwide interdependence of resource supplies, product markets, and business competition." (pp. 82)

Reasons for increased globalisation include trade agreements such as NAFTA and GATT and the fact that more nations are developing their industrial base. All this has led to competition in both local and export markets being fiercer than ever before (Kobu *et al*, 1991; Hurley *et al*, 1996).

Darwinian free market forces ensure that only manufacturers with strong competitive advantages will prosper in today's competitive market places. Goldratt *et al* (1986) defines competitive advantage in terms of three dimensions: product, price and responsiveness. Each dimension consists of two sub-dimensions (see Table 1.1).

With world-wide affluence, price is arguably becoming secondary to "fitness for purpose" (product presence) and responsiveness (Hurley *et al*, 1996; Juran, 1992). Responsiveness, or time-based competitiveness, has become increasingly important in today's competitive markets (Stalk, 1988). Consumers expect increasing levels of due date performance and with this, shorter and shorter quoted lead times to market. Time-based competitiveness is fast becoming the key competitive dimension for the 1990's (Blackburn, 1991).

Competitive Advantage	Aspect
Product presence	- Quality - Engineering
Price presence	- Margins - Investment per unit
Responsiveness presence	- Due date performance - Quoted lead time

Table 1.1: Competitive Dimensions (Goldratt *et al*, 1986)

The increasing affluence of Pacific rim economies offers a number of export opportunities. New Zealand manufacturers, geographically isolated from these markets, are faced with greater shipping times. The challenge facing New Zealand manufacturers is how to compete effectively along time-based dimensions given the greater distance to these markets.

The production and operation function plays an important role in determining an organisations competitiveness along time-based dimensions. There are two main reasons: firstly, the manufacturing lead time forms a significant component of the time to market that is largely under the control of the manufacturer; secondly, reliable due date performance is critically dependent on a consistent and predictable production rate.

Time-based competitiveness has focused renewed attention on one of the primary concerns and challenges of production management: scheduling. Baker (1974) in his seminal work defines scheduling as:

> "...the allocation of resources over time to perform a collection of tasks." (pp. 1)

A collection of tasks, in a broad manufacturing context, is a series of operations that must be performed before a product reaches the market place. As time-based competition is important, a schedule is judged to be effective if it minimises both the manufacturing lead time and any costs associated with a less than prefect due date performance.

1.2 Manufacturing System Behaviour

Conceptually, a production system is viewed as an open system consisting of a number of inputs, a transformation process, and a series of outputs. There are two basic phenomena that describe the dynamics of a manufacturing facility:

- existence of dependent events and interactions; and
- occurrence of random events and statistical fluctuations.

The transformation process of a manufacturing organisation is a series of ordered events where each event is dependent on the previous. An example is the routing sequence for a product. To manufacture a product a definite sequence of events (processing operations) must occur in a particular order, each event "dependent" on the one preceding it.

An example of an "interaction" is the simultaneous arrival of two products at a work-station. Hence, one product will have to wait for the other to be processed before it can be processed. Interactions such as these disturb the smooth flow of materials through a chain of dependent events.

"Random events" can never be predicted with any certainty, for example, the failure of a work-station. Random events are extremely disruptive to the smooth flow of work and can never be totally eliminated. In addition, all processes are subject to sources of inherent variability, or "statistical fluctuations". Specific examples include variable setup, processing times and quality defects. Statistical fluctuations are more predictable than random events and occur with greater frequency. Statistical fluctuations and random events are collectively known as variability (or production disruptions) and disrupt the smooth flow of work through a manufacturing facility.

1.3 Effective Scheduling

Interactions and the variability found in manufacturing facilities cause the flow of work to become uneven and unpredictable. This is a substantial obstacle that must be overcome if a schedule of work is to be met. A scheduling technique to be effective should achieve: firstly, a planned product flow (to meet market demand); and secondly, an inbuilt immunity against the effects of statistical fluctuations and interactions.

Manufacturers have not been successful in immunising production schedules against variability. Robustness is an important aspect of an effective schedule: especially if due date promises are to be kept and the manufacturing lead times minimised. The effect of variability has traditionally been dealt with by the use of protective inventories and safety lead times to maintain (or "buffer") a constant output rate. In some instances, manufacturers have chosen to use protective capacity (Atwater *et al*, 1994).

Excessive reliance on inventory buffers and safety lead times has a number of drawbacks including quality problems, obsolescence, larger than necessary lead times and increased work in progress (Atwater, 1991). Adding protective capacity to existing work-stations is costly and must be done in incremental quantities. Further, this requires an understanding of the difference between excess and protective capacity - an issue yet to be addressed by research.

Two contemporary scheduling philosophies that have gained favour with manufacturers are Just-In-Time (JIT) and Drum-Buffer-Rope (DBR). Both JIT and DBR fall under the label of synchronous manufacturing, in which work flow is synchronised with market demand by connecting operational decisions to the overall goal of the company (Umble *et al*, 1990).

Arguably the success of both techniques lies with the ability to address the requirements of an effective schedule. JIT and DBR use well established and tested methodologies to plan an orderly product flow. JIT uses kanban cards (see Cheng *et al*, 1993), while DBR, utilises

a detailed master production schedule (see Spencer *et al*, 1995). JIT and DBR, however, have adopted two different approaches when immunising production schedules.

JIT adopts a scatter gun approach to eliminating all sources of variability in a manufacturing facility. In addition, the productive capacities of resources is balanced to achieve a smooth flow of work thus eliminating interactions. DBR, on the other hand, strategically buffers key resources and adopts a constraint focused improvement program.

1.3.1 *Just-In-Time (JIT) Systems*

Just-In-Time (JIT) systems enjoy an enormous following in the West (Billesbach, 1991; Fawcett *et al*, 1993). Numerous research papers have been written on JIT (Hedin *et al*, 1992; Im *et al*, 1994; Joo *et al*, 1993; Moras *et al*, 1992). Manufacturers who have successfully implemented JIT claim to have reduced waste and improved the quality and efficiency of production thus strengthening their competitive position in the market place (Cheng *et al*, 1993). It is widely believed that JIT systems consistently out-perform other manufacturing systems including MRP II and traditional job shop management techniques.

The introduction of JIT has made buffer resources unfashionable (Atwater, 1991). JIT proponents regard buffers as mechanisms for hiding production problems and as a result, a hindrance to production.

The inventory cost, however, is overstated by many "zero inventory" adherents who fail to understand that JIT is a long term commitment to excellence (White, 1993). The philosophy underlying JIT systems is Kaizen, which translated means continuous improvement (Cheser, 1994). Buffers under JIT are only temporary solutions.

Protective inventory is progressively reduced by decreasing the number of kanban cards that limit the total amount of work-in-progress (WIP) in the manufacturing facility (Cheng *et al*, 1993). Protective capacity is also systematically eliminated, producing a balanced plant.

Safety lead times are discarded in favour of synchronising product flow with market demand.

While the progressive elimination of buffer resources considerably reduces the manufacturing lead time, the output rate could become unstable as it is sensitive to any production variability. Rigorous total quality management (TQM) and total preventive maintenance (TPM) programs form an integral part of a JIT implementation. As a consequence, JIT systems require long implementation lead times before strong competitive positions develop (Hurley *et al*, 1996).

The long implementation lead time precludes the usefulness of JIT for the vast number of manufacturers who have declining or stagnant competitive positions in the market place. Despite this criticism, it is widely accepted by practitioners and researchers that the JIT approach not only meets but exceeds world-class manufacturing standards. Certainly, the strong competitive positions of Japanese companies (notably Toyota (Moden, 1983)) offer strong testimony of the validity of JIT. However, if current performance levels are to be maintained, manufacturers need to implement JIT slowly.

1.3.2 *Drum-Buffer-Rope (DBR) Systems*

Drum-Buffer-Rope (DBR), the production application of the Theory of Constraints (TOC), offers a viable alternative for manufacturers. The strategic placement of buffer resources, as opposed to the elimination of buffers, enables manufacturers to compete effectively along time-based competitions. Further, given time the iterative nature of improvements under DBR is likely to approach a JIT system.

DBR is arguably one of the most effective yet practical production planning and control techniques to date (Gardiner *et al*, 1993). Manufacturers that have implemented DBR have reported significant reductions in WIP levels, enhanced due date performance and reduced lead times (Gardiner *et al,* 1993). The philosophy and practice of DBR has been extensively

discussed in the literature (Goldratt, 1990; Hurley *et al*, 1996; Schragenheim *et al*, 1990; Umble *et al*, 1990).

As discussed in section 1.2, the existence of "dependent events and interactions" and occurrence of "random events and statistical fluctuations" are responsible for the behaviour of manufacturing systems. The basic premise of DBR is that the output rate of any manufacturing facility is limited by the work-station with a loading greater than 100% - the capacity constraint (Umble *et al*, 1990).

Work-orders scheduled to arrive at a constraint work-station will, commonly, be delayed thus starving the constraint of work. To achieve a high level of utilisation, a buffer of protective inventory is maintained in front of the constraint. Under the DBR approach, buffers are also located at assembly points that use constraint processed parts and in the shipping area. No other work-stations are buffered since they are not constraints and by definition have spare capacity. By restricting inventory buffers to key points, the total amount of WIP is minimised and production disruptions that potentially endanger the output rate can readily be identified.

DBR promises manufacturers an enhanced ability to compete along time-based dimensions. The strategic placement of inventory buffers at assembly, shipping and the constraint work-station maximises due date performance, minimises manufacturing lead times and also helps to direct continuous improvement activities. DBR can be quickly implemented in established manufacturing facilities (Hurley *et al*, 1996). As DBR allows a manufacturer to compete effectively along time-based dimensions, this research adopts DBR as its base model.

Buffer sizing is an important issue in a DBR implementation. Buffer size directly influences the effectiveness of a schedule and with it the time-based competitiveness of a manufacturing facility. Too small a buffer may result in the constraint being starved of work and due date performance endangered. Too large a buffer can result in longer than

necessary lead times and missed opportunities for continuous improvement. Outcomes that can threaten competitiveness.

Buffer sizing in a DBR environment has received relatively little attention from researchers. Advice is often vague and non-specific and relies heavily on the experience and intuition of practitioners. Other more "concrete", yet equally heuristic approaches are empirical and do not consider the many factors that influence buffer size. For example, Umble *et al* (1990) recommends that buffer size should be at least half the manufacturing lead time (under a MRP style job-shop management techniques).

Inappropriate buffer sizes reduces the effectiveness of DBR diminishing the competitive ability of the manufacturing facility. A more structured and considered approach to buffer sizing is likely to yield significant benefits.

While DBR validates the need for protective inventory in a manufacturing facility, a number of issues need to be considered. Buffer sizing is one example. The framework for this is the Buffer Management Problem (BMP).

1.4 Buffer Management Problem (BMP)

Chow (1987) suggests two reasons why the BMP remains unsolved. First, the BMP is analytically intractable: no algebraic relationship exists between buffer size and output rate (a commonly used objective function). Second, the combinatorial nature of the BMP makes it difficult to obtain an exact solution.

Existing buffer management research has focused on three questions:

1. What objective function to use?
2. Where to locate production buffers?
3. What is the correct buffer size?

In many cases, the answer to (2) and (3) is critically dependent on the chosen objective function. An objective function of maximising the utilisation of every work-station, for example, will lead to extensive use of buffering throughout a manufacturing facility. No one objective function has been universally accepted and many different solutions (and formulations) to the BMP have been developed.

As with many combinatorial problems, the BMP is simple to formulate, but notoriously difficult to solve. Chapter two, a subject review, surveys a number of proposed solutions to the BMP. Many are attempts at building stylised models, that lend themselves well to standard operations research (OR) techniques. Gradient search, experimental design, and dynamic programming solutions have been applied. However, to make the BMP tractable to traditional operations research techniques such as these, a number of stylised and often inappropriate assumptions are made. Consequently, results are of limited value. Results are also inaccessible to practitioners who lack the prerequisite mathematical sophistication required by each technique. Further, the use of OR is arguably inappropriate since the problem characteristics are many, complex and subject to change. An optimal solution is then only of limited use. It is of little surprise that practitioners often use intuition and experience to locate and size buffers (Hurley, 1996). Intuition, however, is inexact and does not guarantee the achievement of the highest performance for a manufacturing facility given a particular set of circumstances.

1.5 Purpose of the Research

A structured and considered approach to buffer sizing is likely to enhance a manufacturers ability to compete along time-based dimensions. However, the non-specific advice of DBR and inherent complexity associated with manufacturing facilities make the utility and appropriateness of an OR-based solution approach to buffer sizing questionable. A new approach that can produce practical and implementable buffer sizing techniques is needed. Requirements of a practical solution are discussed in chapter two.

A fuzzy logic solution to the BMP has many characteristics that make it an ideal platform for sizing production buffers. This research investigates the applicability of fuzzy logic to buffer sizing. The result is a robust and practical solution to buffer sizing, extending the work of Goldratt (1990) and Umble *et al* (1990).

Zimmerman (1985), a noted fuzzy theorist, defines fuzzy logic as:

> "an extension of theoretic multivalued logic in which the truth values are linguistic variables (or terms of the linguistic variable truth)." (pp. 132)

In order words, fuzzy logic is an extension of set theory that underlies the development of operations research models. This allows vagueness and ambiguity - often a stumbling block for traditional OR approaches based on Boolean logic - to be captured in a structured approach. Fuzzy logic is a human centred approach to problem solving, that through the use of fuzzy rules, provides a mechanism for capturing expert knowledge, or intuition.

1.6 Methodology

A fuzzy model is developed in MATLAB v.4.2 (MathWorks, 1992), a computer program for mathematical computing. The vague and general advice of DBR is used to derive the fuzzy sets and fuzzy rules. This thesis proposes a simple model based on a small rule base and unoptimised membership functions.

A simulated DBR logistic system consisting of six work-stations is implemented in SIMSCRIPT II.5 (CACI, 1996) and used to gauge the effectiveness of the fuzzy logic model. A simulation approach is adopted for two reasons. Firstly, simulation is capable of dealing with the complex and the stochastic nature of manufacturing facilities. Secondly, simulation allows the appropriate buffer size to be evaluated. This can is then used to gauge effectiveness of the fuzzy buffer size.

Two dependent variables (mean protective capacity and coefficient of variation of processing times) were used to characterise a variety of manufacturing facilities or environments. To estimate the appropriate buffer size for the manufacturing facility the delay time of each work-order is collected. The delay time is defined as the difference between the actual arrival time and the expected arrival time at the buffer. By equating the delay time distribution with the buffer size, the appropriate buffer size can be calculated.

For the purposes of this research, a simple measure of buffer effectiveness is proposed:

$$| \text{Estimated Buffer Size} - \text{Appropriate Buffer Size} | \qquad \textbf{(1.1)}$$

A two sample t-test is used to compare the effectiveness of the fuzzy model with Umble's *et al* (1990) heuristic over a variety of manufacturing environments. In addition, statistics are collected on mean constraint utilisation and cycle time.

1.7 Thesis Structure

Chapter two examines the BMP. Previous OR-based solution methods are critically examined and issues associated with each method are highlighted. DBR is examined in detail and contrasted with OR-based solution methods. Chapter two concludes with a research agenda for effective buffer management.

Chapter three reviews the fundamentals of fuzzy logic. Fuzzy set theory is contrasted with classical set theory and the fuzzy modelling process is examined. A fuzzy approach to the BMP is examined.

The fourth, fifth and sixth chapters detail the design of the fuzzy model and its implementation in a simulated DBR environment. In particular, chapter five examines the methodological issues associated with simulation modelling of the BMP in a DBR environment. Chapter seven presents the experimental design and analysis of the results.

The fuzzy model's performance is assessed and a comparison is made with the Umble's *et al* (1990) heuristic. Chapter eight details opportunities for future research. Chapter nine concludes.

Chapter Two

Buffer Management Subject Review

2.1 Introduction

This chapter reviews buffer management research and argues that the majority of this research has lost sight of its primary purpose: an implementable and usable solution to the buffer management problem. Previous solutions are critically examined. The wisdom of locating buffers throughout a manufacturing facility, and the use of complex combinatorial optimisation methods to solve overly-stylised problems, is questionable. Lastly, a research agenda based on the Theory of Constraints (TOC) is presented which has the potential to yield relevant and implementable results.

They are many sources of uncertainty which cause disruption and inefficiency in a manufacturing facility of which work-station failure, scrap, varying product mixes and processing times are just a few examples. Today's production manager is under increasing pressure to shorten lead times, reduce costs, and continually improve quality, producing higher profitability, greater return on investment and ultimately ensuring corporate survival. The variability and complexity inherent in many manufacturing facilities present a substantial obstacle to the achievement of these goals. Given the numerous product-work-station interactions existing in a plant, a real understanding of this variability often remains illusive to the production manager. Rather than invest the considerable time required to analyse and resolve this uncertainty, production managers tend to maintain large buffer stocks of raw materials, work in progress (WIP) and finished goods (Umble *et al*, 1990). The negative effects of excessive inventory are well-documented: poor quality, increased holding costs, damage to work in progress, and unpredictable and excessive lead times (Goldratt *et* al, 1986; Umble *et* al, 1990). Excess inventory has been a clear contributor to the decline in the competitiveness of western manufacturing concerns.

This chapter presents a survey of the literature and outlines a research agenda for investigating practical heuristics for effective buffer management.

2.2 Production Buffers

A queue of work in front of a work-station is sometimes called a buffer; its purpose is to protect a work-station's output from the variability in feeding work-stations upstream.

The role of buffers in manufacturing facilities has been investigated for over 35 years. Koenigberg (1959) identified the location and allocation of buffer capacity as important design parameters for manufacturing facilities and hence defined the buffer management problem (BMP). There is no consensus of research opinion on either the unit of buffer size; or on the role that buffers play in manufacturing facilities.

Buffer size is commonly defined in terms of a physical quantity of WIP (Tompkins, 1990). This definition is limited as it does not quantify how much protection the buffer offers the work-station. An alternative definition of buffer size is in terms of time (Hurley, 1993; Schragenheim *et al*, 1990; Umble *et al*, 1990). Buffer size then represents how long it takes a work-station to process the work in the buffer (assuming no more work arrives at the buffer).

Defining the role buffers play in manufacturing facilities is an important starting point for any research into buffer management. Some researchers, such as Park (1993), believe that buffers *enhance* a manufacturing facility's output rate. Others (Goldratt *et al*, 1986; Schragenheim *et al*, 1990) maintain that the production constraint is the only determinant of the output rate. The role of buffers is then *protection* of the output rate.

There is general agreement that buffers are needed because of variability in manufacturing facilities. Variability arises from one of three sources (Chow, 1987):

1. variable processing (set-up and operating) times;
2. work-station failure; or

3. varying product yields, not for chemical processes, but for any process that produces quality defects[1].

Varying product yields and work-station failure, however, are becoming less of a problem with the adoption of statistical process control and total preventive maintenance (Baker, 1992). As a result there is greater need to understand the effects of variable operating and set-up times on the effectiveness of a buffer management policy.

2.3 The Buffer Management Problem

Allocation of buffer capacity is equivalent to buffer location, that is, if no buffer capacity is allocated at a site, then that site is not buffered. Nevertheless, the location and allocation of buffer capacity remain separate questions because researchers do not agree on where buffers should be located. Chow (1987) suggests that in the presence of variability, a buffer should be employed. This inevitably leads to the "buffers everywhere" approach that places buffers between each work-station in a manufacturing facility. This has been adopted as a base model by many researchers. In contrast, Goldratt *et al* (1986) maintains that only constraint work-stations, assembly points using constraint processed parts, and shipping areas should be buffered. Other non-constraint work-stations can "go idle" for certain amounts of time. They do not affect the throughput rate of the facility, as the throughput rate is dictated only by the processing rate of the constraint, the most heaviest loaded work-station.

There are many possible solutions to the BMP (Chow, 1987) and accordingly, to separate profitable from non-profitable answers, a measure of "goodness" - an objective function - is needed.

In short, BMP research must address the following areas:

1. identification of the objective function(s);

---

[1] A common research assumption is to ignore quality defects, although Bulgak (1992) explicitly addresses this. Jafari *et al* (1989) has also considered the possibility of quality defects arising from machine breakdowns, and their effect on the predictability of product flow.

2. location of the buffer; and
3. allocation of buffer capacity (or size).

Clearly the goal of the second and third point is the optimisation of the identified objective function.

2.3.1 *The Objective Function*

The criterion for measuring the "goodness" of each solution, the objective function, varies from one research paper to the next. Maximisation of output rate, or throughput rate, is a commonly-used objective function, although other measures such as line efficiency and buffer utilisation cost are also used. Table 2.1 groups research papers according to the objective function employed.

Objective Function	**Author(s)**
Maximise output rate (or throughput rate)	Chow (1987), Goldratt *et al* (1986), Hillier *et al* (1993), Hillier *et al* (1991), Ho *et al* (1979), Iyama *et al* (1989), Jafari *et al* (1989), Park (1993)*, and Smith *et al* (1988)*.
Maximise line efficiency	Ohmi (1992)
Minimise buffer cost	Anderson et al (1969) and Smith *et al* (1988)*.
Minimise average demand backlogged	So et al (1988)
Maximise system availability	Baral (1993)
Minimise total buffer capacity	Park (1993)

* Dual objective function.

Table 2.1: Objective Functions Used in the BMP

Traditional objective functions, such as utilisation or buffer cost are myopic and fail to translate operational decisions faced by the production manager to the business bottom line, and in many cases can mislead decision-makers. These objective functions encourage increased WIP inventory, larger buffers, and longer lead times (Fry, 1990). For-profit businesses exist to make money both now and in the future (Goldratt *et al*,

1986). To be judged by any other measure is not good science, unless the lesson taught by Ockman's razor (simplest explanation is best) (Microsoft Encarta, 1994) is to be ignored. Throughput is a simple, obvious and objective measure of choice for successful buffer management.

For any "real world" application, throughput takes on an expanded definition. Not only do products need to be produced, but they also need to be marketed and sold. Although the ability to generate revenue is a business's most important criterion, in all research papers surveyed the demand for any finished product was assumed to be unlimited. In this case, output is equivalent to throughput. This assumption is rarely met and while output rate and throughput are used interchangeably by researchers, the production manager needs to bear this distinction in mind.

2.3.2 *Formulation of the Buffer Management Problem*

Recognising the importance of throughput, Park (1993) presents four standard formulations of the BMP for m-machine serial manufacturing facilities. The objective functions are the maximisation of throughput (related to profit), and the minimisation of buffer capacity (related to cost or the loss of responsiveness to customer needs). The decision variables represent the amount of buffer capacity to be allocated to each buffer (including zero capacity for no buffer).

Formulation 1

$$Min \sum_i b_i$$

$$S.T.\ f(\mathbf{b}) \geq K$$

where b_i is the capacity of the i^{th} buffer feeding the $(i+1)^{th}$ machine,

$\mathbf{b}$ is a vector with elements $\{b_1, b_2, \ldots, b_{m-1}\}$,

$f(\mathbf{b})$ is the throughput of the line given $\mathbf{b}$, and

K is the required throughput.

Defining throughput as a function of buffer size and location is questionable, as throughput is determined solely by the constraint. Furthermore, if b_i is defined in terms of quantity, rather than time, $f(\mathbf{b})$ becomes even more unworkable because throughput is a function of time. Nevertheless, for the simple two work-station exponential case, Hunt (1956) derived an expression that related the buffer size to the maximum utilisation of the system (for a generic expression see Anderson *et al*, 1969).

Formulation 2

$$Max\ f(\mathrm{b})$$

$$S.T. \sum_i a_{ij} b_i \leq B \quad \forall j$$

where a_{ij} is the "cost" of a unit of i^{th} buffer capacity for the j^{th} constraint, and B_j is the total "cost" associated with each constraint (for example, B_j may be the total buffer capacity available to constraint j).

Unlike Formulation 1, which placed a minimum restriction but no maximum restriction on the total buffer size, Formulation 2 restricts to the buffer size to quantity B_j. B_j may reflect physical realities, such as limited floor space. Indeed, when WIP inventory becomes uncontrollable, the manufacturing facility line becomes more difficult to manage.

Formulation 3

$$Max \sum_i b_i$$

$$S.T.\ f(\mathrm{b}) \geq K$$
$$\sum_i a_{ij} b_i \leq B_j \quad \forall j$$

Since buffer size has a direct effect on lead time, it is often in the best interest of the company to minimise the total buffer capacity allocated (Umble *et al*, 1990). However, consideration should also be given to throughput rate and any work-station limitations. Given these two competing constraints, it is possible that no optimal solution exists for Formulation 3. As Park (1993) notes, management may need to revise their formulation, possibly opting for a lower throughput rate. With the relaxation of the work-station constraint, Formulation 3 becomes equivalent to Formulation 1.

Formulation 4

$$Max\, f(\mathbf{b})$$

$$Min \sum_i b_i$$

Unlike the previous formulations, here there are no stipulations, except that throughput is maximised while buffer size is minimised. This approach differs from those of most researchers, who generally use only one objective function, throughput. Baral (1993) agrees with this formulation, contending that the effectiveness of buffer capacity diminishes as more capacity is utilised. Formulation 4, therefore, extends Formulation 2, which often results in an optimal solution that has more buffer capacity than is strictly necessary (Park, 1993).

2.3.3 *Generic Design Issues*

Most research efforts concentrate on modelling serial manufacturing facilities. Common assumptions used in these models include:

- the manufacturing facility has entered steady state conditions;
- work-stations never fail nor produce defective parts;
- the first work-station never starves for work;
- demand for the finished product is unlimited; and
- processing times are independent random variables.

Other assumptions relate to the application of the specific solution technique. The dynamic programming formulation of Chow (1987), for example, assumes the BMP obeys the optimality principle (Winston, 1987).

A model's assumptions can limit the practical application of the results. The exponentially-distributed processing times used by Smith *et al* (1988), for example, are becoming increasingly uncommon with the wide-spread use of SPC. Hurley (1996) has been particularly critical of academic research into production management problems, asserting that the focus on elaborate mathematical techniques to analyse generic models, based on impractical assumptions, fails to solve real problems. Justification for these limited models is their simplicity (Baker, 1992). They are intended to provide the understanding necessary to formulate appropriate heuristics. A review of the literature, however, reveals that such heuristics are few. Moreover, there is a definite lack of guidelines for production managers wanting to implement research results. The "science" of designing manufacturing facilities appears to be still in it's infancy (Baker, 1992), in spite of the fact that buffers have been investigated since Hunt (1956) presented his analytical formulation over 30 years ago. The BMP remains unsolved.

2.4 Solution Approaches

In terms of simplicity, the most elegant solution to the BMP is to allocate infinite buffer capacity at every possible location. Physically, this solution is represented by a never-ending queue of work in front of every work-station. Many authors, such as Anderson *et al* (1969) and Baral (1993), discuss infinite buffers. Storage requirements, handling costs, and the diffusion of quality problems are just a few reasons that make this solution both undesirable and unattainable. Perhaps the most significant cost of an infinite buffer, is the inability to respond to changing customer requirements, simply because of the resulting infinite lead time! Short lead times are an important element of an organisation's competitive stance. Today's market place is dominated by time-based competition, hence buffers should be as short as possible (Blackburn, 1991).

Park (1993) believes that as the size of the buffer increases so does the performance of the manufacturing facility. Baral (1993) has shown mathematically this is true for system availability, however, increasing buffer capacity results in diminishing returns. Indeed, small buffers are often adequate to deal with much of the uncertainty in manufacturing facilities (Hendricks, 1992).

The remainder of this chapter surveys the numerous approaches applied to solve the BMP. The suitability of each approach to the BMP is assessed and comment is passed on the research conducted to date. Such approaches include: analytical, enumerative, simulation, experimental design, search methods, and heuristics. Finally the Theory of Constraints (TOC) philosophy is presented, as a solution to the BMP and an agenda for future research.

2.4.1 *Analytical Solutions*

If an algebraic relationship existed between throughput and buffer size, then given the desired throughput, the required buffer size could be calculated and the BMP would be trivial. Researchers have attempted to derive such a mathematical relationship. Hunt (1956) developed a simple result for a two-stage manufacturing facility, with work-stations having identical and exponentially-distributed processing times. Throughput and buffer size were related using the following formula:

$$T = \frac{B+2}{B+3} \qquad \textbf{(2.1)}$$

where B is the buffer size between the two stages and T the line's throughput. Throughput is maximised when the buffer size tends to infinity, obviously an impractical result. Hunt's model is too simplistic for any practical use.

More recent work by DeKok (1990) and Blumenfeld (1990), cited by Baker (1992), approximates the throughput of a balanced line, with arbitrary processing distributions, and set buffer sizes. Blumenfeld's approximation requires an estimate of the unbuffered

line's throughput, which in practice may be difficult to determine. Practical application of both approaches is limited because unbalanced serial lines are far more common in manufacturing facilities. In the more realistic case of an unbalanced line, throughput is determined by the constraint, thus relating buffer size to throughput would appear unwise. One must therefore conclude that a practically applicable algebraic relationship between buffer size (measured by product quantity) and throughput cannot be derived (Chow, 1987).

Hunt (1956) derived his exact expression by assuming that processing times were exponentially distributed. This simplifying assumption is inappropriate. Processing times and work-station repair times are described by general distributions because of the wide-spread adoption of statistical process control and total preventive maintenance (Koulamas, 1993). For more complex (and arguably realistic) manufacturing facilities consisting of more than three stages, subject to breakdowns and general processing distributions, the mathematics becomes extremely complex.

Analytical models therefore appear to have limited practical use because of their restrictive assumptions. Analytical models, however, can provide an informative starting point for researchers wishing to understand the important issues of the BMP.

2.4.2 *Enumerative Solutions*

One Factor at a Time

The one-factor-at-a-time approach alters the buffer size iteratively by one "unit", until buffer capacity is optimally distributed. A variety of objective functions have been used, with throughput being the most popular. While intuitively appealing, this approach is flawed because the BMP does not have a unique solution because buffer sizes are dependent (Park, 1993). Moreover, being time intensive and cumbersome, this approach is limited to balanced manufacturing facilities with small numbers of buffers.

Dynamic Programming

Dynamic programming is an efficient enumerative method that divides a large problem into a series of smaller problems, which are then individually solved in series. Researchers have used have used dynamic programming, maximising throughput by allocating a given total buffer size. In the absence of an algebraic relationship between throughput and buffer size, Jafari *et al* (1989) used a heuristic, while Chow (1987) recursively used a regression equation to evaluate throughput.

The optimality principle is an important assumption in dynamic programming (Winston, 1987). In the case of the BMP, this implies that an optimal decision for one buffer location does not depend on optimal decisions for other locations. The interaction effects between buffers make this assumption unrealistic (blocking, for example, propagates upstream, effecting later work-stations) (Park, 1993). These limitations mean that the solutions generated are, at best, approximate. Kubat *et al* (1985) found the optimal number of infinite buffers for a manufacturing facility, but this was only possible because infinite buffers lend themselves to closed-form solutions and obey the optimality principle (i.e., problem is decomposable) (Jafari *et al*, 1989). Infinite buffers are neither practical nor desirable in realistic settings. Therefore though limited in application, nor as complex as analytical models, dynamic programming is nevertheless limited by its assumptions.

Another criticism of dynamic programming is its computational efficiency. All enumerative methods, regardless of astuteness, suffer because of the combinatorial nature of the BMP, or "...a malady melodramatically labelled the curse of dimensionality" (Goldberg, 1989, pp. 12).

2.4.3 *Simulation Solutions*

For the researcher, simulation is a powerful tool, providing an efficient way to understand manufacturing facilities of moderate to high complexity. Uses of simulation include ascertaining the significance of factors affecting the BMP and providing

empirical support for qualitative arguments. Simulation is an appropriate approach for the BMP as it is not limited by stylised and restrictive models.

Simulation is often used in conjunction with other solution approaches. Anderson *et al* (1969) successfully applied regression analysis and simulation to solve the BMP. Using regression analysis, an empirical relationship was found between production cost and buffer size (and length of manufacturing facility). This extended the work of Hunt (1956) by deriving a general expression for the maximum utilisation of the manufacturing facility. The assumption that the cost of buffers was adequately represented by a delay cost and an inventory cost, which in turn enables the optimal buffer capacity to be determined by differentiation.

After a series of simulation experiments on identical stations, Conway *et al* (1988) found a repetitive pattern in the optimal allocation of buffer capacity. Called the "bowl phenomenon", it is analogous to the optimal allocation of work (Baker *et al*, 1993). A plot of the optimal buffer size against the buffer's location results in a bowl-shaped graph; this pattern is described as "centred weighted" with slightly more buffer capacity allocated to the centre work-station. While the bowl phenomenon is not intuitive, Conway *et al* (1988) propose that the emphasis on the centre work-station can be explained by considering the effect of starving and blocking. As the first work-station is never starved of work, the effect of starving becomes more severe further down the line. Likewise, because the last work-station is never blocked - due to an infinite product demand - the effect of blocking becomes more prevalent up the line. The effect of starving and blocking thus implies that buffer capacity is best served in the middle of the line.

Given a known amount of buffer capacity, Conway *et al* (1988) proposes the following practical heuristic:

1. Allocate buffer capacity as equally as possible.
2. If buffer capacity remains after equal allocation, allocate the rest of the buffer capacity in approximately equal intervals with the first and last buffer location getting the lowest priority.

According to Baker (1992) exceptions have been found to the equal allocation rule by Hillier *et al* (1991) and so "the concept of distributing buffer slots evenly, as a general guide-line, is open to debate" (pp. 394).

Investigating the bowl phenomenon, Hillier *et al* (1993), like Conway *et al* (1988), found that centre buffers should be given preferential treatment. This research has extended Conway *et al*'s (1988) second step to specific values, and notes that the total buffer size is an important decision variable, which is not addressed by Conway's heuristic.

Conway *et al* (1988) also investigated balanced lines of unequal variability and unbalanced lines, where throughput is limited by a bottleneck, the latter more realistic. Their conclusion was that buffers are more important in balanced lines than for unbalanced lines, where buffer capacity should be allocated to utilise the bottleneck's capacity. Goldratt (1990) developed this argument further, asserting that in an unbalanced line non-constraint work-stations offer "protective capacity". This is the ability of "faster" upstream stations to serve as buffers to counteract any adverse effects of variability.

2.4.4 *Design of Experiments (DOE) Solutions*

As an approach to experimentation, DOE is both economic and informative. The usefulness of DOE, however, depends on the researcher's experience and understanding of the BMP for a given manufacturing facility. Factor levels (associated with buffer size for example) need to be set and changed until a possible optimal solution is reached. A characteristic of the BMP is a number of local optima and, like search methods, DOE does not guarantee a global optimal solution. However, as Park (1993) notes, DOE does provide an understanding of the solution space and the location of possible optimal solutions.

2.4.5 *Search Method Solutions*

Gradient Techniques

Gradient techniques use the direction of greatest increase (or decrease) to find an improved, if not optimal, solution. Ho *et al* (1979) presented a solution to the BMP using gradient optimisation which they claim is both efficient and robust. But although moving in the direction of greatest improvement is intuitive, Bunday (1984) has listed a number of limitations of this method:

- independent variables such as buffer size need to be continuous. While the buffer size can be approximated by a continuous variable, the result is only an approximation (Park (1993) also notes that discreteness of the buffer size will stall the method.);
- frequent direction changes may reduce the efficiency of the method; and
- the gradient is a local measure and as a result, may find a local optimum.

Direct Search Methods

Direct search methods utilise only the objective function's value and do not need the gradient. One such direct search method is the Hooke and Jeeves, which Bunday (1984) describes as a "very efficient and ingenious procedure" (pp. 32). Like many of the traditional optimisation methods, this method is dependent on the initial starting conditions of the "search", as throughput over buffer size is not unimodal (Park, 1993). The Hooke and Jeeves method guarantees a local optimum and not necessarily a global optimum. Other direct search methods, such as the Complex Method (see Bunday, 1984), use random numbers to introduce an element of chance into their search to avoid local optima. By their very nature, however, methods that use random numbers are inefficient.

Smith *et al* (1988) modelled automated assembly lines as finite open queuing networks. The optimal size of each buffer between work-stations was determined using the dual objective function: maximise throughput while minimising buffer costs (Formulation 4). An approximate analytical decomposition technique was used to calculate throughput, while an unconstrained optimisation technique - based on Powell's method (see Bunday, 1984) - was used to find optimal buffer sizes. The authors believe the procedure is

effective for balanced assembly line configurations where work-stations are not over-utilised. Although the methodology may not produce an optimal solution, it is useful for investigating alternative configurations of buffer capacity. The authors intend to present a case study application of their methodology in a later paper.

Genetic Algorithms (GA) are general optimisation methods that combine the robustness of random techniques with the efficiency of direct search techniques (Goldberg, 1989). A review of the literature indicates that GAs have not been used to solve the BMP. GAs do not suffer from the discrete nature of buffer capacity, and will eventually find the global optimum. As with other direct search methods, GAs are ideally implemented with a simulation approach, since only information about the line's throughput is required. For a thorough discussion of GAs see Goldberg (1989).

2.4.6 *Heuristic Solutions*

Without training in advanced mathematical or statistical techniques, the average production manager will look to heuristics or "rules of thumb" for solutions. The limitations of traditional optimisation methods, necessitates the use of heuristics to develop solutions for the BMP (Park, 1993). Such heuristic methods, however, are scarce and only Goldratt (1990) appears to have developed a heuristically-based approach that could be easily transferred to the manufacturing setting.

Park (1993) uses a two-phase heuristic to determine buffer sizes. The first phase generates a near-optimal feasible solution, given the property of quasi-concavity. This states simply that throughput (or output rate) is a monotonically non-decreasing function of buffer size. The next phase employs a beam search method which improves the quality of the solution. For a discussion of beam search methods see Morton *et al* (1993).

2.5 Criticism of the Optimal Buffer Concept

A high degree of mathematical sophistication is needed to use many of the optimisation techniques discussed. Accordingly, operations research has been criticised for replacing common sense with sophisticated mathematical and statistical decision models (Goldratt, 1990; Huysmans, 1994). Reimer (1991) illustrates how simple solutions, rather than complicated answers, are more likely to succeed in an industrial environment. Simple, common-sense solutions allow staff to assume ownership and adapt solutions as circumstances warrant (Schermerhorn, 1996). The average production manager, struggling to understand the complexity of a manufacturing facility, can lose the meaning and the relevance of overly-sophisticated optimisation techniques. The potential impact of these techniques is thus diminished, if not lost, as the production manager continues to solve the BMP intuitively.

The word "optimality" suggests completeness and finality: one can do no better. In a competitive environment "optimal" buffer sizes are in direct conflict with "continuous improvement", an organisational value that is fast becoming the norm. If a buffer offers optimal protection against uncertainty, there is little incentive to uncover and eliminate sources of variability. In fact, production problems often facilitate improvements by forcing managers to closely examine their systems. While the argument against an optimal buffer is not a technical one, it does still acknowledges an aspect of human nature (Freud (1920) termed this "the pleasure/pain principle": human nature seeks pleasure and avoids pain).

Goldberg (1989) reflects "It would be nice to be perfect: meanwhile, we can only strive to improve." (pp. 7). Optimality may not necessarily be desirable. In fact, given the current dynamic and competitive business environment, a solution to the generic BMP should perhaps be judged according to satisficing (Simon, 1969). The concept of satisficing or continuous improvement (arguably overlooked in the calculus-based optimisation methods (Goldberg, 1989)) has become one of the central ideas of total quality management and may represent a more practical solution approach.

Satisficing, distinct from optimisation, is helpful for characterising heuristics usable by the typical production manager. A technique is required to manage the buffer resources effectively, using the objective function of throughput, and to encourage continuous improvement. This is what implementations of the Drum-Buffer-Rope (Goldratt *et al*, 1986) application of the Theory of Constraints claims to have been successful in achieving (Reimer, 1991).

2.6 Drum-Buffer-Rope

Drum-Buffer-Rope (DBR) is a "common-sense" set of tools to be used by the production manager responsible for manufacturing facility control (Goldratt, 1990). DBR, an application of the Theory of Constraints, hinges on the identification of the primary constraint limiting a manufacturing facility's throughput rate. Before discussing DBR and its satisficing solution to the BMP, it is pertinent to review the characteristics of manufacturing systems previously discussed in section 1.2.

2.6.1 *System Characteristics and Dynamics*

Umble *et al* (1990) has characterised the total system behaviour of every manufacturing facility by the following phenomena:

Dependent Events and Interactions

To achieve any desired objective, a series of events must occur in a particular order. An example of a chain of dependent events is the routing sequences for jobs in a manufacturing facility. To manufacture a product, a sequence of events (processing operations) must occur in a particular order, each event "dependent" on the one preceding it.

"Interactions" occur within dependent event chains. An example of an interaction is the simultaneous arrival of two products at a work-station for processing. One product will have to wait for processing. Interactions such as this disturb the smooth flow of

materials through a chain of events. Devising ways to deal with these interactions, that is, via scheduling, is one of the primary challenges of production management.

Random Events and Statistical fluctuations

Interactions cause difficulties in scheduling a manufacturing facility. This is further complicated as each manufacturing stage is subject to "random events". These occurrences, such as a broken tool or a power failure, happen on a "random" basis and are very disruptive to the smooth flow of work. Random events can not be predicted with certainty, and their sources can never be totally eliminated.

Variability inherent in all processes are termed "statistical fluctuations", for example, variable processing times or yield rates. Such variation is more predictable than random events but does create uncertainty as to the outcome of an event.

Goldratt believes these phenomena describe the total system behaviour of any manufacturing facility, and are responsible for the "fires" a manager must "put out" in a normal working day (Umble *et al*, 1990).

2.6.2 *Drum-Buffer-Rope (DBR)*

The DBR solution proposes that a constraint work-station dictates that potential throughput of every dependent event chain (Umble *et al*, 1990).

Although the following example is quite simple it illustrates the importance of the constraint in maximising a chain's throughput (Hurley *et al*, 1996). Figure 2.1 depicts a manufacturing facility that produces one product, using, in sequence, work-stations A - E.

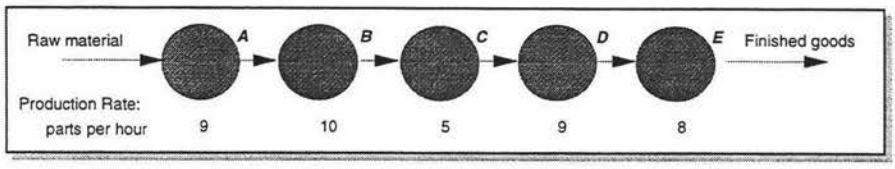


Figure 2.1: Example Manufacturing Facility

Production rates are in parts per hour. Assumptions are: a constant supply of raw material, a worker at each station, unlimited demand (output is equivalent to throughput), and a 24-hours-per-day operation. For simplicity, it is assumed the set-up time is zero for each work-station.

The traditional approach to operating this line would be to maximise the throughput of each link (manufacturing event) in the chain. Historically, production managers have chosen to maximise the utilisation of each work-station, believing that this would maximise the throughput of the whole chain (Crandall *et al*, 1993; Umble *et al*, 1990). Buffers of work are allocated in front of every work-station leading to the "buffers everywhere" policy.

In the example outlined above, work-station A should produce nine parts in one hour. Work-station B, although having the potential to produce 10 parts would actually process nine parts, being limited by the output rate of work-station A. Work-station C would process only five of the parts waiting in its queue. Regardless of their capacity, work-stations D and E would only process five parts per hour, being limited by what work-station C can process and pass to them. Running all the work-stations continuously would result in a throughput of five parts per hour, and a build up of four parts per hour of WIP in front of work-station C. The output of the manufacturing facility is clearly limited by the slowest work-station, C. It is illogical for any production manager to have work-station B operating on average more than 30 minutes every hour. Thus, maximising the output of every link in a chain of dependent events does not maximise the output of the whole chain. Throughput is limited by the work-station that has the least potential throughput, called the *capacity constraint* (Umble *et al*, 1990). In

every organisation's complex network of dependent event chains, there is at least one constraint that limits throughput (Goldratt, 1990).

A total system approach to manufacturing is required, focusing on maximising the throughput of the chain of events, not the output of each link in the chain. The maximum throughput of the line is clearly limited by the capability of the constraint (work-station C). The other work-stations need only work at the same rate as the constraint work-station. A utilisation of 50% at work-station B (a non-constraint work-station) is acceptable provided it supplies, in a timely manner, all materials required by the constraint.

As the throughput of the entire manufacturing facility is dictated by the constraint work-station, any downtime at work-station C results in lost throughput. By definition, non-constraint work-stations have spare capacity. If downtime on one of these work-stations is recovered from before the constraint is starved of work then no throughput will be lost.

According to Goldratt, a facility's potential throughput can only be realised by maximising or exploiting all the capacity of the constraint work-station. To realise this potential throughput, material release and activation of non-constraint work-stations must both be subordinated to the constraint's schedule, that is, the constraint's schedule must dictate both activities. An important issue that then arises from this subordination is the use of production buffers to protect the constraint from the effect of variability in the feeding work-stations (Hurley *et al*, 1996).

Material Release

Under the Theory of Constraints approach to production control, subordinating material release to the bottleneck's schedule means releasing material at the constraint's processing rate. Faster release of material would only increase inventory levels, with no corresponding increase in throughput.

The timing of material release is also understood to be extremely important (Fry, 1990). In a hypothetical manufacturing facility, for example, if the actual processing time from material release to the constraint is two hours, statistical fluctuations and random events will cause a part to take more than two hours to be processed through to the constraint. To ensure timely arrival at the constraint, material release must therefore occur more than two hours before the constraint is to process the part. If the buffer time is sufficient (for example, six hours), the part should arrive at the constraint sometime after two hours has elapsed, but before eight hours. In this case, the buffer is large enough to protect the constraint from the effects of variability in the upstream processes, the constraint is never starved, and throughput is protected.

Activation of Non-Constraint Work-Stations

With subordination of non-constraint work-stations, upstream work-stations will only process what the constraint needs to consume. The material release policy determines when work will be available for processing at non-constraint work-stations. Utilisation of non-constraint work-stations therefore depends solely on the constraint's processing rate, and is independent of an individual work-station's own capacity - a premise that is contrary to traditional methods of operating (Umble *et al*, 1990).

To summarise: in accordance with the "road runner" principle, non-constraint work-stations work at one of two possible "speeds"; not at all, or as quickly as possible[2] (Goldratt, 1996).

Variability and Production Buffers

A common measure of buffer size is the physical quantity of work the buffer contains (Tompkins, 1990). With the Drum-Buffer-Rope approach, the measure used is the *amount of processing time* required to clear the buffer. Under DBR, there are only three buffer locations:

[2] With due regard to quality.

constraint work-station buffer - located in front of the constraint work-station. The purpose of this type of buffer is to protect the constraint from the effect of variability inherent in the feeding work-stations;

assembly buffer - located where a constraint-processed part is assembled with a non-constraint-processed part. The non-constraint-processed part must arrive at the assembly point before it is actually required, thereby ensuring that there will be no interference with the future processing of a constraint-processed part; and

shipping buffer - located in the finished goods dispatch area. Finished products should arrive at the shipping area before the delivery due date. This buffer protects the shipping schedule from the effect of variability in the processing steps between the constraint and the finished goods store.

Larger buffers offer more protection but result in a longer production lead time, although there has been no research to date characterising this trade-off. Logically, the "best" size of a buffer is a trade-off between protection from the effect of variability and a competitive lead time. Hurley (1996), however has shown that WIP levels and production lead times will be far lower under DBR than under traditional methods, and that throughput protection will also be far greater under DBR. The buffers also provide greater throughput protection than under Kanban systems (Umble *et al*, 1990).

Buffer size is a trade-off between the amount of variability and protective capacity of non-constraint work-stations, and due date protection and having a competitive lead time (although, again, research is needed to mathematically model this relationship). Disruptions upstream from the constraint will erode the buffer. In order to recover from these disruptions, non-constraints require extra capacity - termed "protective capacity" - which ensures that the non-constraints can work faster than the constraint and rebuild the buffer. Lower levels of protective capacity will require larger buffer sizes. Conversely, higher levels of protective capacity will allow smaller buffers. Umble *et al* (1990) recommends a buffer of half the production lead time to the constraint (under

traditional manufacturing practices). Scope exists for heuristically-based methods to provide "quick and dirty" estimates of initial buffer sizes for all three buffer types.

2.6.3 *Buffer Management and Continuous Improvement*

Advocates of JIT, such as Crawford *et al* (1991) and Schonberger (1986), regard buffers as a mechanism for "hiding" the effects of production problems. Contrary to this, buffers under the DBR approach are central to ensuring both the smooth flow of throughput and the success of continuous improvement activities.

There are potentially many small improvements that could be made within a manufacturing facility, yet it is infeasible to perform each improvement. A method is therefore needed to determine which improvements yield the greatest impact on the goal of increased throughput and consequent profit. In the DBR methodology, the buffers are central to the identification of production problems.

A buffer is essentially a series of jobs waiting to be processed. Hurley (1996) uses the representation in Figure 2.2 to illustrate the buffer profile. The white boxes, in Figure 2.2, are jobs already in the buffer as scheduled. The length of each box is proportional to a product's processing time. Statistical fluctuations and random events can cause short, temporary, queues to form at non-constraint work-stations, in turn causing jobs to arrive at the buffer later than scheduled. These late or missing jobs, called "holes" (Umble *et al*, 1990), are represented by the grey boxes in Figure 2.2.

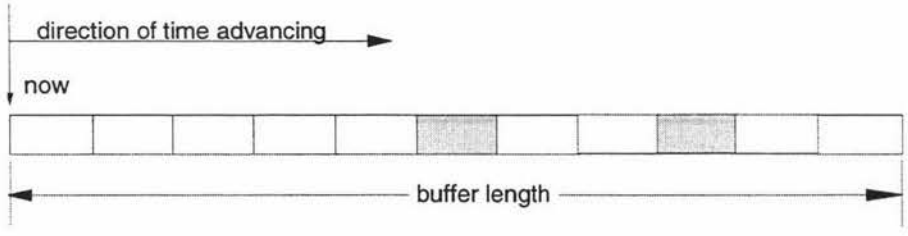


Figure 2.2: Buffer Profile

To enable on-going improvement efforts, Schragenheim *et al* (1990) split the buffer into three regions, as depicted in Figure 2.3.

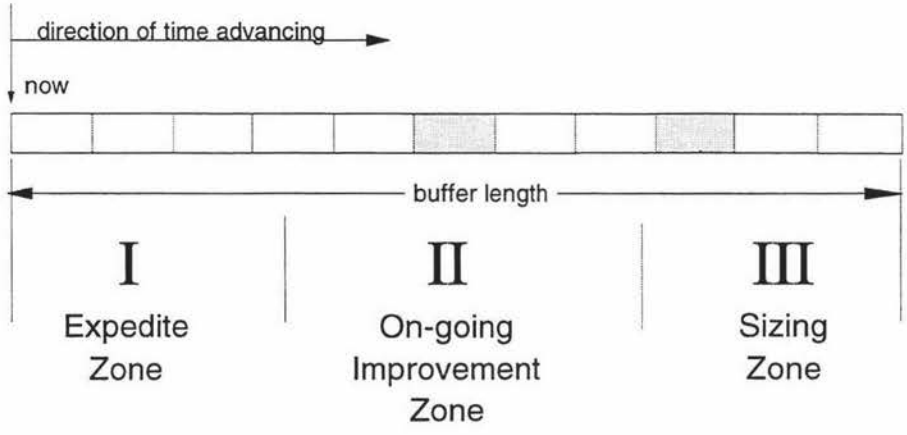


Figure 2.3: The Three Regions of a Buffer

Schragenheim *et al* (1990) characterised holes in the three regions by their potential impact on the entire production schedule. Holes in Zone III reflect variability inherent within the production process, and jobs will normally be missing from this region. Holes in Zone II reflect variability that has the potential to jeopardise throughput and should be the target for on-going improvement efforts. A production manager should investigate the reasons for holes in Zone II, and plan to expedite the job if necessary. A hole in Zone I requires urgent managerial attention to expedite jobs to the constraint and avoid a potential loss in throughout. The level of urgency will depend on the time remaining before the "hole" reaches the end of the buffer. Comparing actual buffer content to scheduled content makes it possible to systematically direct expediting in the short term, and to direct longer-term, on-going, improvement efforts.

To prioritise the importance of a buffer's holes Umble *et al* (1990) developed a measure called a "disruption factor", calculated by:

Size of the hole x Lateness **(2.2)**

Buffer profiles coupled with the "disruption factor" appear to be a potential means of identifying and ranking production disruptions.

Once disruptions have been ranked, basic problem-solving techniques, such as brainstorming, cause-and-effect diagrams, and basic statistics, can be employed to identify sources of disruptions. The use of disruption factors, combined with quality improvement tools, can thereby produce gains of significant value, as disruptions that potentially reduce throughput are addressed.

Schragenheim *et al* (1990) suggest that if a disruption can not be eliminated in the short-term, and continues to result in holes progressing to a buffer's expediting zone (Zone I), buffer length should be increased. An increase in buffer size, however, will decrease responsiveness to customer demands.

Solvable problems, however, represent potential gains to a company. Removing a source of disruption will result in lower statistical fluctuations, smaller buffers, reduced lead times, and increased responsiveness to customer demand. An improvement in throughput and lowered operating expense are likely to follow, resulting in greater net profit and return on investment. Hence, focused continuous improvement under Drum-Buffer-Rope has the potential to move a company towards improved competitiveness, profitability, and sustainability.

2.7 Discussion and Research Agenda

Much emphasis in manufacturing research has been placed on building stylised models and applying sophisticated solution approaches to solving the buffer management problem. Methods such as dynamic programming, experimental design, direct searches, and analytical models do not supply production managers of small to medium-sized companies with the tools needed to effectively manage buffer resources. As a result, most practitioners still use intuition to solve the BMP.

The Drum-Buffer-Rope methodology is perhaps the most practical production control methodology to be developed in recent years. Case studies (Ashcroft, 1989; Reimer, 1991) have indicated that its appeal rests in its immediate applicability and intuitive feel.

Adopting a DBR approach to buffer design implies that:

1. throughput is the objective function of preference;
2. buffer size is defined in terms of protection time and relates to throughput, hence profit;
3. constraint capacity should be exploited to maximise throughput;
4. material release and non-constraint activities should be subordinated to the constraint's schedule to realise this potential throughput;
5. buffers should only be located in front of constraint work-stations, at assembly points using constraint processed parts, and in the shipping area; and
6. buffer size is a trade off between product's lead time and protection of the constraint (alternatively the shipping schedule).

Research into the BMP must ultimately be justified in terms of an implementable solution to the BMP. Rather than developing results applicable to solving overly stylised mathematical models, research into the BMP should develop practical heuristics. The required solution must minimise lead time at the same time as protecting throughput and encouraging continuous improvement. Using the DBR approach as the underlying theory, practical research needs to be carried out to:

- develop heuristic methods for setting initial buffer sizes;
- characterise the relationship between buffer size and:
  - the amount of protective capacity at feeding work-stations,
  - the variability in work-stations feeding the constraint work-station;
- develop means to practically implement a buffer that will be monitored for day-to-day expediting decisions and longer term continuous improvement activities;
- develop structured approaches to changing the size of the buffer;
- develop heuristics for judging when and by how much buffer sizes should be changed; and
- investigate the relationship between the size of the manufacturing facility's three buffers (constraint, assembly, and shipping).

Although sophisticated mathematical techniques can be applied to stylised models in each of these research areas the end result must be easily-applicable heuristics. Future

researchers must base their assumptions and choice of objective function on real problems faced by actual manufacturing facilities.

2.8 Summary

Buffer management has real and measurable consequences for the management of manufacturing facilities. For over 30 years researchers have investigated the BMP. A reason given for the researchers using sophisticated OR techniques to investigate simple-stylised problems is to gain an understanding of the BMP. Using this understanding it should then be possible to develop usable heuristics for application in the more complex environment of an actual facility. The number of usable heuristics to come from this avenue of research is limited.

It has been shown that the DBR provides a robust and usable set of tools for the operation of a manufacturing facility. Instead of locating buffers in front of every work centre buffers should be located in front of the production constraint, at assembly points using constraint processed parts and in the shipping area. This limited number of buffers considerably reduces the solution space. Research into the BMP should use the DBR methodology as the underlying theory, hence it is adopted as the base model for this research.

Chapter Three

Fuzzy Logic

Subject Review

3.1 Introduction

A brief survey of fuzzy logic applications demonstrates the breath and width of the potential promise of fuzzy logic (Marks, 1994). Fuzzy logic applications are found in areas as diverse as robotics, industrial engineering and organisational psychology.

Fuzzy logic is not universally accepted as a valid approach. In particular, members of the statistical community object to fuzzy logic, asserting that fuzzy logic is nothing more than subjective probability. Further, Lindley (1987), a prominent statistician, comments:

> "anything fuzzy logic can do, probability can do better" (pp. 81)

However, there have been numerous successful applications of fuzzy logic, especially in the Japanese consumer electronic market. Brown *et al* (1995) gives a brief review:

- fuzzy washing machines that automatically determine an optimal cycle time based on the colour of the waste water and weight of the load;
- efficient fuzzy air conditioning systems;
- fuzzy subway controllers that produce a smoother ride and use up to 10% less control energy;
- autofocus mechanism in video cameras that minimise the effect of "shaky" hands; and
- fuzzy controllers for automatic gear shifting, antilock breaking, steering etc. in automobiles.

Fuzzy logic has also been applied in a number of production and operations management studies. Examples include production scheduling and production planning (see Custódio *et al*, 1994; Ward *et al*, 1992; Ishibuchi *et al*, 1994; and Slany *et al*, 1992).

The results are encouraging and help to establish the validity of further investigation into a fuzzy logic approach. As a result this thesis adopts a pragmatic view of fuzzy logic; the past successes of fuzzy logic are difficult to argue with. This research aims to assess the applicability of a fuzzy logic approach to practical buffer sizing.

Chapter two concluded that the utility of an OR approach to the BMP is questionable. The complexity and uncertainty associated with the BMP limits the usefulness of OR-

based solutions for developing practical and implementable heuristics for buffer management.

In this chapter the fundamentals of fuzzy logic are examined. Section 3.2 examines the nature of fuzziness and section 3.3 discusses the fuzzy logic modelling process. Section 3.4 concludes with the specific advantages of applying fuzzy logic to the BMP. Chapter four details the implementation of fuzzy buffer sizing model.

3.2 Fuzziness

Practitioners frequently use intuition to locate and size buffers. Practitioners have a remarkable ability to round off detail and distil the essence of the BMP. This creative flexibility produces an implementable solution to the BMP. Intuitive solutions, however, do not guarantee the best possible performance. Solutions will certainly not be optimal and solution quality will sometimes be inconsistent. A need therefore exists to combine the structure and consistency of an OR-based approach with the intuition and experience of practitioners.

The ability to round off details relates to a form of imprecision known as fuzziness. Fuzziness differs from measurement error or probabilistic uncertainty and is a form of "descriptive imprecision" that is largely the product of how people think and solve problems. Cox (1994) argues that people tend to view the world in terms of fluid or general categories with shifting boundaries. In this way complexity can be reduced through the use of semantic simplifications or natural language labels which describe a given concept. For example, today is *cold* which serves to capture the essence of what is measured or calculated.

Fuzziness is a useful problem solving tool when:

- system complexity and the vagueness of the objective function (for example, the appropriate buffer size) may mean that there is no adequate mathematical representation of the system; or
- there is little utility in developing complex mathematical models because the quality of input data is poor.

A popular example, often used to explain fuzziness, is temperature. People rarely talk about temperature as a numerical quantity. Instead, temperature is delineated into a number of general categories such as hot, warm, mild and cold. This is a sensible approach as few people carry thermometers and appreciate the complex nonlinear relationship between temperature and its many determining factors. There is also little utility in knowing this relationship since temperature can readily be expressed in descriptive terms such as quite cold, fairly mild and very hot.

Complexity associated with temperature can be managed using semantic simplifications. In this way, fuzziness balances the need for significance with precision: a point humorously illustrated in Mathworks (1995).

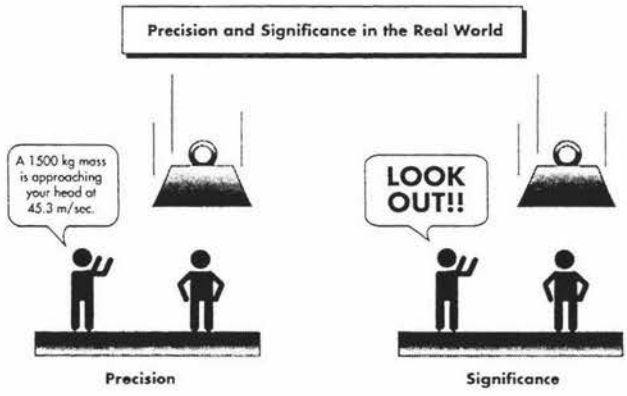


Figure 3.1: Balancing Significance With Precision (Mathworks, 1995)

Fuzziness also reflects a fundamental property of continuous variables. In this regard, fuzziness is an intrinsic form of descriptive imprecision. This has important implications for manufacturing systems research as a number of performance variables are continuous in nature.

To illustrate this form of imprecision consider the *truth* or *falsity* of the following statement:

"Due date performance is excellent"

In this example, assume that due date performance is measured using the percentage of on-time deliveries. Table 3.1 lists the percentage of practitioners (out of 100) that agreed with the proposition, due date performance is *excellent* (expert interpretation of the due date metric).

Percentage of On-Time Deliveries	**Percentage of Planners Who Agree with the Proposition**
99	95
90	80
85	65
50	0

Table 3.1: Expert Interpretation of the Due Date Metric

Intuitively, Table 3.1 gauges how compatible due date performance is with the underlying vague concept of *excellent*. Table 3.1 can be extended to include opinions on other realisations of the due date metric, as in Figure 3.2. This illustrates the idea of graded membership: how similar due date performance (element) is compatible or similar to the concept of *excellent* (set).

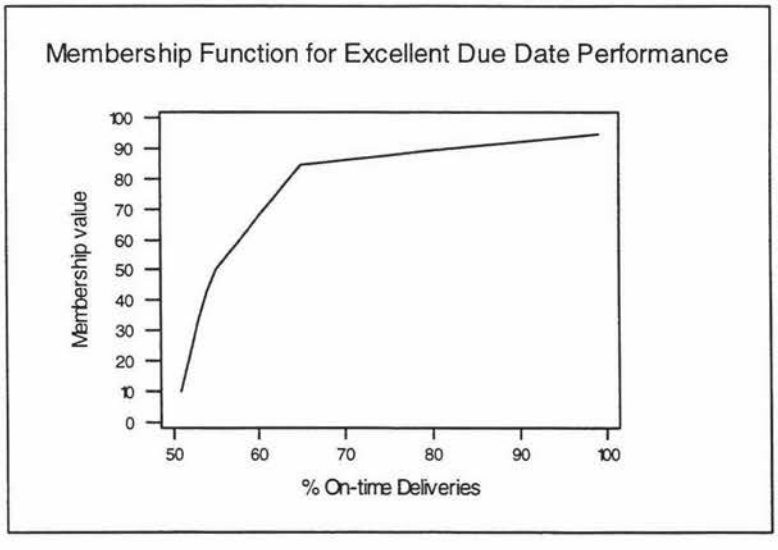


Figure 3.2: Graded Membership Function for *Excellent* Due Date Performance

Figure 3.2 is a graphical representation of a fuzzy set. Evans *et al* (1989) formally defines a fuzzy set as:

> "A fuzzy subset A of a universe of discourse U is defined by a membership function f_A: U→[0,1] which associates with each element u of U a number $f_A(u)$ in the interval [0,1], where $f_A(u)$ is defined as the grade of membership of u in A." (pp. 4)

A definition of the universe of discourse is given in section 3.3.1. A fuzzy set captures the essence of fuzziness. A fuzzy set is nevertheless a precise mathematical representation which models real world concepts described by natural language labels.

3.2.1 *Fuzzy and Boolean Sets*

In its broadest sense, the concept of *excellent* is a set. It is not, however, a Boolean or crisp set. Membership in the due date performance is *excellent* set is not abrupt. The transition between *excellent* due date performance and not *excellent* due date performance is gradual and represented in the interval [0,1].

The vertical axis of Figure 3.2 represents how similar or compatible (or the membership value of) the due date metric is to the concept of *excellent*. In this regard, a value of [1] indicates that due date performance is truly *excellent*. Similarly, a membership value of [0] indicates that due date performance is truly not *excellent*.

Figure 3.2 is a natural representation of the concept of *excellent* due date performance that closely conforms with intuition. It is an accurate reflection of reality, unlike OR techniques based on Boolean or crisp sets.

In a crisp set, due date performance belongs either to the set *excellent* or its complement, not *excellent*. The transition between the two sets is abrupt and is governed by a boundary value τ. If the percentage of on time deliveries, x, is less than τ then it follows

that due date performance is not *excellent*. Likewise, if x equals or exceeds τ then due date performance is *excellent*. For illustrative purposes assume that τ is equal to 90%.

For crisp sets, inclusion in the set *excellent* can be formally expressed by the characteristic function, $\mu_{excellent}(x)$. That is,

$$\mu_{excellent}(x) = \begin{cases} 1 \text{ if } x \geq 90 \\ 0 \text{ if } x < 90 \end{cases} \quad \textbf{(3.1)}$$

$\mu_{excellent}(x)$ is represented by Figure 3.3:

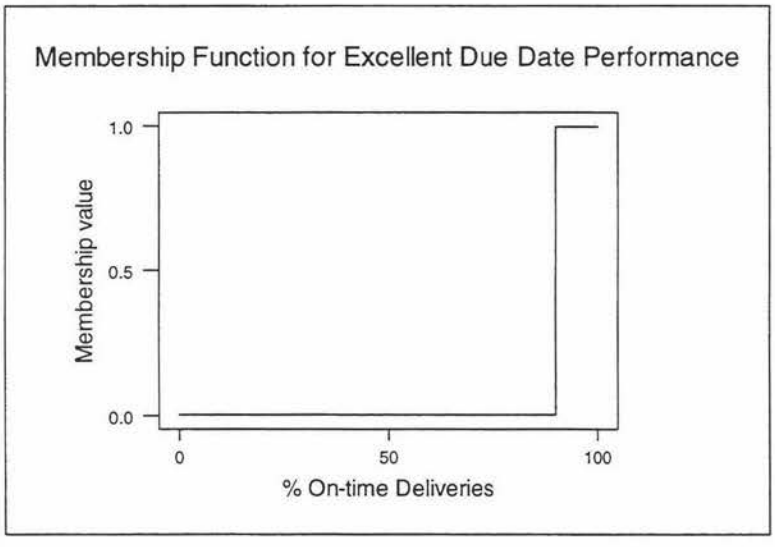


Figure 3.3: Boolean Membership Function for *Excellent* Due Date Performance

The transition between *excellent* and not *excellent* is not representative of the semantic meaning of the natural language label *excellent*. If x equals 89%, intuitively due date performance is still *excellent*; however, $\mu_{excellent}(x)$ classifies the due date performance as not *excellent*. Similarly, if x equals 89.9% due date performance would still be classified as not *excellent*. Clearly the intended meaning of *excellent* is lost. Boolean or crisp sets destroy the ordering information implicit with real world descriptions (Brown *et al*, 1995). As a result they are not representative of natural language labels that are used to solve real world problems.

For complex systems - those found in manufacturing systems - a large number of boundary conditions are needed to describe the behaviour of a given system. Estimating, these seemingly arbitrary boundary or cut off points is difficult, time consuming, and arguably inappropriate. Zadeh (1963) questions the utility of crisp sets. Fuzzy sets seem to be a more natural and structured way of approaching the modelling of real world problems.

It is important to note that Figure 3.2 is context dependent. Its shape and domain is not universal and will vary between markets, manufacturers and with time.

3.3 Fuzzy Modelling Process

One of the strengths, and yet paradoxically, the primary weakness of fuzzy systems is number of parameters that define a fuzzy model. There is little structured advice for researchers constructing fuzzy models. The advice available is anecdotal and drawn from many successful case-studies. A number of researchers, however, have noted that fuzzy systems (unlike OR models) are relatively tolerant to model parameter approximations. For example, the shape of fuzzy membership functions has relatively little effect on model performance.

More recently research has concentrated on mathematically designing classes of fuzzy systems - giving structured advice to fuzzy researchers and practitioners designing fuzzy systems. Lewis *et al* (1996) presents one general case of fuzzy controllers. They conclude that unevenly spaced triangular membership functions and product inferencing (as opposed to the original max. and min. rules proposed by Zadeh (1963)) should be used.

Cox (1994) provides a general fuzzy system design methodology based around the fuzzy modelling process. A good overview of the fuzzy modelling process is given by Brown *et al* (1995) and as a result this modelling process will not be examined in detail here. Instead the essential components are reviewed following Cox (1994); these components are:

- fuzzification of input variables;
- fuzzy inference system; and
- defuzzification of output variables.

Conceptually, a fuzzy system can be visualised as an open system consisting of a series of real valued inputs, a transformation process, and a series of outputs. A model of the transformation process is of particular interest to researchers as it can be used for prediction. In fuzzy systems (as opposed to statistical or neural network models) the transformation process is understood (at least partially) in terms of heuristic or expert knowledge through the use of fuzzy *if then* production rules.

3.3.1 *Fuzzification*

The process of fuzzification allows vagueness to be captured in a structured and consistent manner using fuzzy sets. Fuzzy sets are mathematically represented by a one to one mapping between a real valued input and a graded membership value.

In a fuzzy system, an input or output variable is represented by a series of fuzzy subsets[1] that span the operating range (universe of discourse) of the variable. Collectively, fuzzy sets form what is known as a fuzzy term set. Individually, each fuzzy set is labelled with a linguistic variable which reflects its intended meaning.

Figure 3.4 is an example of the fuzzy term set due date performance, defined by three fuzzy sets (and linguistic variables): *poor*, *acceptable* and *excellent*.

[1] There is no fuzzy set theory; only fuzzy subset theory, though the two are used interchangeably.

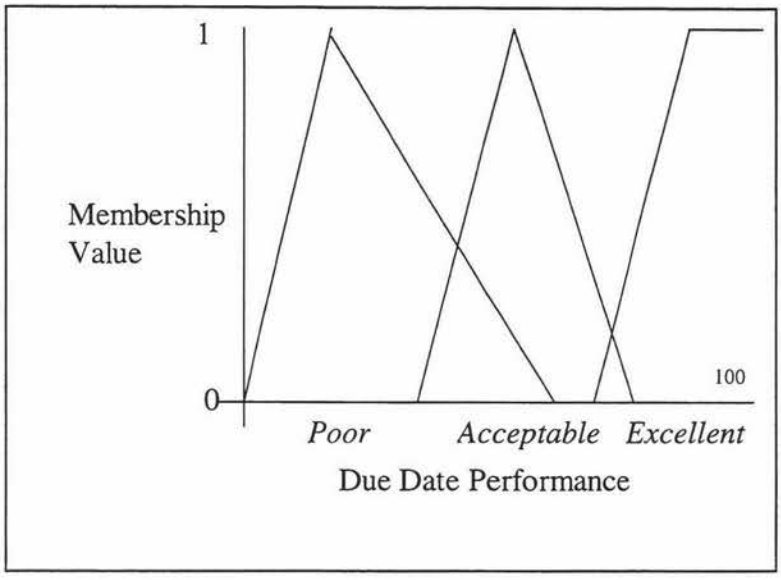


Figure 3.4: Fuzzy Term Set Due Date Performance

There is little advice for designing fuzzy sets. Fuzzy sets are application dependent and rely heavily on context. However, two general design decisions need to be considered before a fuzzy set is constructed: applicability of a fuzzy set; and how the semantic meaning of a fuzzy set is mathematically encoded, that is, the membership function.

Applicability of the Fuzzy Set

There are three closely related design decisions that define the applicability of a fuzzy set in a fuzzy system: (1) the number of fuzzy sets; (2) the domain of each fuzzy set; and (3) the overlap of each fuzzy set. Together these three design decisions reflect how complete the heuristic understanding of the transformation process is (or how certain the input data is). Figure 3.5 illustrates these design decisions.

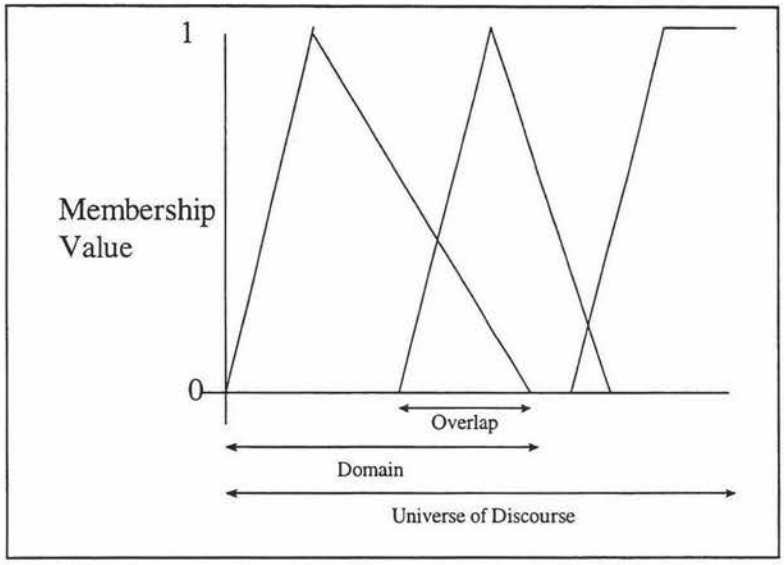


Figure 3.5: Domain and Overlap of Fuzzy Term Set Due Date Performance

The first decision determines how many fuzzy sets are used to span the universe of discourse. This will depend on the expert's ability to verbalise the fuzzy *if then* production rules that define the transformation process. If the transformation is poorly understood, fewer fuzzy sets should be employed; however, if the transformation is transparent, more fuzzy sets can be used. The limiting case is a deterministic model where the transformation process is completely understood. An example is the relationship between force, mass and acceleration. In this case, input variables can be modelled as crisp (singleton) sets. There is no intrinsic vagueness and a precise one to one mapping is possible.

The cognitive ability of the expert is not the only constraint on the number of fuzzy sets used to model input and output variables. Fuzzy systems also have the ability to interpolate between fuzzy sets. More fuzzy sets will not necessarily result in improved model performance. In fact the number of fuzzy sets is a trade-off between enhanced model performance and modelling "cost". Modelling cost can measured in terms of increased number of design decisions; memory requirements; and computational time.

The second decision, overlap (see Figure 3.5), also depends on the cognitive ability of the expert. The amount of overlap is an important design decision that differentiates whether one fuzzy set can be distinguished from other neighbouring fuzzy sets. With

little or no overlap, fuzzy sets become disjoint and sudden and unpredictable changes in rule behaviour can occur (Cox, 1994). Disjoint fuzzy sets (as with traditional OR techniques that use Boolean logic) also fail to model vagueness associated with an input variable. Overlap also quantifies the amount of intrinsic vagueness associated with articulating the fuzzy model both in terms of the knowledge of the expert and the quality of the input data. As a result, fuzzy system can also deal with uncertain input data.

The domain (see Figure 3.5), or bound, of a fuzzy set is a collection of monotonically increasing real values that define the applicability (or alternatively, the operating range) of the semantic concept modelled by the fuzzy set. Fuzzy domains should be conformally mapped. That is, the domains of the output variables should correspond to the domains of the input variables. If this is not the case then fuzzy *if then* production rules have little meaning.

Membership Functions

Membership functions are mathematical representations of fuzzy sets. A variety of membership functions have been proposed including triangular, sigmoid and gaussian curves (see Figures 3.6, 3.7 and 3.8). Cox (1994) gives a thorough review of the many common cases of membership functions that have been suggested.

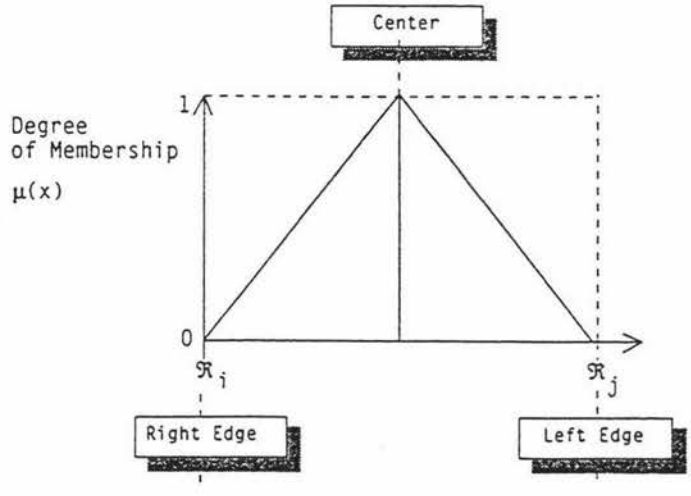


Figure 3.6: Triangular Membership Function (Cox, 1994)

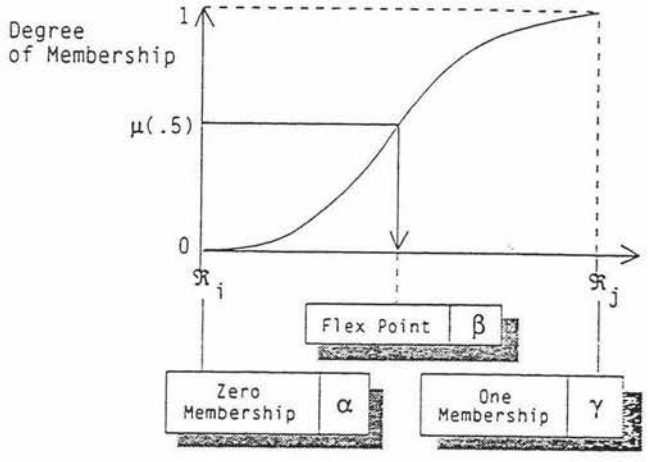


Figure 3.7: Sigmoid Membership Function (Cox, 1994)

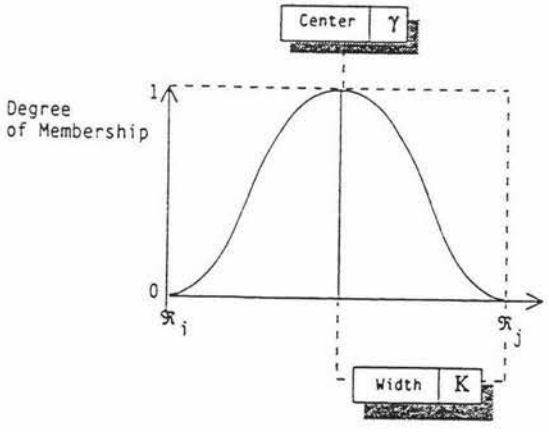


Figure 3.8: Gaussian Membership Function (Cox, 1994).

The shape of the membership function encodes the meaning of the underlying semantic concept. The closer the membership function fits the underlying concept described by the fuzzy set, the better the fuzzy model will perform. Selecting the most appropriate membership function is an open question as membership functions are application dependent and rely heavily on context. However, they should possess certain desirable properties, such as a partition of unity and compactness. Researchers historically have adopted triangular membership functions and more recently B-splines (Brown *et al*, 1995).

Cox (1994) provides a succinct review of the numerous techniques that have been applied to determine the appropriate shape of a membership function. Probability frequency distributions, neural network models, mathematical surface sampling, subjective approximation and voted for distributions have been applied. Subjective approximation closely corresponds to the way experienced designers fit membership functions and is certainly the best method for fitting membership functions in terms of opportunity costs. For example, Cox (1994) argues that the added mathematical sophistication of neural networks has little utility.

3.3.2 *Fuzzy Inference System*

The real power of fuzzy sets is the ability to manipulate them in similar ways to Boolean sets, using fuzzy logic. This allows expert or heuristic knowledge to be implemented on computers and it is the fuzzy inference system that controls this.

The fuzzy inference system maps a series of real valued inputs to a number of real valued outputs. A precise representation of the mapping sought after by standard OR techniques is not needed. Rather, the essence of the mapping can be instilled as a series of fuzzy *if then* production rules. As an example consider two fuzzy *if then* production rules that relate due date performance with manufacturing lead time to determine the competitiveness of a manufacturing facility.

> If due date performance is *excellent* and manufacturing lead time is *short* then competitiveness is *high*
>
> If due date performance is *acceptable* and manufacturing lead time is *medium* then competitiveness is *average*

This research considers the case where a number of input variables (antecedents) are related to one output variable (consequence) through the use of fuzzy *if then* conditional production rules. That is,

IF ($\mathbf{x_1}$ is A^i_1 AND $\mathbf{x_2}$ is A^i_2 AND ... AND $\mathbf{x_n}$ is A^i_n) THEN ($\mathbf{y}$ is B^j) c_{ij} **(3.2)**

where A^i_n is the n^{th} fuzzy set (of $\mathbf{x}_i$ the i^{th} fuzzy input term set) related to B^j (the j^{th} output fuzzy set of the fuzzy output set $\mathbf{y}$), with confidence c_{ij}. A collection of fuzzy *if then* production rules is known as a rule base.

There are two separate processes used by the fuzzy inference system: implication and composition.

Implication

As with Boolean sets four set operations can be carried out on fuzzy sets. This enables the relative truth of the antecedents to be evaluated and equated to the output variable - establishing a mapping between the input and output variables. This process is known as implication. The Zadeh (1963) style operators, which have similar properties to Boolean operators, are commonly used in fuzzy inferencing systems.

For illustratively purposes, define $\mu_X(x)$ and $\mu_Y(y)$ as the membership function of two fuzzy sets, X and Y respectively.

1. *Complement*:

$$\mu_X(x)' = 1 - \mu_X(x) \quad \textbf{(3.3)}$$

$$\mu_Y(y)' = 1 - \mu_Y(y) \quad \textbf{(3.4)}$$

The complement corresponds to the Boolean NOT operator. The resulting membership function is interpreted as how dissimilar x (y) is with X (Y).

2. *Intersection*:

$$\mu_X(x) \cap \mu_Y(y) = \min (\mu_X(x), \mu_Y(y)) \quad \textbf{(3.5)}$$

The intersection corresponds to the Boolean AND operator. $\mu_X(x) \cap \mu_Y(y)$ is interpreted as the compatibility that an element belongs to both X and Y.

3. *Union*:

$$\mu_X(x) \cup \mu_Y(y) = \max (\mu_X(x), \mu_Y(y)) \quad \textbf{(3.6)}$$

The union corresponds to the Boolean OR operator. $\mu_X(x) \cup \mu_Y(y)$ is interpreted as the compatibility that an element belongs to either X or Y.

4. *Product*:

$$\mu_X(x) * \mu_Y(y) = \mu_X(x)\mu_Y(y) \quad \textbf{(3.7)}$$

The product operator is similar to the intersection operator yet is a more stringent variant of the AND operator.

Cox (1994) and Brown *et al* (1995) review a number of other operators. Examples include bounded sums and Yager compensatory type operators.

The fuzzy operators are used to compute the relative truth of each rule's antecedent in the fuzzy rule base. The truth of each fuzzy output set is equated or correlated with the antecedent of each rule. Taking the previous example, if the relative truth of (due date performance is *excellent* and manufacturing lead time is *short*) is [0.3] then competitiveness is [0.3] compatible with the concept of *excellent*. In this way, a mapping is established between input variables and the output variable.

The relative truth of a rule's antecedent is also influenced by two parameters: a rule's alpha cut and confidence.

If relative truth of an antecedent is greater than zero then the rule is said to have fired. In many cases it is undesirable for fuzzy rules with antecedent of small relative truth to contribute to the fuzzy output set. An alpha cut will ensure this will not happen and effectively restricts the range over which the rule is active. That is, the rule will not fire unless the relative truth of the antecedent is greater than the alpha cut.

The rule's confidence measures the strength of the association between the input and output variables - or alternatively, "faith" in the validity of the rule verbalised by the expert. In some cases a rule will have a confidence of zero; in this case it is ignored by the fuzzy rule base. The rule's confidence is applied by multiplying the relative truth of the antecedent with the confidence.

Composition

After the implication operations have been applied to the relative truth of each antecedent, and the relative truth of each consequence has been equated, the relative truth of each fuzzy output set is combined using the process of composition based on either max. composition or sum composition which update the output fuzzy space. Details of each can be found in Cox (1994).

3.3.3 *Defuzzification*

Composition produces an output fuzzy set. In most cases, knowing the relative truth of the output set has little utility. A fuzzy output set is defuzzified to produce a real value that is used in the decision making process.

There are two common methods of defuzzification: the centroid and maximum method. Both have advantages and disadvantages. The centroid is the most commonly used method where the crisp estimate of the true output variable is calculated by finding the centre of gravity of the fuzzy output set.

3.4 Fuzzy Logic Applied to the Management of Production Buffers

An obvious issue in fuzzy logic design methodology is the question of whether it is an appropriate technique to use. When the system under study is defined and well-behaved, the use of fuzzy logic is arguably inappropriate. For a well defined problem OR-based solutions can readily be used. OR techniques have been extensively applied to a variety of manufacturing based problems. Winston (1991) provides an informative, yet brief overview of successful OR applications.

Manufacturing systems are, however, characterised by intrinsic imprecision which is further confounded by complexity and non-linearities. Manufacturing systems are neither well understood nor well defined. Fuzzy logic is then an appropriate solution approach.

This section details why a fuzzy logic solution approach to the Buffer Management Problem (BMP) is appropriate.

3.4.1 *Fuzzy Systems are Easier to Understand*

Research results that use complex mathematical and statistical techniques to solve the BMP have failed to achieve widespread acceptance (Hurley, 1996). Practitioners still prefer to use intuition when solving the BMP despite 35 years of research.

Fuzzy logic is a human centred and often transparent solution method which directly encodes expert knowledge through the use of fuzzy sets and fuzzy *if then* production rules. Fuzzy systems are structurally simpler, require fewer rules, and closely conform to expert understanding. A fuzzy solution is intuitive and readily understood by practitioners. This addresses the need for an implementable solution for the mathematically unsophisticated user. Further, fuzzy logic provides the mathematical formalism for implementing this heuristic advice on computers.

3.4.2 *Fuzzy Input Variables*

Fuzzy logic addresses a significant problem of conventional Boolean logic: an ability to model variables characterised by intrinsic imprecision.

As previously discussed, metrics that model continuous quantities are characterised by intrinsic imprecision. Consider protective capacity and up-stream variability - the two dependent variables that influence buffer size.

Exactly what point protective capacity levels are *high*, or *medium*, is debatable and subject to expert interpretation. Likewise, whether variability levels are *disruptive* or *minimal* will depend on the actual manufacturing facility. The ideas of graded membership and context dependency provide an alternative, more natural, way of interpreting the meaning and significance of the factors that influence buffer size.

Protective Capacity

Protective capacity is a subtle and often overlooked factor in the BMP. It is a relatively new concept and few researchers have examined its implications (Atwater, 1991).

APICS (1990) dictionary defines protective capacity as:

> "The difference between the output rate of a non-constraint resource and the constraint resource that is required to achieve a given level of output at the constraint."

Protective capacity is not the difference between the capacity of non-constraint work-station and the capacity of the constraint work-station as intuition would suggest. Instead protective capacity is that part of the unused capacity that enables a non-constraint to work faster than the constraint work-station in order to rebuild the buffer following production disruptions. The remaining spare capacity is called "excess capacity". The relationship between protective and excess capacity is given by equation 3.8.

Constraint Capacity - Nonconstraint Capacity = Protective + Excess Capacity **(3.8)**

where capacity is measured in minutes of processing required per part.
A literature search reveals that there are no protective capacity metrics that model the APICS (1990) definition of protective capacity. Instead protective capacity levels are estimated intuitively or by the left hand side of equation 3.8: an absolute upper bound on protective capacity. Atwater (1991), for example, who conducted one of the first studies on protective capacity utilised equation 3.8.

A protective capacity metric is required to consider both the frequency of severity of production disruptions. Because of the existance of dependent events and interactions the protective capacity of a non-constraint is also a function of the distance to the constraint work-station. Downstream work-stations require extra capacity to deal with disruptions that upstream work-stations could not cope with.

Derivation of a protective capacity metric would be difficult and complex. The mathematics required to amalgamate the many production disruptions is likely to produce expressions that are analytically intractable; any assumptions that lead to tractability are likely to result in unrealistic results. For example, exponentially distributed processing times.

Equation 3.8 represents an absolute upper bound on protective capacity levels. Consequently, for a given manufacturing facility its real or objective meaning is unknown and subject to interpretation. Protective capacity levels can be estimated in a more structured manner by fuzzifying equation 3.8. This avoids the need to make unrealistic assumptions in order to develop a protective capacity metric.

The Effect of Upstream Variability

There are many sources of variability found in manufacturing facilities. A few examples include:

- re-work resulting from quality defects;
- variable setup and processing times;
- work-station failure;
- operator, raw material and tool inavailability; and
- unplanned queues that form in front of non-constraint work-stations.

Variability results in uneven and unpredictable flow of work causing work to arrive later than scheduled at the buffer. Umble *et al* (1990) proposed a metric to measure the impact of variability on the flow of work in a manufacturing facility. A possible form is given by equation 3.9 (discussed in chapter two):

$$\text{Size of the Hole x Amount of Work} \qquad \textbf{(3.9)}$$

The actual meaning of equation 3.9 is unknown and requires expert interpretation as the "effect" (that is, cost) of constraint starvation will vary from manufacturer to manufacturer.

3.4.3 *Analytical Intractability*

The buffer sizing problem is simple to formulate but difficult to solve. At the heart of this is the relationship between the amount of protective capacity and the effect of variability that determines buffer size. This is an issue which has not be satisfactorily addressed by research. It is very likely to be analytically intractable.

The fuzzy inference system provides a mechanism that is capable of accommodating the intractable relationship between variability and protective capacity and buffer size. The fuzzy inference system maps a series of real valued inputs to a real valued output. A precise representation of the mapping sought after by standard OR technique is not needed. Rather the essence of the mapping is instilled by a series of fuzzy *if then* production rules.

The primary advantage of this is that we need not rely on developing over stylised analytical models with restrictive assumptions. Further as fuzzy models capture expert

knowledge which is context dependent, this research is not limited by non-generalisable results.

3.4.4 *Room to Grow*

Fuzzy logic also has a number of benefits as a research tool. This is in part a consequence of the stringent nature of the fuzzy modelling process. In order to accurately model expert knowledge, a thorough understanding of the BMP is required. This enhances our understanding of the factors that influence buffer size and shifts the focus away from applying stylised solution techniques ensuring that the BMP does not become solely an issue of mathematical sophistication.

Fuzzy models can change with our understanding of the problem domain. As all production rules are simultaneously considered, a fuzzy model can be easily augmented with further factors as their importance becomes apparent. Another advantage in applying a fuzzy approach to the BMP is that fuzzy systems form the framework on which hybrid systems, incorporating analytical and empirical expressions and probability models, can be built (Cox, 1994).

Chapter Four

Fuzzy Buffer Sizing Model Implementation

4.1 Introduction

This chapter develops an intuitive and initial fuzzy buffer sizing model based on unoptimised memberships and standard and commonly used inference and defuzzification strategies.

A standard fuzzy model is developed for two reasons: (1) the primarily purposes of this research was to assess the applicability of a fuzzy solution approach to the BMP; and (2) an initial model should be simple.

There are four fundamental steps in the fuzzy modelling process.

1. selection of input and output variables;
2. fuzzification of input variables;
3. fuzzy inference; and
4. defuzzification of output variables.

This chapter details the model's implementation in MATLAB's Fuzzy Logic Tool Box (Mathworks, 1995).

4.2 Selection of Input and Output Variables

An input/output analysis is an important, though often difficult, first step in the fuzzy modelling process. A sound substantive and theoretic basis is essential if practical and implementable solutions are to be developed for the BMP. Here, the production application of the TOC, DBR is used as the base model.

4.2.1 *Output Variables*

The three questions of the BMP are:

1. What objective function to use?
2. Where to locate buffers?
3. What is the appropriate buffer sizes?

DBR provides precise answers to the first and second questions. Throughput is the objective function of choice and buffers are strategically located at the constraint work-station, assembly (using constraint processed parts) and shipping areas.

However, DBR fails to answer the third question satisfactorily, offering only vague and non-specific advice for buffer sizing. Buffer size is a suitable fuzzy output variable.

There are two ways the buffer size output can be represented. First, an absolute buffer size can be calculated; second, the BMP could be formulated as a fuzzy control problem and the buffer size expressed in terms of "increase or decrease".

The latter approach is intuitive and gives practitioners a sense of control over the implementation of the buffer size. However, the practicalities of changing the buffer size limits how frequently the buffer size can be changed (see Section 8.5.3 for a discussion on changing the buffer size). Buffer size is better represented as an absolute quantity (and should implemented in a controlled fashion).

4.2.2 *Input Variables*

For the purposes of this research, two variables are considered sufficient to adequately size buffers. That is,

$$\text{Buffer Size} = f(\text{Effect of Variability, Protective Capacity}) \quad \textbf{(4.1)}$$

Effect of Variability

When coupled with dependent events, up-stream variability causes work-orders to arrive at the constraint work-station later than scheduled. To ensure the constraint does not starve for work (thus protecting the validity of the constraint's schedule) work-orders need to be released a buffer time early. The more disruptive variability is to work flow, the larger the buffer that is needed; conversely, if variability is less disruptive, a smaller buffer should be sufficient.

The role of the buffer is to protect the output rate from the effect rather than the source of variability. The constraint work-station in this regard is myopic and only concerned with the availability of work in the buffer. There is little utility in considering every individual source of variability; rather the effect of variability can be quantified with an Umble *et al* (1990) style disruption factor (DF) of the form:

$$\text{DF} = f(\text{Lateness, Amount of Missing Work}) \quad \textbf{(4.2)}$$

The simplest realisation of equation 4.2 is equation 3.9, the product of lateness and the amount of missing work.

The effect of variability can be measured and summarised using an Umble style disruption factor. This generic "catch-all" measure of the effect of variability has a number of advantages. Firstly, equation 4.2 greatly simplifies the problem of accurately and timely quantifying variability. There is no further data collection requirements as this information is readily available in a typical DBR implementation[1]. Secondly, a catch-all measure of variability limits the number of fuzzy sets and rules which reduces model complexity and increases transparency and maintainability.

[1] Disruption factors are used in a DBR implementation to identify and focus quality improvement initiatives.

For the purposes of this research, as the effect of the coefficient of variation of processing times (cv) is intuitively understood by the researcher, the cv is adopted as the operational measure of variability. The fuzzy model can be easily extended to include the Umble disruption factor, a quantity that is readily understood by practitioners.

Protective Capacity

Protective capacity can be understood by considering the production disruptions that occur up-stream from the constraint, eroding the buffer. To recover from these disruptions, non-constraint work-stations require extra capacity - termed "protective capacity" - which ensure that the non-constraint work-stations can "work faster" than the constraint work-station to rebuild the buffer. Lower levels of protective capacity require larger buffers; conversely, high levels of protective capacity will allow smaller buffers.

Protective capacity, while difficult to measure, can be estimated using the manufacturing facility's mean protective capacity (MPC), equation 4.3. This is based on the mean percentage difference between non-constraint work-station and constraint work-station capacity. That is,

$$\text{MPC} = 100\left(1 - \frac{\text{mean nonconstraint capacity}}{\text{mean constraint capacity}}\right) \qquad \textbf{(4.3)}$$

Capacity is measured in terms of minutes required per part.

The remainder of this chapter details the MATLAB implementation using the Fuzzy Logic Tool Box (MathWorks, 1995).

4.3 Fuzzification

4.3.1 *Fuzzy Term Sets*

In the previous section, the two input variables, protective capacity and the effect of up-stream variability and one output variable, the absolute buffer size, were selected. The fuzzy model describes each input and output variable in terms of a series of fuzzy sets, or membership functions that are known as a fuzzy term set.

Accordingly, three fuzzy term sets with linguistic variables are proposed:

1. The protective capacity of feeding work-stations (MPC) {low, medium, large}.
2. The effect of up-stream variability (UV) {minimal, nominal, disruptive}.
3. Appropriate buffer size (BS) {small, medium, large}.

The remainder of this section details how the implemented fuzzy term sets (shown in Figures 4.1 to 4.3) were constructed in MATLAB.

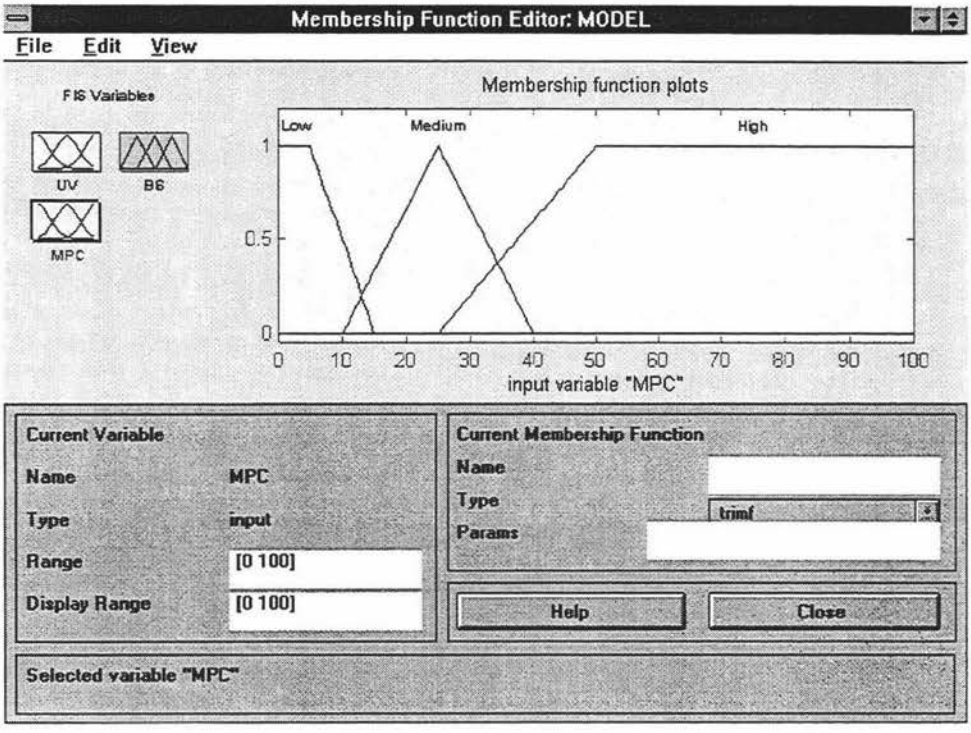


Figure 4.1: Mean Protective Capacity (MPC) Term Set

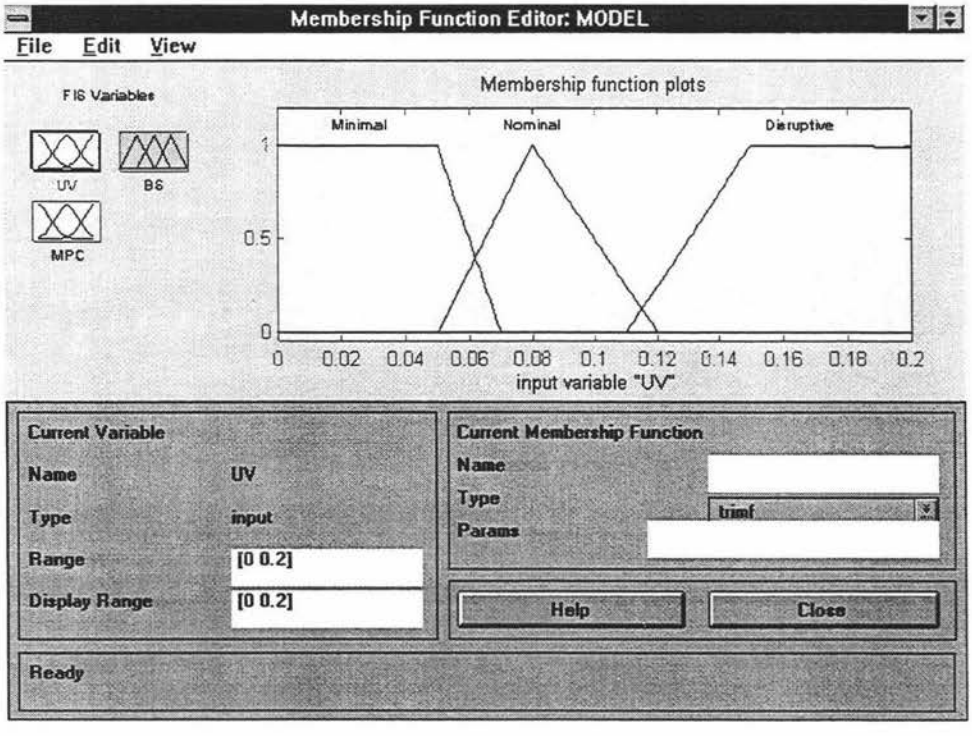


Figure 4.2: The Effect of Up-stream Variability (UV) Term Set

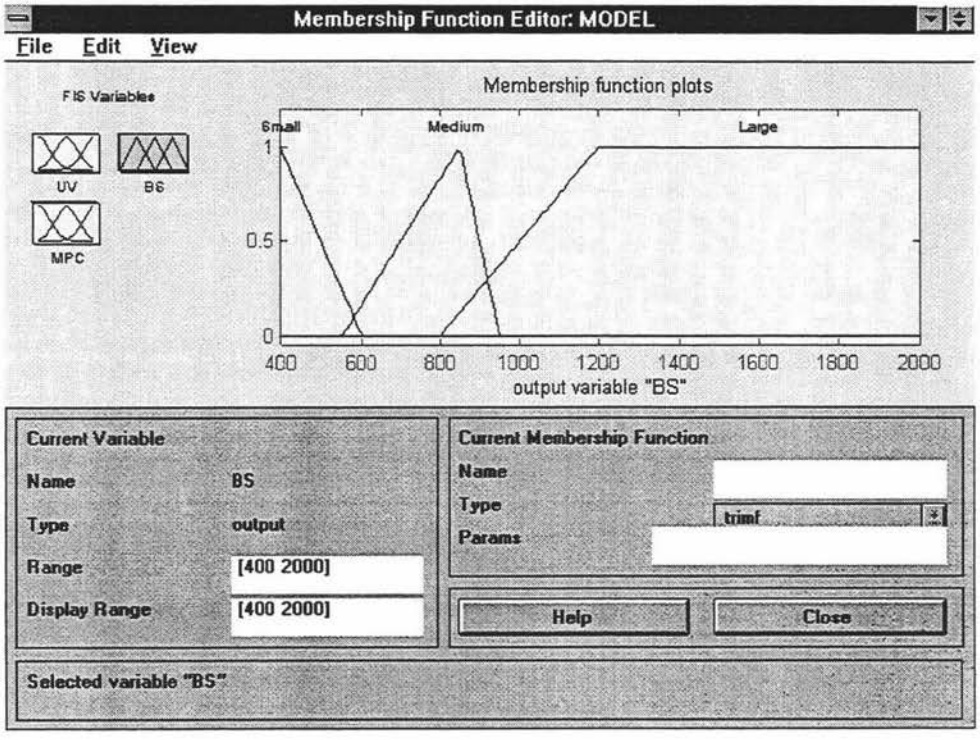


Figure 4.3: The Appropriate Buffer Size (BS) Term Set

The universe of discourse for each fuzzy term set is determined using *a priori* knowledge about the system. That is:

- MPC is logically bounded by the interval [0, 100]%;
- for the purposes of this research, cv is restricted to the interval [0.01, 0.2]; and
- buffer size is restricted to the interval [400, 2000] minutes in order to ensure that the buffer size is conformally mapped with the effect of up-stream variability and protective capacity fuzzy term sets.

Applicability of Each Fuzzy Set

Three fuzzy sets were used to represent each input and output variable. This decision was made after discussion with two practising production planners, who felt that they could confidently interpret protective capacity, up-stream variability and buffer size in terms of a maximum of three general categories. Using a small number of fuzzy sets does not present any problems and in practice the number of fuzzy sets should be determined by practitioner.

Chapter six develops a DBR-based simulation model that is used to test the effectiveness of the fuzzy logic buffer sizing model; accordingly the fuzzy model is developed for this environment. From experience gained from this simulation model, the domain and amount of overlap for each fuzzy set was determined, mimicking the way practitioners would specify the fuzzy model in practice (using expert knowledge).

As an example consider the fuzzy set *disruptive* up-stream variability. This fuzzy set is defined by a trapezoid bounded by [0.11, 0.2]. This fuzzy set shares a small amount of overlap with the *nominal* membership function as experience has shown that the effect of increasing cv becomes quite pronounced after cv equals 0.1. When cv greater than or equal to 0.15 variability can be completely described as *disruptive*.

A closely related decision to the domain of the fuzzy set is the amount of overlap. This is important to fuzzy systems. With little or no overlap, fuzzy sets become disjoint and fail to model the semantics of the fuzzy term set. This is particularly true for the BMP. Practitioners can not be expected to clearly differentiate between the differing levels of protective capacity; accordingly, a *medium* buffer size should also be considered to a degree as *small*. Further, intuition suggests that as protective capacity levels (for example) approach the meaning of *high* then logically the protective capacity levels should become less *medium*; some overlap therefore is obviously required.

However, too much overlap can cause problems. For example, no interpretation is possible if buffer size is *small* with membership value of [1] and also *medium* with a truth value above the alpha cut threshold. This can make the fuzzy model overly sensitive to perturbations in the input variables. To avoid this, a number of researchers and practitioner impose the constraint given by equation 4.4 which normalises the membership functions.

$$\sum_i \mu_i(x) = 1 \qquad \textbf{(4.4)}$$

In practice the actual degree of overlap depends on the fuzziness associated between each fuzzy set. Determining this is arguably subjective; one recommendation after Cox (1994) is

that overlap should average somewhere between 25% to 50%. Overlap for the fuzzy model presented here was determined intuitively and hence the membership functions are not necessarily normalised. The average overlap for each fuzzy term set is given in Table 4.1.

Fuzzy Term Set	Average Overlap
Up-stream Variability	27%
Protective Capacity	40%
Buffer Size	25%

Table 4.1: Fuzzy Term Set Overlap

4.3.2 *Membership Function Shape*

Membership functions are used to assess the compatibility or similarity associated with a given real-valued input and the semantic meaning of the fuzzy set. Determining the correct membership function is an open question that is dependent on the context in which the membership functions are used.

The fuzzy buffer sizing model utilises two simple membership functions: (1) triangular; and (2) trapezoidal membership functions. Both the triangular and trapezoidal membership functions are similar in structure but differ in their semantic meaning.

Research by Lewis *et al* (1996) indicate that triangular membership functions work well, and despite being simplistic, provide a good initial guess. A similar argument is put forward for trapezoidal membership functions. Both triangular and trapezoidal membership function are easily defined in practice: a mode, 5% and 95% easily translate as the most commonly occurring value and the extreme values that bound the concept. Triangular and trapezoidal membership functions are also been commonly used by researchers and practitioners. The initial fuzzy buffer sizing model that was developed was purposely kept unoptimised to enhance understanding of the BMP.

It is important to note that the initial fuzzy model is not restricted to using triangular membership functions. Any membership function can be fitted and this is a possible area for future work, though the benefit of doing this is likely to be small. An alternative method, the voted for distribution, can be used to construct membership functions for the BMP. Membership values are interpreted as the proportion of "voters" (that is, practitioners) who classify each value of a given domain according a semantic concept (for example, *low* protective capacity).

Imagine a situation where ten practitioners are asked to classify buffer size in terms of the fuzzy set *large*. Is 10 days large? A total of 5 practitioners indicate this is so. In this case, the membership value of the fuzzy set *large* for a buffer size of 10 days is [0.5]; this process continues until the complete membership function is defined.

While some information about the what the appropriate buffer size was used to tune the initial fuzzy model, the domains and overlap of each fuzzy set in retrospect could have easily be determined using intuition. This indicates that fuzzy model can be set up in practice drawing on practitioner expert knowledge.

4.4 Fuzzy Inference System

4.4.1 *Fuzzy Rules*

The relationship between protective capacity, variability and buffer size is likely to be analytical intractable. However, a partial understanding of the relationship between protective capacity, variability and buffer size can be drawn from DBR, the theoretic frame-work used in this research. Provided each fuzzy set is conformally mapped this understanding can be expressed as a series of fuzzy *if then* production rules that establish a relationship between the input fuzzy sets and the output fuzzy set.

With (3 protective capacity fuzzy sets) x (3 up-stream variability) by (3 buffer size) fuzzy sets, 27 rules are possible. Each rule is assigned a confidence (a relative truth), determined intuitively, by applying a probabilistic interpretation based on equation 4.5.

$$\sum_{j} c_{ij} = 1 \; \forall \, i \qquad \textbf{(4.5)}$$

Not all 27 rules need to be considered, as some rule confidences are obviously zero. For example, if the effect of up-stream variability is *disruptive* and protective capacity is *low* then it is unlikely that the buffer size would be either *small* or *medium*. Likewise, if the effect of up-stream variability is *minimal* and protective capacity is either *high* or *medium* then it also seems unlikely that the buffer size would be *large*. Rules of this type are assigned a confidence of zero and ignored, leaving only 17 non zero confidence fuzzy *if then* production rules. They are:

1. If the effect of up-stream variability is *disruptive* and protective capacity is *low* then buffer size is *medium* **0.1**

2. If the effect of up-stream variability is *disruptive* and protective capacity is *low* then buffer size is *large* **0.9**

3. If the effect of up-stream variability is *disruptive* and protective capacity is *medium* then buffer size is *medium* **0.7**

4. If the effect of up-stream variability is *disruptive* and protective capacity is *medium* then buffer size is *large* **0.3**

5. If the effect of up-stream variability is *disruptive* and protective capacity is *high* then buffer size is *medium* **1.0**

6. If the effect of up-stream variability is *nominal* and protective capacity is *low* then buffer size is *small* **0.1**

7. If the effect of up-stream variability is *nominal* and protective capacity is *low* then buffer size is *medium* **0.5**

8. If the effect of up-stream variability is *nominal* and protective capacity is *low* then buffer size is *medium* **0.4**

9. If the effect of up-stream variability is *nominal* and protective capacity is *medium* then buffer size is *small* **0.3**

10. If the effect of up-stream variability is *nominal* and protective capacity is *medium* then buffer size is *medium* **0.7**

11. If the effect of up-stream variability is *nominal* and protective capacity is *high* then buffer size is *small* **0.7**

12. If the effect of up-stream variability is *nominal* and protective capacity is *high* then buffer size is *medium* **0.3**

13. If the effect of up-stream variability is *minimal* and protective capacity is *low* then buffer size is *small* **0.1**

14. If the effect of up-stream variability is *minimal* and protective capacity is *low* then buffer size is *small* **0.9**

15. If the effect of up-stream variability is *minimal* and protective capacity is *medium* then buffer size is *small* **0.5**

16. If the effect of up-stream variability is *minimal* and protective capacity is *medium* then buffer size is *medium* **0.5**

17. If the effect of up-stream variability is *minimal* and protective capacity is *high* then buffer size is *small* **0.9**

The fuzzy production rules are executed in parallel and contribute to the fuzzy buffer size output set. The combined effect of each rule for a given value of protective capacity or cv is shown in Figures 4.4 and 4.5.

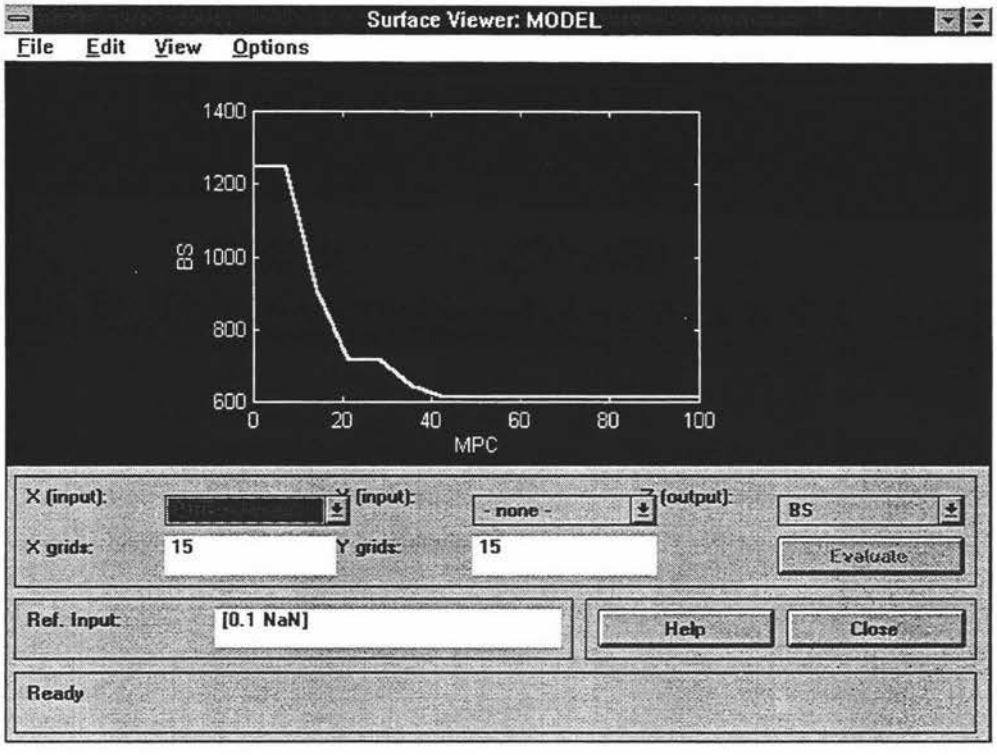


Figure 4.4: Effect of MPC on Buffer Size

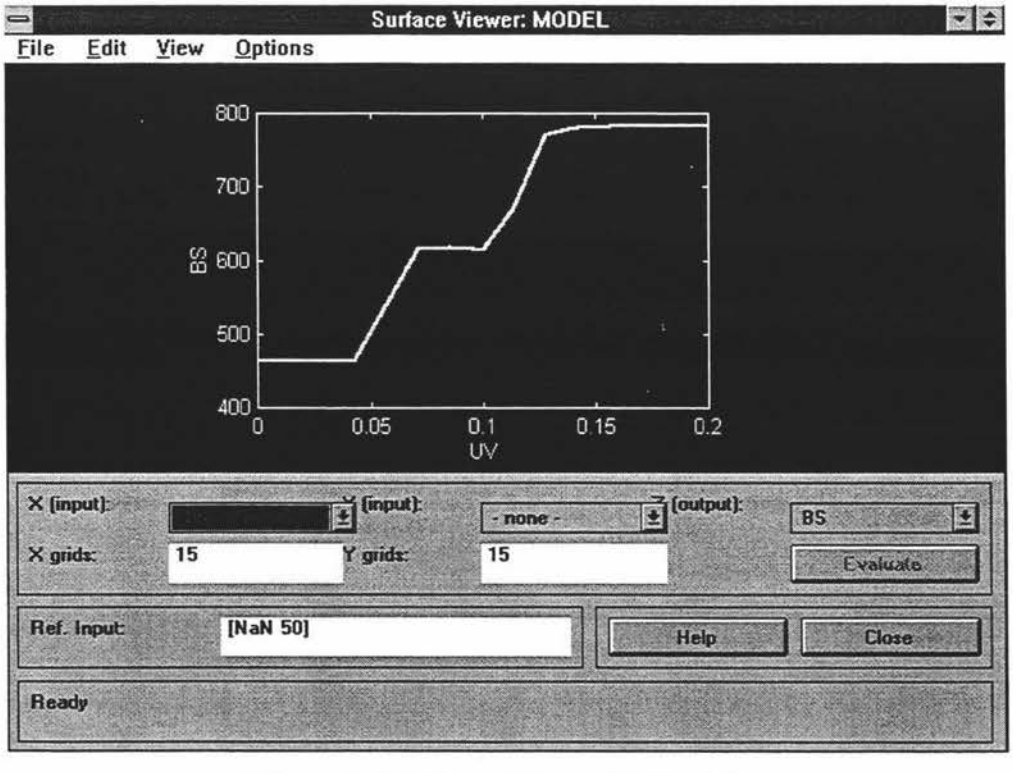


Figure 4.5: Effect of UV on Buffer Size

4.4.2 *Fuzzy Implication and Aggregation*

The standard min. operator proposed by Zadeh (1963) is the basis for inferencing. That is,

$$\mu_{MPC} \text{ and } \mu_{UV} = \min (\mu_{MPC}, \mu_{UV}) \qquad \textbf{(4.6)}$$

To illustrate, consider as a example of when cv = 1 and MPC = 10%. For rule 1:

> If the effect of up-stream variability is *disruptive* and protective capacity is *low* then buffer size is *medium* **0.1**

The relative truth of rule 1 consequence is min. (1, 0.5) or 0.5. This value correlated with the relative truth that the buffer size *medium*. That is, compatibility that buffer size is *medium* is equal to the compatibility that up-stream variability is *disruptive* and protective capacity is *low*. Further, since the researcher has little confidence in the validity of this rule, the relative truth is then reduced by the rules confidence, 0.1, to 0.5x0.1 = 0.05.

As the details of aggregation, the process of combining the fuzzy buffer size set, is well known (see Cox (1994) for an excellent discussion), it is only noted that aggregation is carried out using the sum method. Further, for the purposes of this research, alpha cuts that restrict the operating range of each fuzzy set are set equal to zero.

4.5 Defuzzification

Fuzzy sets are Mandami type, and so a centroid (centre of gravity) approach is used to defuzzify the fuzzy buffer size set into a crisp estimate, expressed in terms of time.

4.6 The Fuzzy Buffer Sizing Model and a Worked Example

Figure 4.6 summarises the essential components of the fuzzy buffer sizing model developed in this chapter.

The real valued inputs representing protective capacity levels and the effect of up-stream variability are fuzzified via membership functions. Fuzzy rules are then applied and the relative truth of whether the buffer size is *small*, *medium*, or *large* is determined. Finally, the buffer size set is defuzzified to produce a real valued output of the appropriate buffer size - expressed in terms of time. This section presents a worked example of illustrating the process of fuzzification, fuzzy inference and defuzzification.

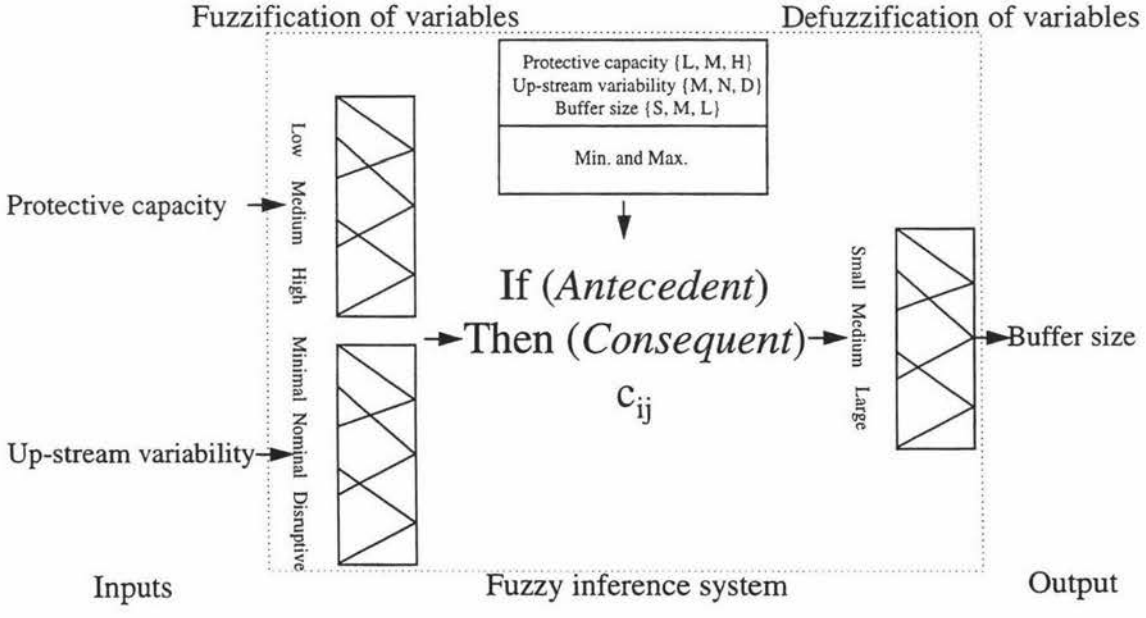


Figure 4.6: The Fuzzy Buffer Sizing Model

Figure 4.7 illustrates the process of fuzzification, implication, aggregation and defuzzification, for the fuzzy buffer sizing model. The membership functions for each 17 rules are presented; real valued inputs of cv = 0.07 and MPC = 30% are fuzzified and correlated with the appropriate buffer size output fuzzy set. The appropriate buffer size

output fuzzy sets are then combined to produce a final output set. The centroid is then calculated and from this a real value estimate of the buffer size is obtained.

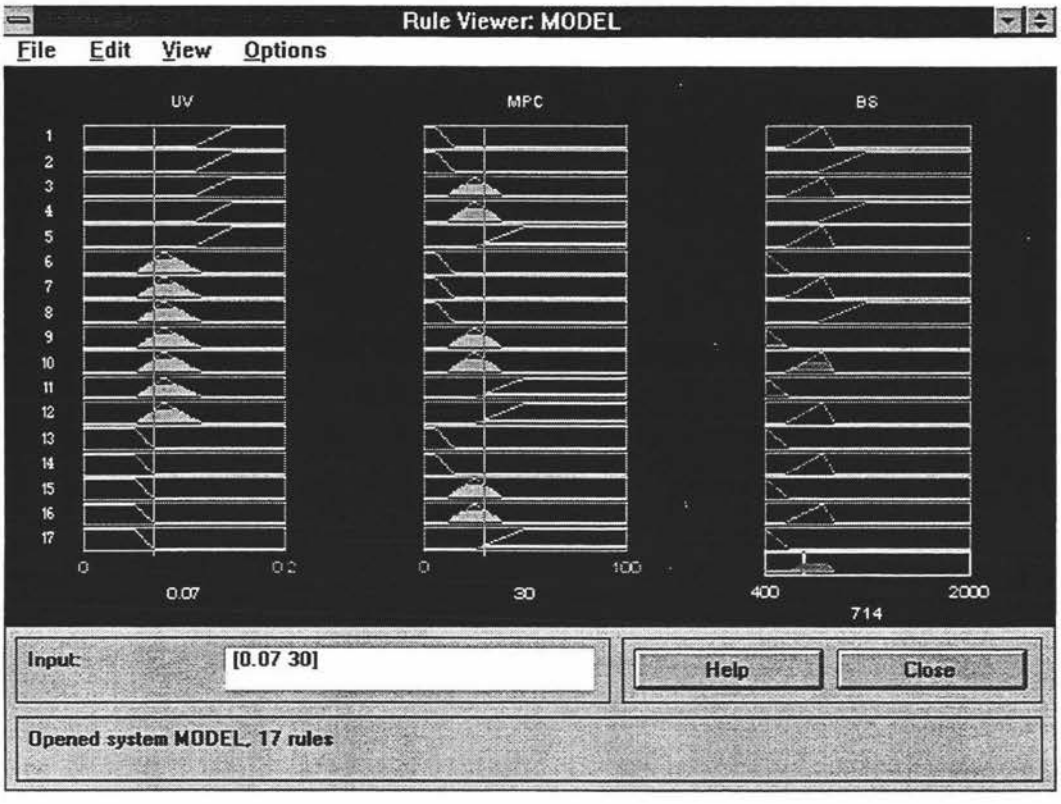


Figure 4.7: Worked Example of Fuzzy Buffer Sizing Solution (MPC = 30% and cv=0.07)

Figure 4.7 illustrates the process used to calculate the fuzzy buffer size estimates used in chapter seven to assess the effectiveness of the fuzzy solution.

Chapter Five

Simulation Methodology

5.1 Introduction

Many of the weaknesses in BMP research models (as discussed in chapter two) arise from the an inability to model manufacturing dynamics caused by the existance of statistical variability in dependent event chains.

Consequently, research models do not reflect the reality of manufacturing and it is of little surprise that the results are of limited practical value. Differing assumptions have meant that to date no standard model has been adopted by researchers, making it difficult to compare proposed solution methods.

This chapter proposes a simulation-based methodology for investigating the research questions suggested in chapter two (discussed further in chapter 8). That is,

- developing heuristic methods for setting initial buffer sizes;
- characterising the relationship between buffer size, variability and protective capacity;
- developing heuristics for judging when and by how much buffer sizes should be changed; and
- developing structured approaches to changing the size of the buffer.

Four methodological issues are examined:

1. appropriateness of a simulation solution approach;
2. the manufacturing model;
3. modelling variability; and
4. buffer management performance measures.

5.2 Simulation as a Solution Approach

Simulation is viewed by many operations research analysts as the method of last resort; appropriate only when analytical and optimisation models fail. Arguably it is exactly for this reason that simulation should be adopted for BMP research. Chapter two argued that manufacturing systems, characterised by complexity and uncertainty, are not amenable to OR techniques. Simulation can model the many complex and often intractable relationships found in manufacturing facilities. Other advantages include the

ability to tightly control experimental conditions (thereby reducing confounding and improving the quality of the results) and the facility to experiment without financial penalty.

Simulation has found much favour with practitioners and researchers, who use it extensively in the decision making process. van der Walde (1991) provides a succinct review of simulation applications including:

- planning of new manufacturing operations;
- process improvement;
- training and education of staff; and
- study of production dynamics.

While simulation may lack in elegance and mathematical sophistication, it is the solution approach of choice by default for the BMP.

5.3 Manufacturing Model

A valid representation of a DBR implementation is important if relevant and effective buffer management techniques are to be developed. Two issues need to be addressed: (1) existence of dependent events; and (2) sources of statistical fluctuations, random events and interactions (collectively known as variability).

This section examines the manufacturing model used to represent the dependency found in a DBR implementation and addresses the need for a standardised model and improved simulation model credibility. Section 5.4 discusses how variability is modelled.

Under DBR the output rate is maintained with the strategic placement of buffer inventories. Buffers are located before the constraint, assembly and shipping areas. A schematic of a DBR implementation is depicted in Figure 5.1.

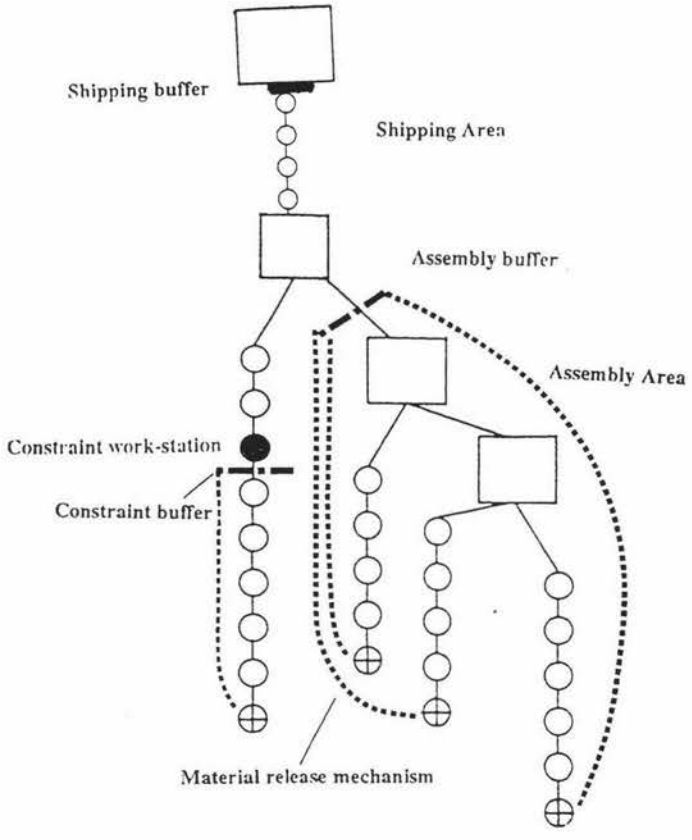


Figure 5.1: A Full DBR Implementation (Goldratt *et al*, 1986).

Buffer locations are a natural way of decomposing a manufacturing facility into a number of individual sections. Work-stations are categorised according to three separate and distinct groups:

1. work-stations that supply work to the constraint area (constraint group);
2. work-stations that supply work to the assembly area (assembly group); and
3. work-stations that supply work to the shipping area (shipping group).

This is a reasonable representation. In a well designed manufacturing facility work-stations are likely to be organised around the flow of work. It is possible, however, that a work-station can belong to any one of the three groups. Further, there is not necessarily only one example of each group type.

Delineating work-stations in this manner permits simplified representation of the full DBR implementation portrayed in Figure 5.1. Figure 5.2 depicts the essential elements of

a simplified, or generic, DBR representation. A single constraint buffer is used to protect the output rate of the constraint work-station.

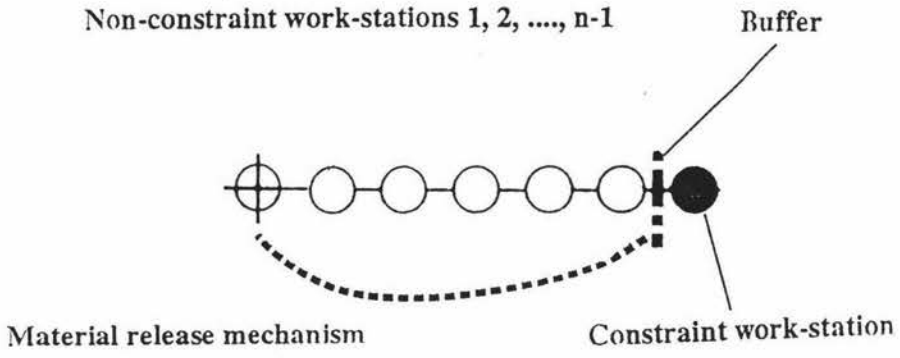


Figure 5.2: A Simplified DBR Implementation

This simplification is important for a number of reasons. Figure 5.2 although simplified, is realistic and still likely to be intractable. This provides the motivation for a fuzzy logic approach. Because of the simplification the fuzzy logic and simulation model is expected to be less complex, require less development time and be easier to verify, adding credibility to the results.

Figure 5.1 shows that the assembly buffer is independently sized from the constraint and shipping buffer. Production disruptions that disrupt the smooth flow of work to the constraint and the shipping buffer have no influence over the arrival of work at the assembly buffer. The assembly buffer is sized according to variability and protective capacity levels in the assembly group.

A similar argument and simple observation also show that the constraint buffer can be sized independently. As before, production disruptions in the assembly and shipping groups have no effect on the arrival of work at the constraint buffer.

Interactions from the constraint and assembly buffer mean that the shipping buffer can not be independently sized. The constraint buffer (a similar argument applies to the assembly buffer) will not completely decouple the shipping group from production

disruptions in the constraint group. Theoretically, complete decoupling is only possible given a buffer of infinite length: clearly, neither a practical or desirable solution.

Production disruptions that cause constraint starvation, and internal production disruptions (such as highly variable constraint processing times) will increase the delivery schedule's dependence on the shipping buffer.

In this case, such a disruption could be viewed as equivalent to a larger than normal production disruption experienced by the first work-station in the shipping group. If this delay caused by the constraint group is significant, it will show as a "hole" (or missing work-order) in the shipping buffer. Its effect can be quantified using an Umble *et al* (1990) style disruption factor which in practice should be adopted as the operational definition of variability by the fuzzy logic model. The shipping buffer can then be appropriately sized to account for the constraint buffer/shipping area interaction. This is analogous to raw material unavailability that may be experienced by the constraint and assembly groups. Instead of using an Umble style "catch-all" measure of variability, constraint and assembly buffer interactions could be modelled as raw material unavailability.

Each group and buffer in Figure 5.1 is equivalent. A generic representation of DBR can be adopted consisting of a single constraint work-station and a constraint buffer. This is the manufacturing model depicted in Figure 5.2.

5.4 Modelling Variability

This thesis has argued that without a realistic model, research results are unlikely to have any practical value. How to realistically representing variability is an important issue in BMP research. This section discusses modelling variability in the context of the BMP.

There are many sources of variability that are responsible for the number of "fires" a production manager must "put out" daily. As mentioned, there is little utility in

individually modelling every source of variability. Here processing time variation is used as an Umble style "catch-all" measure of variability.

Transient queues are implicitly addressed in this research by the simulation model developed in chapter six to test the effectiveness of the fuzzy logic model. A total of five product types compete for six work-stations.

Constraint work-station processing variability is also not considered for the following reasons:

- Variability resulting in an extremely large processing time at the constraint work-station is equivalent to the constraint work-station being blocked, allowing the buffer to build and recover from any previous processing distributions. This will however increase the dependence on the shipping buffer; a detail that does not need to be considered by the manufacturing model.

- Variability resulting in an extremely short processing time will momentarily increase the throughput rate. This research, however, is interested in maintaining a predictable (that is, constant) output rate.

Three important parameters which define (or summarise) a non-constraint processing time distribution are discussed in sections 5.4.1, 5.4.2 and 5.4.3. First, the distribution type. Second, an appropriate measure of central tendency (of which the mean is most commonly used). Third, variability, which characterises the spread of the distribution.

5.4.1 *Distribution Type*

A variety of processing time distributions have been used to simulate job-shops. Law *et al* (1991) lists a number of possible processing time distributions including beta, erlang, exponential, gamma, lognormal, normal, pearson vi, triangular, and truncated normal.

Virtually all job-shop scheduling research has been based on hypothetical models. Little research has been conducted on empirically describing processing distributions found in industry. Processing time distributions, however, should posses a number of common

sense characteristics. Muralidhar *et al* (1992) notes that a processing time distribution should:

- exist only for non-negative values; and
- as processing time variability decreases, the form of the distribution changes from:
    (a) monotonic decreasing, to
    (b) unimodal distributions heavily skewed to the right, to
    (c) normal type distributions, truncated at zero.

A brief review of the literature indicates that a number of processing time distributions used by researchers are inappropriate. For example, the symmetry of the normal distribution is unlikely and its use in BMP research inappropriate. Skewed distributions are common, caused by processing problems such as quality defects that necessitate rework. Use of symmetric distributions in BMP research holds little value. It is the tail of a processing distribution which characterise the production disruptions that endanger the output rate of the constraint work-station.

The exponential distribution, though commonly used, has limited use in BMP research. With the introduction of SPC and TPM programs, few industrial processes follow this distribution (Koulamas, 1993). Further, the exponential distribution is only defined for one level of variability.

The lognormal distribution meets the requirements of Muralidhar *et al* (1992) and has been commonly used (Law *et al*, 1991). Most processing times are concentrated around a central value, although a few extreme observations give rise to the distinct tail that characterises a lognormal distribution (Figure 5.3). However, the use of the lognormal distribution has two drawbacks. When cv is equal to 1, the lognormal distribution ceases to be a monotonic decreasing function. Calculation of the shape and location parameter for a given μ and σ^2 is computationally difficult.

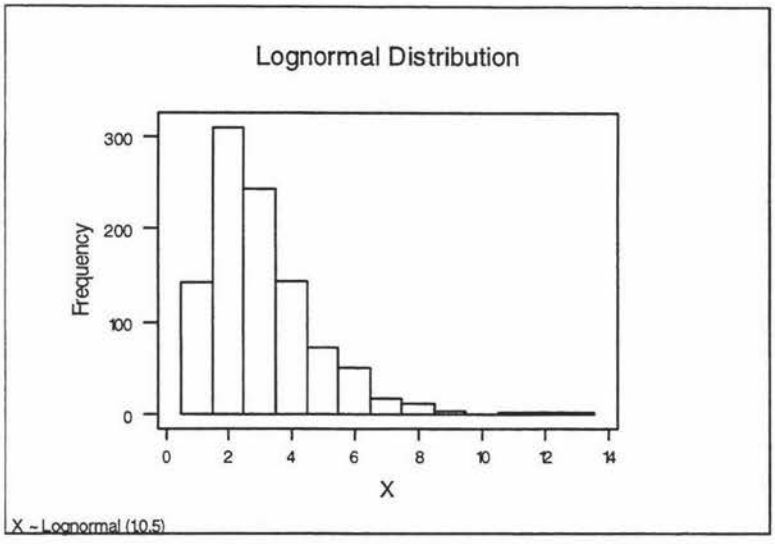


Figure 5.3: A Typical Lognormal Distribution

Muralidhar *et al* (1992) recommends the gamma distribution. Figures 5.4, 5.5 and 5.6 depict typical gamma distributions; Figure 5.4 takes on the shape of a well behaved process while Figure 5.6 models a highly variable and asymmetric process.

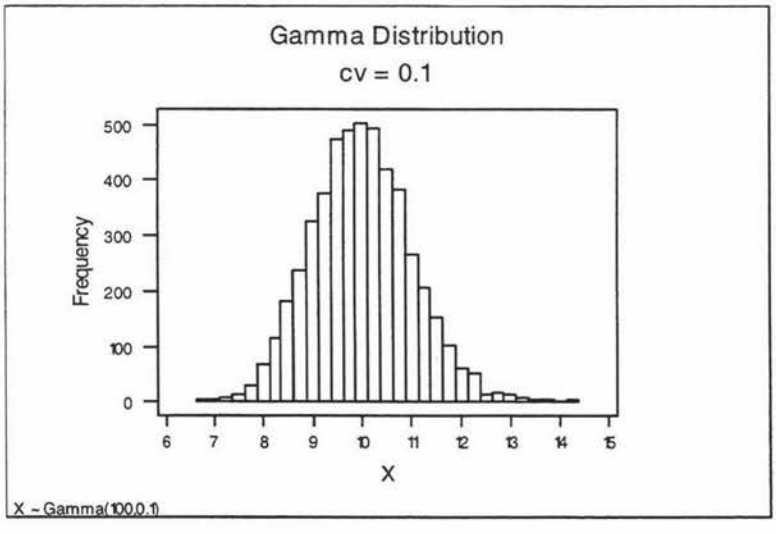


Figure 5.4: Gamma Distribution ($\mu = 10$; $\sigma = 1$)

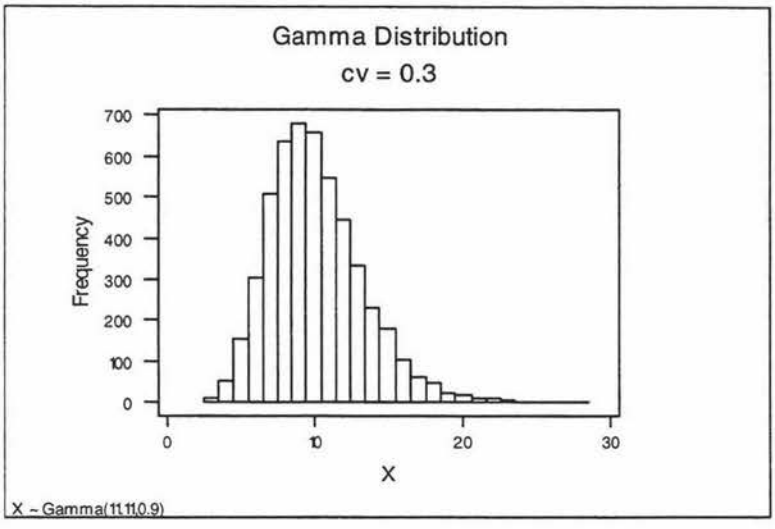


Figure 5.5: Gamma Distribution ($\mu = 10$; $\sigma = 3$)

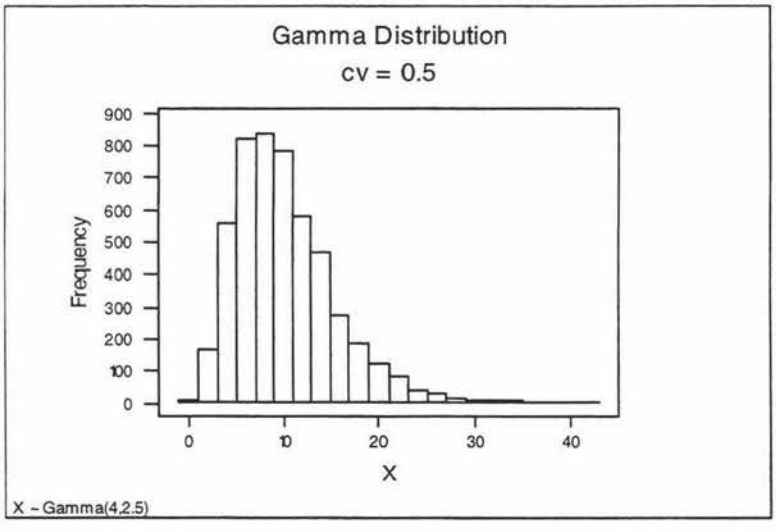


Figure 5.6: Gamma Distribution ($\mu = 10$; $\sigma = 5$)

The mathematical properties of the gamma distribution are listed in Table 5.1.

Property	Formula
Probability Density Function	$f(x) = \frac{x^{\alpha-1}e^{x/\beta}}{\Gamma(\alpha)\beta^{\alpha}}$ **(5.1)**
Mean	$\mu = \alpha\beta$ **(5.2)**
Variance	$\sigma^2 = \alpha\beta^2$ **(5.3)**
Mode	$m = \beta(\alpha-1)$ **(5.4)**

Table 5.1: Mathematical Properties of the Gamma Distribution

Sections 5.4.2 and 5.4.3 examine gamma location and variation issues.

5.4.2 *Central Tendency of Processing Time Distribution*

In a typical DBR implementation, the mean (μ) is used to estimate (characterise or summarise) the expected processing time. This expected or average quantity is used to estimate for product type j (1) the theoretical lead time to the constraint, $\Sigma\mu_{jk}$; and (2) the expected processing time at the constraint work-station, μ_{jn}. Both (1) and (2) are used to determine the material release time and scheduled starting time at the constraint work-station of the i^{th} work-order - two critical scheduling decisions under DBR.

There are two alternative measures that can be used to characterise the average processing time: the mode and the median. A methodological question arises over which average measure is appropriate. In the absence of *a priori* knowledge the Laplace or rationality principle can be used to justify the use of μ. Figures 5.4, 5.5 and 5.6, however, show that the gamma distribution is invariably asymmetric. In this case μ_{jn} and $\Sigma\mu_{jk}$ will overestimate arrival time at the buffer of a typical work-order and hence the release and starting time of the i^{th} scheduled work-order. In effect this is equivalent to increasing the buffer size - with work-orders arriving at the buffer in advance of the expected time. The use of μ is therefore arguably inappropriate.

It can also be argued that the use of μ is still inappropriate over a large number of simulation runs. While μ may be characteristic of the population, the release times and scheduled starting times are calculated for individual work-orders. The mode (or median) is a better estimate as it represents the most probable processing time outcome for a given work-order on a particular work-station.

For a given work-order the mode is arguably a better estimate of the average processing time. In this research, the mode (m) as it represents the most likely outcome is adopted to calculate material release times and scheduled starting times using m_{jn} and Σm_{jk}.

5.4.3 *Processing Time Variation*

Variation is another important aspect of a processing time distribution. Variation can be assessed using two metrics: coefficient of variation (cv) and interquartile range. Researchers have commonly (if not exclusively) used cv to control processing time variation. The cv is defined by equation 5.5:

$$cv = \frac{\sigma}{\mu} \qquad \textbf{(5.5)}$$

The cv is computationally efficient. For a given μ, the variance of the processing time distribution can be easily calculated. As a result the cv also changes the variance in a structured manner by keeping μ constant.

Figures 5.4, 5.5 and 5.6 illustrate the effect of increasing cv for the gamma processing time distribution (cv of 0.1, 0.3 and 0.5 respectively), with μ held constant. As discussed Figure 5.4 is characteristic of a well behaved process where processing times vary slightly around the modal value. Figure 5.6 reflects a highly variable process characterised by a number of processing problems resulting in a skewed distribution.

Intuitively, cv seems to accurately model processing time variability. However, closer examination of Figures 5.4, 5.5 and 5.6 reveals two discrepancies[1].

First, the mode which is used to estimate the material release time and scheduled starting time at the constraint work-station, varies with cv. In particular, as cv increases, the mode decreases; conversely, as cv decreases, the mode increases. This relationship, described by equation 5.6, is depicted in Figure 5.7 for a constant μ of 1 minute.

$$\text{mode} = \mu(1 - cv^2) \qquad \textbf{(5.6)}$$

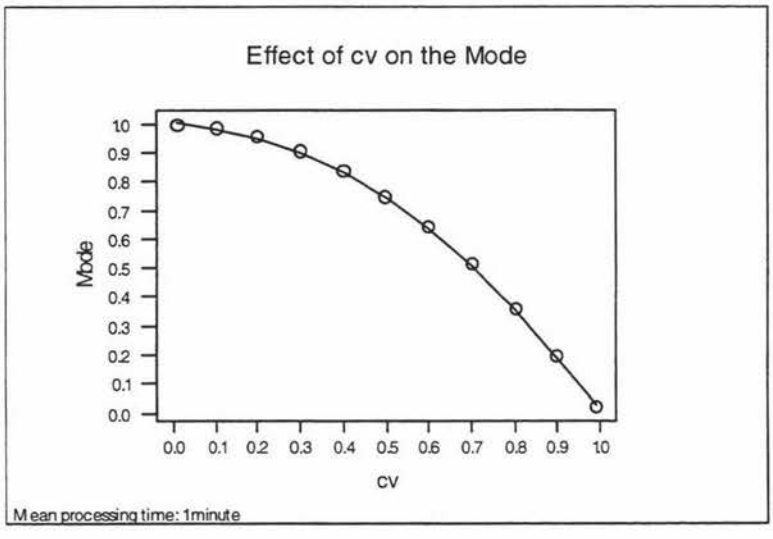


Figure 5.7: Effect of Changing cv on the Mode of the Gamma Distribution

Second, the lower bound of the gamma distribution also decreases as cv increases. This behaviour does not model realistic processing time distributions which describe the time needed to complete a physical task such as welding, cutting or drilling. The lower bound of a processing time distribution represents a case where the operation is free from processing problems. This should not vary with cv. The lower bound should only change if an improvement is made to the process. For example, an improved jig that reduces welding times.

[1] Figures 5.4, 5.5 and 5.6 also demonstrates the inappropriateness of using μ as cv is varied; when cv is small, $\mu \approx$ mode, however, when cv is large, $\mu \neq$ mode.

As cv is varied, both the mode and lower bound vary. As a result, simulation results may be confounded. An increase in throughput for example may be attributed to either an increase in variability or a decrease in the modal processing time.

It is desirable that the mode and lower bound be held constant as cv is varied. This adds another distributional requirement for a processing time distribution. In general, for any asymmetric distribution defined by a shape and scale parameter, this requirement can not be met. A third parameter (or "extra degree of freedom") is needed. Law *et al* (1991) lists two asymmetric distributions that are defined by three parameters: the triangular and pearson vi distribution. In both cases the mode and the lower bound can be held constant while cv is varied. Both distributions meet Muralidhar *et al* (1992) requirements. However, since the triangular distribution is algebraically simple and also has a similar shape, it is used in order to facilitate the development of a realistic processing time distribution.

5.4.4 *Proposed Processing Time Distribution*

The triangular distribution is defined by three points: mode (c), minimum (a) and maximum (b) value. As the triangular distribution is mathematically simple it is often used in the absence of data (Law *et al*, 1995). Table 5.2 details the distributional properties of the triangular distribution.

Property	Formula
Probability Density Function	$f(x) = \begin{cases} \frac{2(x-a)}{(b-a)(c-a)} & if\ a \le x \le c \\ \frac{2(b-x)}{(b-a)(b-c)} & if\ c < x \le b \end{cases}$ **(5.7)**
Mean	$\mu = \frac{a + b + c}{3}$ **(5.8)**
Variance	$\sigma^2 = \frac{a^2 + b^2 + c^2 - ab - ac - bc}{18}$ **(5.9)**
cv	$\frac{\sqrt{a^2 + b^2 + c^2 - ab - ac - bc}}{\sqrt{2}(a + b + c)}$ **(5.10)**
Mode	c **(5.11)**

Table 5.2: Mathematical Properties of the Triangular Distribution

Section 5.4.4 specified that if the mode and the lower bound were held constant and the upper bound was varied to achieve a given level of cv, a more realistic representation of a processing time distribution can be achieved. This section outlines the mathematical development of such a model using the triangular distribution.

First, assume that the modal processing time, c, is known.

The lower bound, a, can be calculated for an arbitrarily level of cv. Substituting equation 5.5 for σ^2 in equation 5.9 yields equation 5.12.

$$(cv.\mu)^2 = \frac{a^2 + b^2 + c^2 - ab - ac - bc}{18} \quad \textbf{(5.12)}$$

Equation 5.8 can be solved for c.

$$c = 3\mu - a - b \quad \textbf{(5.13)}$$

With three unknowns (a, b, μ) and two equations, a and b can not be determined. However, μ can be approximated using equation 5.6 and is denoted as μ_o.

Rearranging equation 5.13 for equation 5.14 demonstrates how c (and μ_o) can be held constant while cv is varied.

$$a + b = 3\mu_o - c \tag{5.14}$$

Values of a and b can be varied provided that equation 5.14 holds. If a increases, b decreases. cv also decreases (equation 5.10). Similarly, if a decreases, both b and cv increase.

Solving equations 5.12 and 5.13 for a results in equation 5.16.

First, let $k = 3\mu_o - c$ **(5.15)**

$$3a + k_1 a + k_2 = 0 \tag{5.16}$$

where:

$$k_1 = -3k \tag{5.17}$$

$$k_2 = 3k^2 - 9k\mu_o + 9\mu_o^2 - 18\sigma^2 \tag{5.18}$$

$$\sigma^2 = (\mu_o cv)^2 \tag{5.19}$$

Equation 5.16 can be solved using the quadratic formula,

$$\frac{-k_1 \pm \sqrt{k_1 - 12k_2}}{6} \tag{5.20}$$

Where b is the first root and a is the second root (though only a needs to be calculated and b solved using equation 5.23). That is,

$$a = \frac{-k_1 - \sqrt{k_1 - 12k_2}}{6} \quad \textbf{(5.21)}$$

and

$$b = \frac{-k_1 + \sqrt{k_1 - 12k_2}}{6} \quad \textbf{(5.22)}$$

The value of a is adopted as the lower bound for all processing distributions. b is then varied until cv (or σ^2) is achieved. Given a and c, b can be calculated from equation 5.9. Solving equation 5.9 for b yields equation 5.23.

$$b = \frac{(a+c) + \sqrt{(a+c)^2 - 4(a^2 + c^2 - ac - 18\sigma^2)}}{2} \quad \textbf{(5.23)}$$

where σ^2 is calculated from equation 5.18.

As cv changes, μ changes (see equation 5.8). Variation about the mean will no longer be constant. In this case, cv will no longer be defined by equation 5.5; rather cv is defined by equation 5.24.

$$cv = \frac{\sigma}{\mu_o} \quad \textbf{(5.24)}$$

The cv is interpreted as the (approximate) variation around the modal value (as when cv is small, $\mu \approx m$ since the processing time distribution is symmetric). This is a structured approach to varying σ^2 and in keeping with the arguments presented in section 5.4.2.

The processing time distribution can be completely specified for a fixed lower bound and constant mode. In this case, the behaviour of the processing distribution is solely determined by cv. Figure 5.8 summarises:

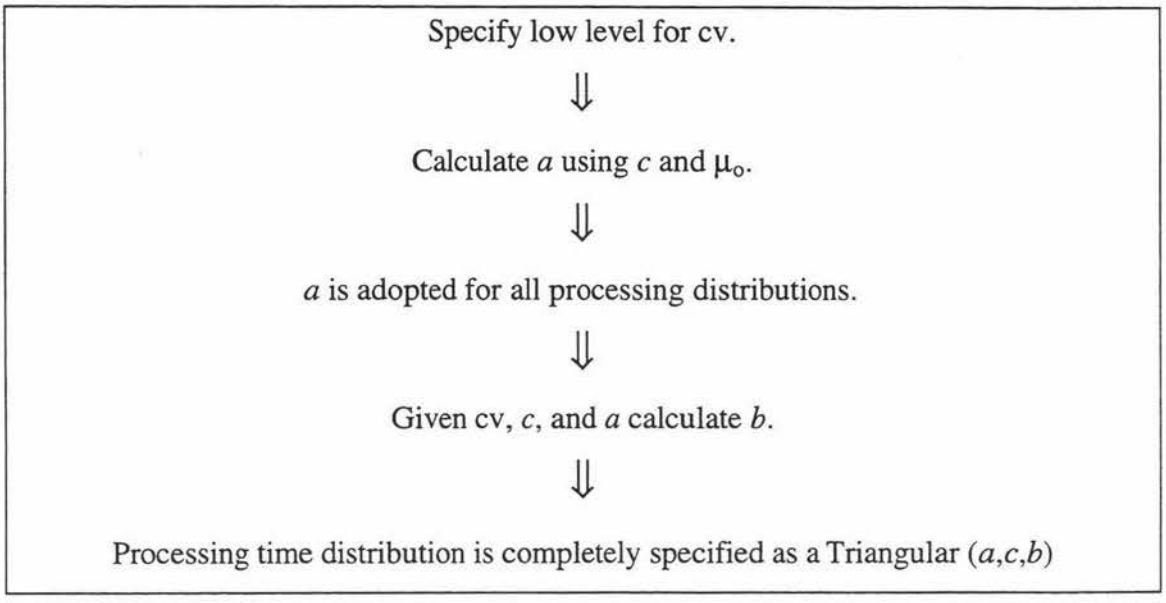


Figure 5.8: Specifying Processing Time Distributions

5.4.5 *Example Processing Time Distributions*

Consider two processes characterised by two levels of variability: cv of 0.1 and cv of 0.2. Each process has a modal processing time of 10 minutes. The triangular probability distribution is used to describe the processing time distributions.

Following the procedure outlined in section 5.4.5 and Figure 5.8 two processing time distributions are specified (see Appendix A for details).

1. triangular (7.68, 10, 12.62); and
2. triangular (7.68, 10, 17.17).

Figures 5.9 and 5.10 depicts the processing time distributions.

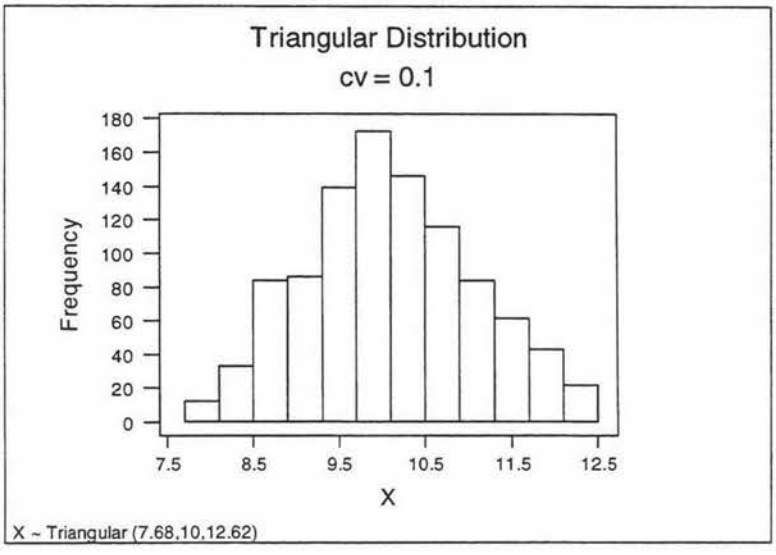


Figure 5.9: Triangular (7.68, 10, 12.62) Distribution

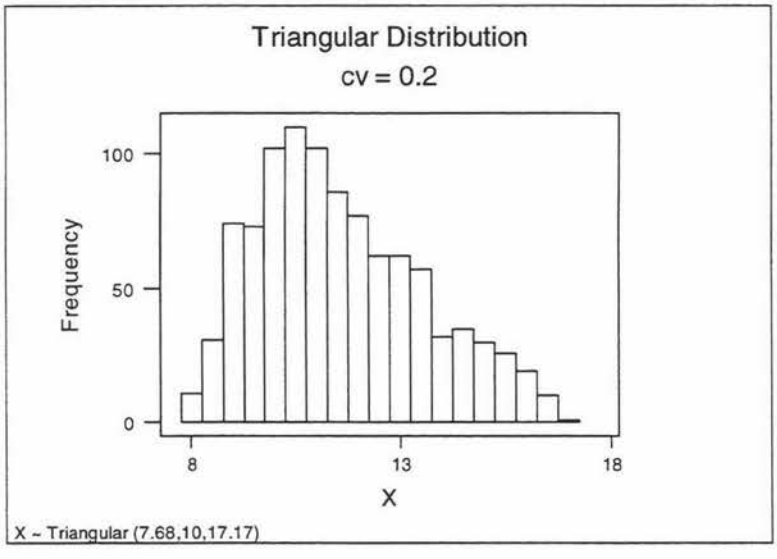


Figure 5.10: Triangular (7.68, 10, 17.17) Distribution

Note how Figures 5.9 and 5.10 do not look too dissimilar from other more complex distributions such as the gamma or pearson vi distribution.

5.5 Comparison of Buffer Sizing Techniques

The final issue addressed by this chapter is buffer management performance measures.

Production disruptions cause work-flow to become uneven and uncertain. As a result work-orders scheduled to arrive at the constraint work-station will be delayed - threatening the output rate of the constraint work-station.

An appropriately sized buffer will minimise the manufacturing lead time and maximise the percentage of on-time deliveries thus enhancing the time-based competitiveness of a manufacturing facility. The question of what is an appropriate buffer size needs to be addressed. One widely adopted approach is the buffer should be sufficiently large enough to protect the output rate of the constraint work-station from 95% of production disruptions, that is, the normal variation found in manufacturing facilities. There is little value in protecting the output of the constraint from "catastrophic" outcomes such as work-station failure. These disruptions, which represent the top 5% of production disruptions, should be solved using TQM and TPM methodologies.

The delay time of the i^{th} work-order is defined by:

$$D_i = \text{Actual lead time to the buffer - Theoretical lead time to the buffer} \qquad \textbf{(5.25)}$$

where the theoretical lead time to the buffer is estimated from the sum of the modal processing times, Σm_{jk} for the j^{th} product type on the k^{th} work-station.

D_i is a function of protective capacity and the up-stream variability. The delay time distribution can be used to estimate the appropriate buffer size. Let $\{D_1, D_2, ..., D_n\}$ be a sequence of n delay times (Figure 5.11).

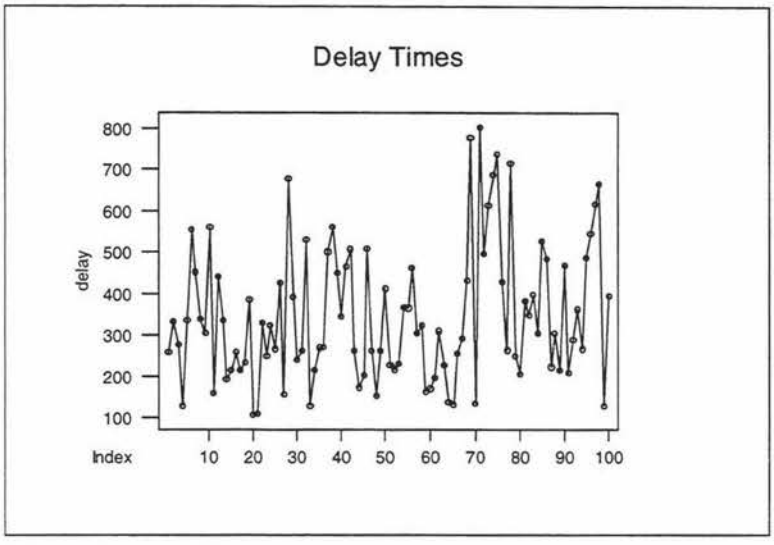


Figure 5.11: Time Series Plot of 100 Delay Times

If the $\{D_i$'s$\}$ are ranked in ascending order, the j^{th} delay time quartile can be estimated from the empirical cumulative density function, $\hat{F}(d_j)$, given by equation 5.26.

$$\hat{F}(d_j) = \frac{j - 1}{n} \qquad \textbf{(5.26)}$$

For large n, the 95% delay time quartile can be estimated from equation 5.26 (as $\hat{F}(d_j) \rightarrow F(d_j)$ as $n \rightarrow \infty$).

The distribution of delay times provides an alternative way of interpreting the meaning of an appropriate buffer size. If the buffer size is equal to the p^{th} delay time quartile ($D_{p\%}$), the probability that a given work-order will arrive at the buffer before its scheduled starting time at the constraint work-station is p. Note that $D_{0\%}$ is defined as 0.

An appropriate buffer size will release a work-order into the manufacturing facility early enough to overcome any disruption that may threaten the constraint's schedule. "Early enough" may mean that the appropriate buffer size is equivalent to the 95% delay time quartile. This will ensure with 95% probability that the work-orders will be processed at the constraint work-station in the planned order. For many manufacturers, a buffer, $D_{95\%}$

long has little utility since in most cases the role of the buffer is to maintain the output rate and not keep the planned order of processing at the constraint work-station. In most instances, a buffer with 900 minutes of work is just as effective as a buffer with 800 minutes of work. A buffer of $D_{95\%}$ provides a pessimistic bound on the appropriate buffer size for a manufacturing facility.

The identification of an appropriate buffer size is then reduced to selecting p such that there is sufficient work in the buffer to keep the constraint work-station working until the planned work-order arrives. Exactly what is meant by "sufficient work" is a management decision, that is unique to a given manufacturing facility, but will be some portion of the pessimistic buffer size.

For convenience, the contents of the buffer are expressed in terms of quantity rather than time since the delay time distribution (if set equal to the buffer size) quantifies the probability that a work-order will arrival at the buffer later than the scheduled starting time. Protection, however, is still defined in terms of time by the delay time quartile.

Pessimistic buffer size is related to work-order quantity via equation 5.27.

$$N_{Pessimistic} = \frac{\text{Pessimistic Buffer Size}}{\overline{m}_n} \qquad \textbf{(5.27)}$$

where $\overline{m}_n$ is the "average" modal processing time at the constraint work-station and the pessimistic buffer size is estimated by the 95% delay time quartile, $D_{95\%}$.

The buffer sizing problem can be expressed more formally as, given a pessimistic buffer size of $N_{Pessimistic}$ work-orders (N), select p such that there is at least x work-orders are in the buffer with probability $(1-\beta)$. This problem can be solved using the binomial probability distribution. That is, select p such that

$$1 - \sum_{i=0}^{x} \binom{N}{i} p^i (1-p)^{N-i} = 1 - \beta \qquad \textbf{(5.28)}$$

where β is the probability that work-orders will have to be expedited to ensure that there are at least x work-orders in the buffer.

Example Calculation

A manufacturer of a single product needs a buffer of 1000 minutes to ensure that work-orders will be processed in the correct order according to the constraint's schedule. The modal processing time at the constraint work-station is 100 minutes. Management requires that a buffer of 800 minutes work (or 8 "typical" work-orders) should be maintained in front of the constraint work-station. However, they only wish to expedite 5% of the time to ensure that this level is maintained.

From equation 5.27, $N_{Pessimistic} = N = 10$. Further, $\beta = 0.05$ and $x = 8$. Equation 5.28 is solved for p using cumulative binomial tables (for example, see Mendenhall *et al*, 1982). In this instance, $p = 0.61$.

That is, to ensure that 8 typical work-orders (or 800 minutes of work) are processed by the constraint in the correct order (with 95% probability), the probability of arrival for work-order before its scheduled starting time needs to be at least 0.61. The appropriate buffer size for the manufacturers requirement is $D_{0.61}$.

Equation 5.28 is valid provided the:

1. work-order arrival at the constraint is independent; and
2. probability of work-order arrival (p) at the constraint is constant.

which are stylised assumptions and cannot be met due to the existance of dependent events, such that the probability of arrival will vary from work-order to work-order. In this case, equation 5.27 produces a lower bound estimate of the appropriate buffer size.

Section 5.5.1 presents a performance metric, based on the population quartile of the delay time distribution and a technique used to calculate this quartile. Section 5.5.2 argues that the buffer sizing effectiveness should be assessed according to practical, rather than statistical, significance.

5.5.1 *Buffer Effectiveness and the Appropriate Buffer Size*

An appropriate buffer size is defined as the p^{th} population quartile of the delay distribution calculated from equation 5.28. That is, the appropriate buffer size is equal to D_p.

The following measure of buffer effectiveness is proposed:

$$\text{Buffer Effectiveness} = | \text{Estimated Buffer Size} - \text{Appropriate Buffer Size} | \quad \textbf{(5.29)}$$

The symmetry of equation 5.29 means that it is a simple measure of buffer effectiveness. That is, use of the absolute value implies that the "cost" of undersizing a buffer is equally as costly as oversizing a buffer. In practice this is not likely to be the case; however, as the economic trade-off between output protection and the length of the manufacturing lead time is unknown, equation 5.29 is valid for the purposes of this research. For ease of comparison equation 5.29 can also be expressed in terms of percentage error.

A number of well established results in the order statistics literature can be used to infer the likely value for the p^{th} delay time quartile using $\hat{F}(d_j)$ described by equation 5.26.

Arnold *et al* (1992) derives a distribution free confidence interval for population quartiles based on a random sample of order statistics. This research applies Arnold's technique to the estimation of the confidence interval surrounding the p^{th} population quartile of the delay time distribution. A less mathematical treatment of the procedure can be found in Mosteller *et al* (1973).

Adopting the technique described by Arnold *et al* (1992), let:

p be the population quartile (determined by equation 5.28);
$D_{i:n}$ be the i^{th} order statistic from a random sample of n delay times;
F(p) be the population cumulative density function (CDF);
$[D_{i:n}, D_{j:n}]$ be the interval containing p; and
$\alpha(i,j)$ be the actual confidence level

Assuming an absolutely continuous CDF (that is, $F(F^{-1}(p)) = p$), D_i and D_j are chosen such that:

$$\alpha(i,j) = P(D_{i:n} \leq F^{-1}(p) \leq D_{j:n}) = \sum_{r=i}^{j-1} \binom{n}{r} p^r (1-p)^{n-r} \qquad \textbf{(5.30)}$$

As $\alpha(i,j)$ is a step function (i and j are integer), the interval $[D_{i:n}\ D_{j:n}]$ will be conservative. In practice, $\alpha(i,j)$ can be estimated from binomial probability tables that are found in most elementary statistics books. Even so, calculation of $\alpha(i,j)$ is tedious. If n is sufficiently large (n>30; Freund (1988)) the normal distribution can be used to approximate equation 5.30. That is,

$$\alpha(i,j) = \Phi\left(\frac{j - 0.5 - np}{\sqrt{np(1-p)}}\right) - \Phi\left(\frac{i - 0.5 - np}{\sqrt{np(1-p)}}\right) \qquad \textbf{(5.31)}$$

where $\Phi(\cdot)$ is the standard normal CDF.

For a desired confidence level, α_o, i and j can be calculated using equations 5.32 and 5.33.

$$i = [np + 0.5 - \Phi^{-1}\left(\frac{1+\alpha_o}{2}\right)\sqrt{np(1-p)}] \qquad \textbf{(5.32)}$$

$$j = [np + 0.5 + \Phi^{-1}\left(\frac{1+\alpha_o}{2}\right)\sqrt{np(1-p)}] \qquad \textbf{(5.33)}$$

The interval $[D_{i:n}, D_{j:n}]$ will contain the p^{th} population quartile with probability $\alpha(i,j)$.

Example Calculation

30 delay times from a Normal distribution with $\mu=10$ and $\sigma^2=1$ are generated using Minitab v.10 (Minitab, 1994) and plotted on a frequency histogram in Figure 5.12:

7.66, 8.78, 8.82, 8.86, 8.92, 9.13, 9.43, 9.62, 9.64, 9.67, 9.69, 9.89, 9.96, 9.97, 10.15, 10.26, 10.33, 10.32, 10.37, 10.46, 10.75, 10.95, 11.09, 11.12, 11.22, 11.26, 11.46, 11.60, 11.67, 12.36

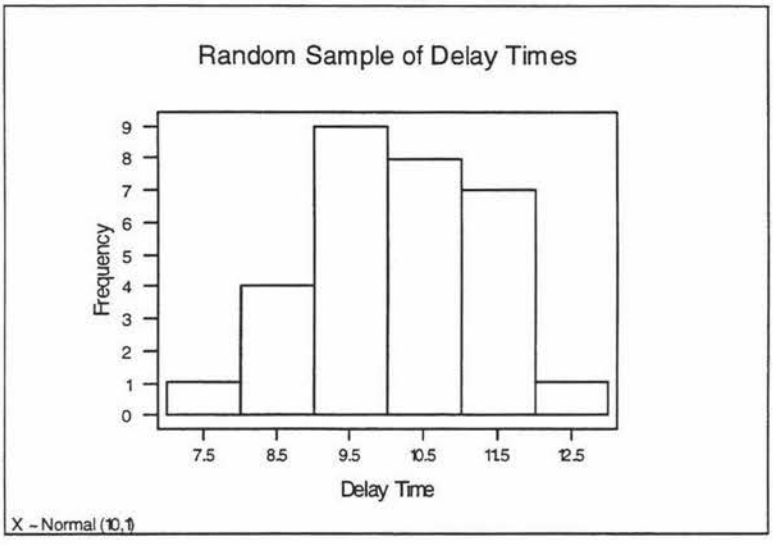


Figure 5.12: Histogram of Randomly Sampled Delay Times

For this example, α_o=0.95, p=0.95 and n=30. The values for i and j are calculated (using equations 5.32 and 5.33) as 28 and 30 respectively. The 95% population quartile is calculated as: 10 + 1.96x1 = 11.96.

That is, the 95% population delay time quartile is contained in the interval [11.60, 12.36] with 0.95 probability which includes the population 95% quartile of 11.96.

The usefulness of the $D_{p\%}$ order statistic technique is limited to simulation based experiments as in practice it is likely that the delay time distribution in a manufacturing facility is unlikely to reach steady state.

5.5.2 *Practical vs Statistical Significance*

Under DBR, if the delay time distribution is known, the appropriate buffer size can be calculated using order statistics. While to many practitioners such a result may be

esoteric, it does allow researchers to assess the effectiveness of buffer sizing techniques. In keeping with the arguments presented in chapter two, an absolute measure of effectiveness (as in equation 5.29) is insufficient. The performance of a proposed buffer sizing technique should be compared against an established "quick and dirty" technique.

Hess *et al* (1980) define quick and dirty as:

> "The whole philosophy behind "Quick and Dirty" management science is to present a simple way to attack complex problems." (pp. 1)

A buffer sizing technique that meets Hess's definition is Umble *et al* (1990) heuristic:

> "We have found by experience for a firm trying to implement a DBR, a *convenient* starting point for the total time buffer is approximately one-half of the firm's current manufacturing lead time." (pp. 145)

The added effort associated with a sophisticated buffer sizing technique should be outweighed by an increase in the relative performance. A comparison with a "quick and dirty" technique is important if buffer sizing heuristics are to be developed for industrial use. A proposed buffer sizing technique will have merit if a significant difference in buffer effectiveness can be demonstrated between it and an Umble type heuristic. Further, this difference should not only be statistically significant but also practically significant. An excellent discussion of practical significance is given in Wang (1993).

What constitutes practical significance for the BMP is an open research question. It will however, vary between manufacturing facilities but is dependent on the competitive "cost" associated with a larger than necessary lead time and the perceived utility of the buffer sizing technique. While more research needs to be conducted, practical significance should be a driving force for BMP researchers. The delay time distribution may enable the effectiveness of a buffer sizing technique to be gauged but it gives the researcher no indication of the technique's utility - arguably the ultimate objective function of manufacturing systems research.

To address this problem, this research proposes to compare a number of performance indicators to assess the utility of a fuzzy logic buffer sizing solution. They are the mean cycle time and mean constraint utilisation.

Chapter Six

Simulation Model Implementation

6.1 Introduction

Chapter five argued that simulation was the method of choice for testing both buffer management methodologies and research hypothesises. This chapter details the research hypothesis, the simulation model parameters, the model implemented and the methodological issues unique to simulation modelling. The chapter concludes by detailing the experimental design used in this research.

6.2 Fuzzy Logic Research Hypothesis

This research aims to develop an effective yet implementable buffer sizing technique. The fuzzy logic model developed in chapter four is one possible solution. A fuzzy logic model is expected to size buffers more effectively than Umble's heuristic since it explicitly considers those factors that influence buffer size. In order words, a fuzzy logic solution is expected to minimise BE - the adopted measure of buffer effectiveness discussed in section 5.5.1. More formally, a one-tailed research hypothesis is proposed:

$$H_O: BE_{(Fuzzy\ logic)} \geq BE_{(Umble)}$$
$$H_A: BE_{(Fuzzy\ logic)} < BE_{(Umble)}$$

Practical significance is assessed by examining the mean cycle time and constraint utilisation.

6.3 Model Specification

This research adopts the manufacturing model proposed in section 5.3. Variable processing times are modelled using the methodology detailed in section 5.4.5. Sections 6.3.1 to 6.3.3 specify the many simulation model parameters and assumptions used in this research.

6.3.1 *Product Type and Work-Order Generation*

A product type is characterised by two processing requirements:

- routing information; and
- modal processing times.

For each product type, routings are generated using a discrete uniform or integer distribution (without replacement) using Minitab v.10 (Minitab, 1994). Routings, however, are subject to two restrictions. Routings must be sequential and in accordance with the manufacturing model include the sixth work-station. Generation of modal processing times is discussed in detail in section 6.3.2.

A work-order consists of a batch size of a single product type that is processed by a number of work-stations. As before, the batch size and product type of each work-order is generated from an integer distribution.

Distributional details for product type and work-order generation is given in Table 6.1.

Simulation Entity	**Simulation Parameter**	**Probability Distribution**
Product type	Number of work-stations visited (V) Routing	Integer [1,5] Integer [1,5]
Work-order	Product type Batch size	Integer [1,5] Integer [10,20]

Table 6.1: Input distributions

Associated with each work-order is a scheduled starting time at the constraint (STC) and a material release time (MRT). The STC is calculated using equation 6.1. As work-orders are sequenced by due date, equation 6.1 maximises the output rate (and due date performance). The MRT is calculated using equation 6.2 which releases the i^{th} scheduled work-order a buffer time early, ensuring that it reaches the constraint work-station before work is scheduled to begin. Nomenclature is defined in Table 6.2.

$$STC_i = STC_{i-1} + m_{jn} b_i \qquad \textbf{(6.1)}$$

$$MRT_i = STC_i - \sum_{k}^{v} m_{jk} - BS \qquad (6.2)$$

Variable	Meaning
STC_i	Scheduled starting time of the i^{th} work-order at the constraint work-station
MRT_i	Material release time of the i^{th} work-order at the gate-way work-station
m_{jk}	Modal processing time of the k^{th} work-station for the j^{th} product type
m_{jn}	Modal processing time of constraint work-station for the j^{th} product type
Σm_{jk}	Total expected processing time to the constraint work-station
b_i	Batch size of the i^{th} work-order
BS	Buffer size

Table 6.2: Nomenclature

A manufacturing facility using MRP scheduling heuristics is used to calculate buffer size using Umble's heuristic. The material release policy is based on the utilisation of the gateway work-stations: if either gate-way work-station is idle a work-order is released. To limit the total amount of WIP in the manufacturing facility, queue lengths between the work-stations were limited to 10 work-orders.

The scheduling information is stored in the final assembly schedule. Table 6.3 presents an example final assembly schedule for DBR. Work-orders are ordered by delivery due date. Note that although Job # 26 is scheduled to begin at the constraint work-station before Job # 27, it is released later because it is processed by fewer work-stations hence has a shorter lead time to the constraint work-station.

Job #	Product type	Batch size	Starting time at constraint work-station	Release time at gate-way work-station
25	3	18	500	300
26	2	12	650	450
27	4	20	730	420

Table 6.3: Example Final Assembly Schedule

6.3.2 *Modal Processing Times*

Let m_{jk} be the modal processing time (per unit) at the k^{th} work-station for a work-order of product type j. Processing times are expressed in terms of minutes per part. For the purposes of this research, m_{jk} does not include setup times. Work-station setups are assumed to be sequence independent. Further, as variability is measure in a generic way, inclusion of setup times adds nothing when assessing the performance of the fuzzy logic buffer sizing model.

The modal processing time per unit at the sixth work-station (the constraint), m_{jn}, is generated using an I[5,10] distribution. The non-constraint modal processing times, however, are calculated from the manufacturing facility's mean protective capacity (MPC) level. The operational definition of protective capacity (Atwater, 1991) is given in equation 6.3.

For a work-order of product type j, visiting V work-stations, MPC is equal to:

$$MPC_j = 100\left(\frac{\sum_{k}^{V}(1 - \frac{m_{jk}}{m_{jn}})}{V}\right) \qquad \textbf{(6.3)}$$

where i is the index of the k^{th} work-station visited by a work-order of product type j and m_{jn} is the modal processing time of the j^{th} product type at the constraint work-station.

Mean protective capacity is expressed as the difference between the non-constraint and constraint capacity. As an illustration, consider a constraint work-station can process a work-order every 20 minutes. In this case, if a non-constraint work-station can supply the constraint work-station with a work-order every 10 minutes, it is said to have a protective capacity of 50%.

Equation 6.3 can be rewritten as:

$$MPC_j = 100(1 - \frac{\overline{m}_j}{m_{jn}}) \qquad \textbf{(6.4)}$$

where $\overline{m}_j$ is the mean modal processing time over V work-stations visited by a work-order of product type j.

Given MPC_j and m_{jn}, $\overline{m}_j$ can be solved for:

$$\overline{m}_j = m_{jn}(1 - \frac{MPC_j}{100}) \qquad \textbf{(6.5)}$$

If m_{jk} is set equal to $\overline{m}_j$ the manufacturing model is said to be balanced. Most manufacturing facilities, however, are unbalanced and $m_{jk} \neq \overline{m}_j$. This research simulates an unbalanced manufacturing facility. Non-constraint capacity, m_{jk}, is allocated using $\overline{m}_j$ via the triangular distribution defined in equation 6.6.

$$m_{jk} \sim \text{Triangular}\,(0.9\,\overline{m}_j,\ \overline{m}_j,\ 1.2\,\overline{m}_j) \qquad \textbf{(6.6)}$$

where the lower and upper bound that defines a triangular distribution have been set to achieve the desired variation in modal processing times.

Product routings and modal processing times of each product type for an MPC of 10% is given in Table 6.4. Due to sampling error the actual MPC is 10.3%.

Product type	Workstation	Modal processing time	Protective Capacity (%)	Mean Protective Capacity (%)
1	1	4.60	8.0	10.2
	4	4.27	14.6	
	5	4.60	8.0	
	6	5.00	0.0	
2	2	8.64	13.6	12.1
	3	9.53	4.7	
	4	8.26	17.4	
	5	8.75	12.5	
	6	10.00	0.0	
3	1	6.31	21.1	15.5
	2	7.30	8.8	
	4	6.68	16.5	
	6	8.00	0.0	
4	2	6.56	6.3	7.8
	3	6.35	9.3	
	6	7.00	0.0	
5	1	7.78	13.6	6.9
	2	8.59	4.6	
	3	9.51	-5.7	
	4	7.33	18.6	
	5	8.71	3.2	
	6	9.00	0.0	Overall: 10.3%

Table 6.4: Routing and Processing Information By Product type

The modal processing times are used in conjunction with the proposed triangular processing time distribution to generate processing times for a given work-order.

Equation 6.6 allocates protective capacity randomly around $\overline{m}_j$. How protective capacity should be allocated, however, is an open research question. Atwater's (1991) research

indicates that a flat arrangement of protective capacity rather than a decreasing or random arrangement of protective capacity is more likely to maintain the output rate. Allocation of protective capacity is another possible input variable for the fuzzy logic model.

The operational definition of protective capacity has a number of drawbacks. Primarily, equation 6.6 fails to differentiate between *excess* and *protective capacity*. Other alternative protective capacity metrics similarly suffer. Equation 6.6 was adopted as it was simple to implement and has an intuitive appeal. It is also, further motivation for a fuzzy logic approach to buffer sizing. The development of a protective capacity metric is a rich area for future research.

6.3.3 *Manufacturing Model Assumptions*

A number of assumptions have been made to facilitate the development of the simulation model used in this research.

Simulation model assumptions include:

- No quality defects;
- No work-station failure;
- No pre-emption;
- Zero travel time between work-stations;
- No alternative routing of work-orders;
- Zero setup times;
- No overlapping of batches; and
- Work-orders are released following the DBR methodology and move through the manufacturing facility according to the heuristic First Come First Serve[1].

Quality defects, sequence dependent setup times, and work-station failure will cause work-orders to be delayed and arrive at the buffer later than planned. The simulation assumptions

[1] As DBR strategically places buffer resources, use of sequencing rules previously used to control queue lengths, is unnecessary (Gardiner, 1993). However, more recent work by Hurley (1996) indicates that DBR may be made more robust with the addition of constraint focused sequencing rules. For the purposes of this research, work-orders are sequenced according to the heuristic first come first serve.

while stylised are not restrictive as variability is measured using an Umble style "catch-all" disruption factor. Little generality is lost and the research results are expected to be applicable in a variety of manufacturing facilities.

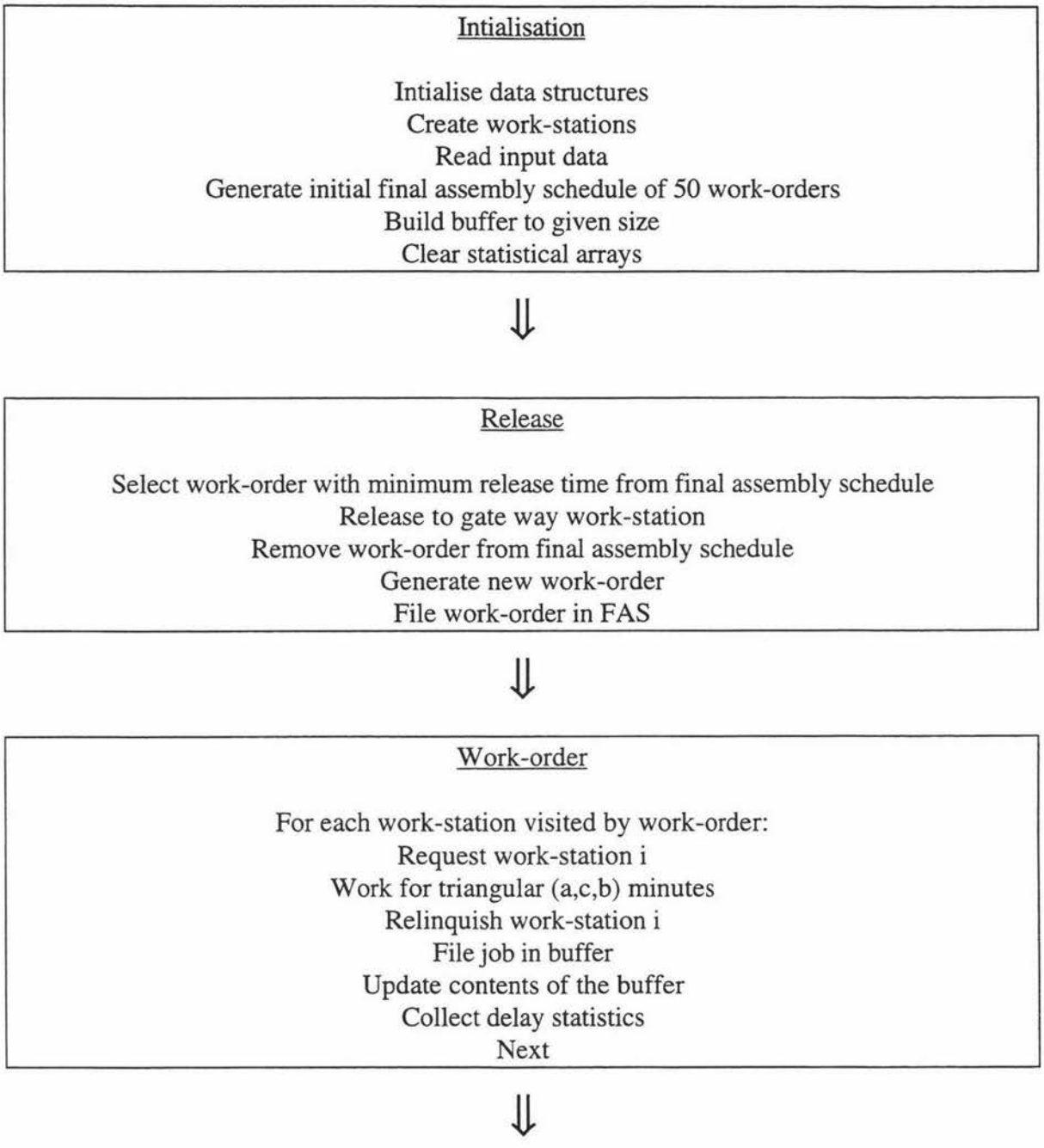
6.4 Computer Implementation

The simulation model was implemented in SIMSCRIPT II.5 (CACI, 1996). SIMSCRIPT II.5 is a powerful general purpose discrete event simulation package and has a number of desirable features including:

- flexibility which ensures that it is capable of modeling complex systems that are often encountered in manufacturing facilities;
- access to a large number of input distributions including the triangular distribution;
- an ability to automatically collect and collate statistics. Reporting and analysis of results is simplified; and
- a programming syntax loosely based on natural language making documentation, debugging, and validation simpler and quicker.

The simulation program begins by intialising all the data structures. The permanent entity, *product type*, contains information on routing and processing requirements for a given product type. The temporary entity, *job*, contains information on product type, batch size, release time and scheduled starting time at the constraint work-station. Associated with each job is also a number of statistics including cycle time, delay time and disruption factor.

Work-orders are created and filed in the final assembly schedule (FAS). Each work-order is released according to its release time. Work-orders are then processed by the non-constraint work-stations and reach the constraint buffer. When the constraint work-station completes the processing of a work-order, the buffer is examined and the work-order with the minimum starting time is selected for processing. The simulation program is run for a pre-determined period of time at which point it terminates, displays the statistical results and resets the statistical arrays. Program logic is broken down into 5 segments as in Figure 6.1. Simulation code is found in Appendix C.


Intialisation
Intialise data structures
Create work-stations
Read input data
Generate initial final assembly schedule of 50 work-orders
Build buffer to given size
Clear statistical arrays
Release
Select work-order with minimum release time from final assembly schedule
Release to gate way work-station
Remove work-order from final assembly schedule
Generate new work-order
File work-order in FAS
Work-order
For each work-station visited by work-order:
Request work-station i
Work for triangular (a,c,b) minutes
Relinquish work-station i
File job in buffer
Update contents of the buffer
Collect delay statistics
Next

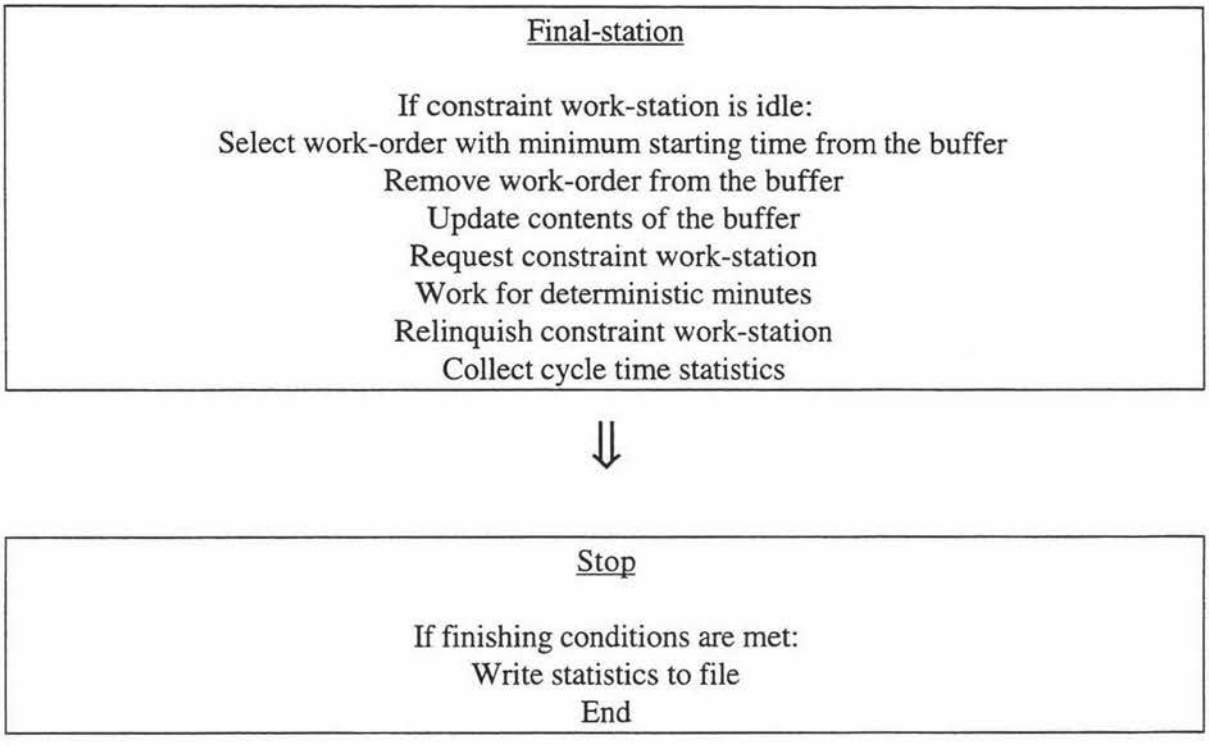


Figure 6.1: Computer Simulation Logic

6.5 Model Verification

Creditability is an important issue in simulation modeling. Unless the simulation output can be trusted, any subsequent analysis is meaningless. Model verification is an essential activity. Law *et al* (1991) defines the process of verification as:

> "...determining that a simulation computer program performs as intended." (pp. 299)

which differs from validation, which is concerned with whether the manufacturing model is an adequate representation of a DBR or MRP driven job-shop.

The manufacturing models are hypothetical and therefore difficult to validate. Section 5.3 argued that the manufacturing model contains the appropriate level of detail to model a

typical DBR implementation. MRP driven job-shop was constructed after discussion with an MRP expert.

Law *et al* (1991) gives excellent general advice on the how to of model verification. A number of their suggestions are followed. This section discusses how the simulation model was rigorously verified. In particular:

- Subsections of the simulation model were independently coded and tested.
- A series of informal structured walk throughs were used to confirm simulation logic.
- A person with extensive programming experience with SIMSCRIPT II.5, help with coding, debugging and walk throughs.
- Simulation output was tested for reasonableness. For the DBR model, the material release and output rate were measured and compared with the average theoretical rate. Further, the theoretical lead time to the constraint was compared with the actual lead time to the constraint. Under ideal conditions when the constraint work-station is fully utilised, both lead times should be equal. In both instances, no significant differences were found ($p>0.10$).
- The simulation model was simplified and simulation output for cycle time, buffer size, and delay times agreed with theoretical figures.
- A "trace" was used to confirm code. The scheduled starting and release times for each work-order was calculated manually and compared with simulation output. Both agreed.
- Statistical distributions were confirmed "post facto". See Appendix B.
- As a final test, the simulation model output was compared to the output of a simple model implemented in SimFactory (CACI, 1993). Results agreed for 1 work-order, 2 work-orders, 3 work-orders, and 10 work-orders - which is sufficient to capture all of the of work-station and product type interactions.

6.6 Methodological Issues

The probability distributions used to generate batch sizes and the processing times ensures that a single simulation run can not be representative of the manufacturing model. Law *et al* (1991) describes simulation as:

> "...a computer based statistical sampling technique." (pp. 523)

In order for simulation results to be meaningful, the simulation study should be statistically designed and analysed. Law's observation has an important implication when assessing the effectiveness of the fuzzy logic model: the comparison with Umble's heuristic should be statistically sound.

There are two significant problems associated with a simulation model output. First, is the simulation output is auto-correlated, nearly always non-stationary and often non-normal. Traditional statistical techniques such as ANOVA requiring independently and indentically distributed observations are inappropriate. Second, initial conditions cause atypical and uncharacteristic behaviour which can bias the output. Both these problems are discussed in detail in sections 6.6.1 and 6.6.2.

6.6.1 *Auto-correlation*

As the raw output of a simulation study is auto-correlated it can not be analysed using traditional statistical techniques.

Given a stochastic output process $\{Y_i\}$:

$y_{11}, \ldots, y_{i1}, \ldots, y_{m1}$

$y_{12}, \ldots, y_{i2}, \ldots, y_{m2}$

... ...

$y_{1j}, \ldots, y_{ij}, \ldots, y_{mj}$

... ...

... ...

$y_{1n}, \ldots, y_{in}, \ldots, y_{mn}$

where y_{ij} is the i^{th} observation (of m) of the j^{th} replication of n replications.

$\{y_{1j}, \ldots, y_{ij}, \ldots, y_{mj}\}$ is auto-correlated.

Law *et al* (1991) suggest a relatively simple technique for obtaining independent (auto-correlation free) estimates of output means which is adopted by this research. If each replication is subject to the same initial conditions, and the statistical arrays are cleared after each replication, $\{y_{j1}, y_{j2}, \ldots, y_{jn}\}$ (j = 1 to m) and $\{y_{i1}, y_{i2}, \ldots, y_{in}\}$ (i = 1 to m) ($i \neq j$) are independent. A reliable and unbiased estimate of $\bar{y}_i$ can then be calculated using equation 6.7.

$$\bar{y}_i = \frac{\sum_{j=1}^{n} y_{ij}}{n} \quad \forall\, i \qquad \textbf{(6.7)}$$

To illustrate, Figure 6.2 depicts the auto-correlation function for 10 cycle time means. Correlation's at each lag are insignificant at the 5% level of significance.

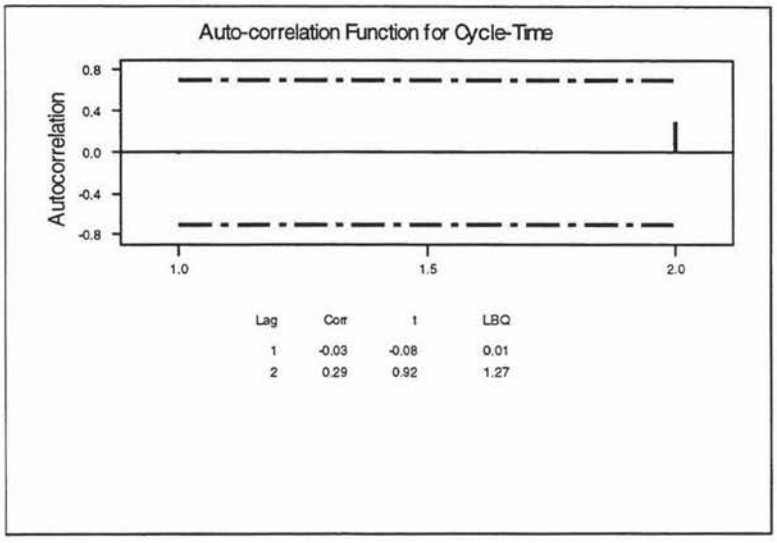


Figure 6.2: Auto-correlation Function of Mean Cycle Time (MPC = 30% and cv = 0.07)

Since the $\{\bar{y}_i$'s$\}$ are iid, $\bar{\bar{y}}$ (average of $\{\bar{y}_i$'s$\}$) can readily be estimated and analyised using traditional statistical techniques.

6.6.2 *Steady State Conditions*

A significant problem with non-terminating simulation models is the difficulty in determining when steady state conditions have been reached. Only after steady state conditions have been reached can data collection begin. The behaviour observed during the transient or warm-up period is atypical and does not reflect the true behavioiur of the system under study. During this time, estimates of mean and variances are likely to be biased. Law *et al* (1991) expresses concisely the steady state problem of reliably estimating the output mean ν, where:

$$\nu = \lim_{i \to \infty} E[Y_i] \qquad \textbf{(6.8)}$$

where $\overline{Y}$ is used to approximate $E[Y_i]$.

Consider a stochastic process $\{Y_i\}$ with population mean, ν. Law *et al* (1991) formally defines steady state as:

> "If $F_i(y|I) \rightarrow F(y)$ as $i \rightarrow \infty \; \forall \; y$ and for any initial conditions I, then F(y) is called the steady state distribution of the output process Y_1, Y_2, ..." (pp. 525)

$\{Y_i's\}$ will not necessarily be identically distributed once steady state conditions have been reached. Instead, $\{Y_i's\}$ will be approximately identical distributed (or treated as such) (Law *et al*, 1991).

In practice, steady state means:

- the effects of the initial conditions are insignificant (since $F_i(y|I) \rightarrow F(y)$ as $i \rightarrow \infty \; \forall$ y);
- the observed behaviour of the manufacturing model is typical; and
- it is possible to obtain reliable and unbiased estimate of the mean and variance of y.

The initial-data deletion method is commonly used to remove any bias caused by initial conditions when estimating ν. Assuming a warm-up period of l observations and given m observations, $E[Y_i]$ can be estimated using equation 6.9. That is, the initial-data deletion method discards the first l observations (which are atypical and uncharacteristic) and estimates ν from the remaining $(m - l)$ (which are typical) observations.

$$\overline{Y} = \frac{\sum_{i=l+1}^{m} Y_i}{m - l} \qquad \textbf{(6.9)}$$

This simple and effective method suffers in practice as it offers no advice on how large l should be except that l should be "sufficiently large". In practice, the length of the transient or start period is difficult to determine. Usually, l is determined graphically.

Welch's method (Welch, 1981) is a simple graphical method that is commonly used to determine *l*. Simulation output is smoothed and plotted as a time series using a moving average filter of length w. As the model enters steady state, $E[\bar{y}_i]$ approaches ν. The value for *l* can be determined visually when:

$$E[Y_l] \approx \lim_{i \to \infty} E[Y_i] \qquad \textbf{(6.10)}$$

6.6.3 *Replications*

The number of simulation run replications needed to yield a given precision can be calculated using procedure suggested by Law *et al* (1991). Let X be the random variable of interest to the experimenter.

- Replicate the simulation n_0 times; set $n = n_0$
- Compute mean output (μ_X) and standard deviation (s_n)
- Let $\sigma(s,n) = t_{(n-1,\ 1-\alpha/2)} * (s_n^2/n)^{1/2}$
- If $\sigma(s,n)/\mu_X \leq$ precision required then use n replications
- Else increase n_0 by one and repeat

6.6.4 *Data Collection Techniques*

Ginting (1995) reviews two common data collection approaches used in simulation modeling. The first approach is to run each replication independently. Each replication begins with a empty manufacturing facility and is run until the simulation model reaches steady state. Statistics are collected for a data collection period and the simulation then terminates. The second approach, after steady state conditions have been reached, runs each replication one after another with the finishing conditions of the previous replication used as the starting condition of the current replication.

Both approaches are mathematically equivalent, but as the second approach has a number of advantages over the first approach. Significant savings in computer time are possible. The second approach is simpler to implement, and greatly facilitates the collation and calculation of epoch means.

For both approaches, each replication is divided into a number of segments or epochs. During each epoch, performance statistics are collected and the epoch means are calculated. Correlation free means are then calculated using the procedure outlined in section 6.6.1.

Figure 6.3 illustrates a data collection scheme based on 3 epochs and 2 replications. The i^{th} epoch mean of the j^{th} replication is denoted by E_{ij}.

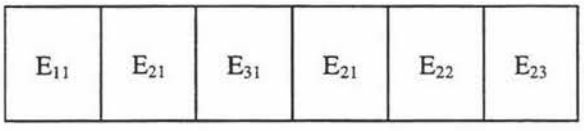


Figure 6.3: Data Collection

6.6.5 *Data Collection in this Research*

The second approach surveyed by Ginting (1995) was adopted in this research. The simulation model detailed in section 6.4 takes 2 hours of real time to reach steady state conditions. As 5 replications are made, the second approach allows 8 hours of a real time to be saved.

In order to specify the data collection scheme for the manufacturing model the following variables were determined:

- time the simulation model enters steady state;
- epoch length (expressed in time) and number of epochs per replications; and
- number of replications required for a given precision.

Steady State Conditions

The physical interpretation of steady state for the manufacturing model is as follows. As the constraint work-station's buffer is built prior to work-order release, only non-constraint queues are subject to start up or transient conditions. Work-station utilisation during this period will fluctuate and converge to a stable level when steady state conditions are reached. This can be used to identify when steady state conditions have been reached.

Work-station utilisation depends on mean protective capacity and variability. Accordingly, l will vary depending on the factor levels for these quantities. A manufacturing experiment characterised by high variability and low protective capacity will reach steady state slower than other environments. l in this case represents a lower bound and can be assumed for all experimental conditions.

DBR model

The value for l was determined graphically using Welch's method for non-constraint utilisation (an appropriate measure for steady state) and delay time. The mean delay time and nonconstraint utilisations were sampled every week or 10080 minutes (24 hours/day, 7 days/week).

A moving average filter of w>1 was not applied to transient data as the high frequency variation was practically "insignificant". Figures 6.4 to 6.8 show time series plots of mean non-constraint work-station utilisation. Figure 6.9 shows a time series plot of mean delay time. It can be readily seen that steady state conditions are reached by at least the 500^{th} week or the 5040000 minute. It was decided to clear the statistical arrays after the 500^{th} week - an action equivalent to using the deletion method detailed in section 6.6.2.

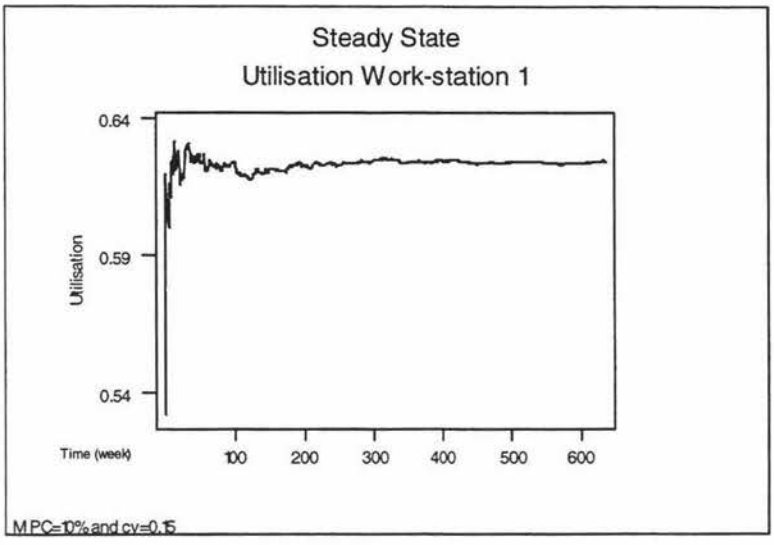


Figure 6.4: Mean Utilisation of Work-station One

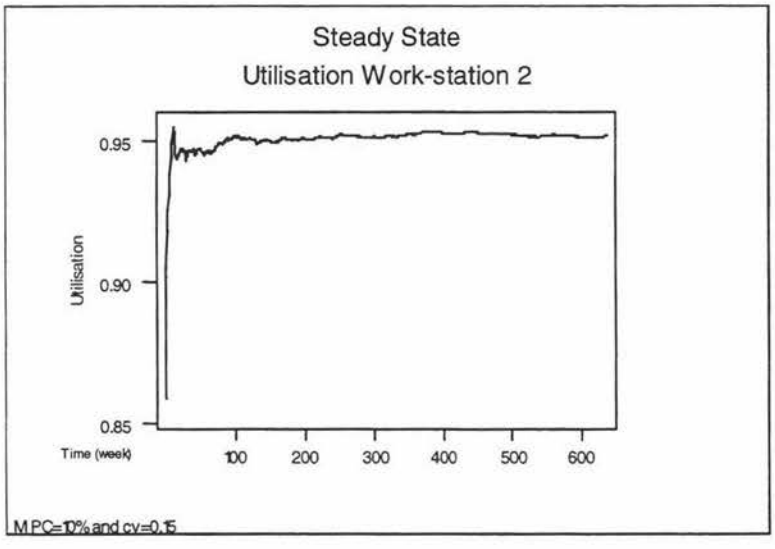


Figure 6.5: Mean Utilisation of Work-station Two

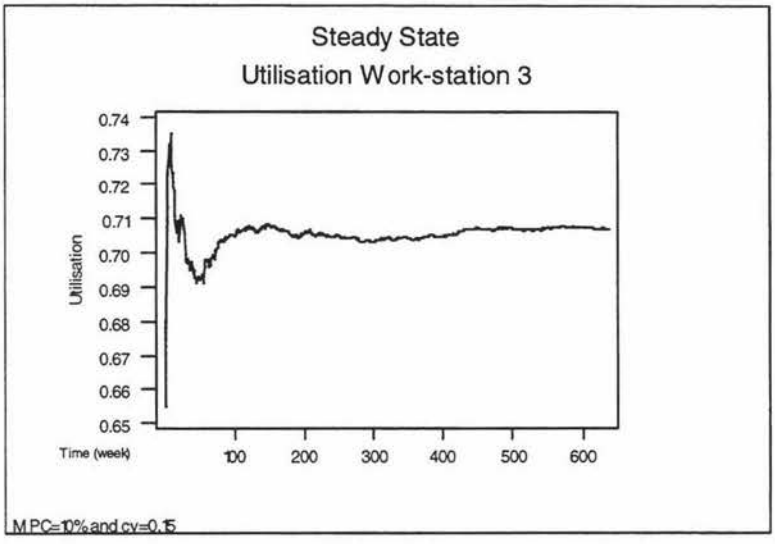


Figure 6.6: Mean Utilisation of Work-station Three

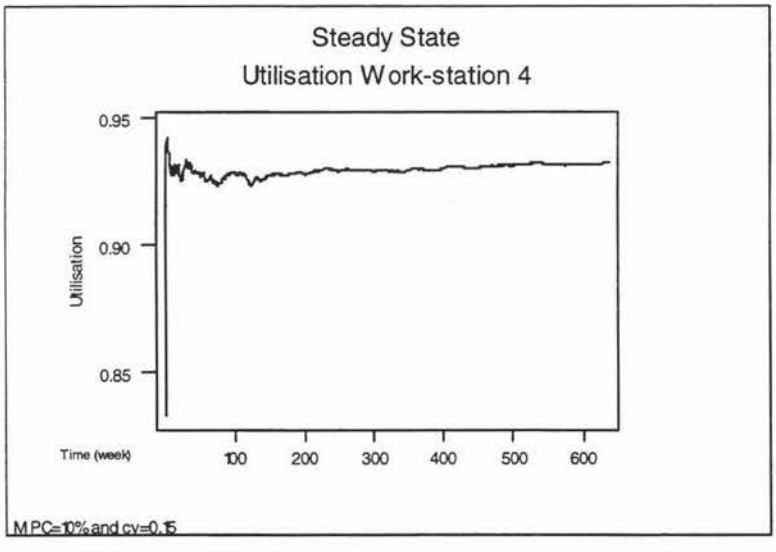


Figure 6.7: Mean Utilisation of Work-station Four

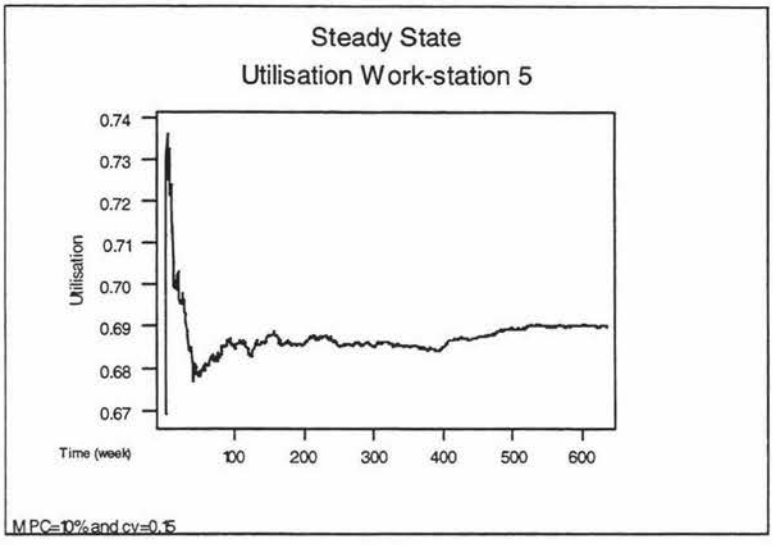


Figure 6.8: Mean Utilisation of Work-station Five

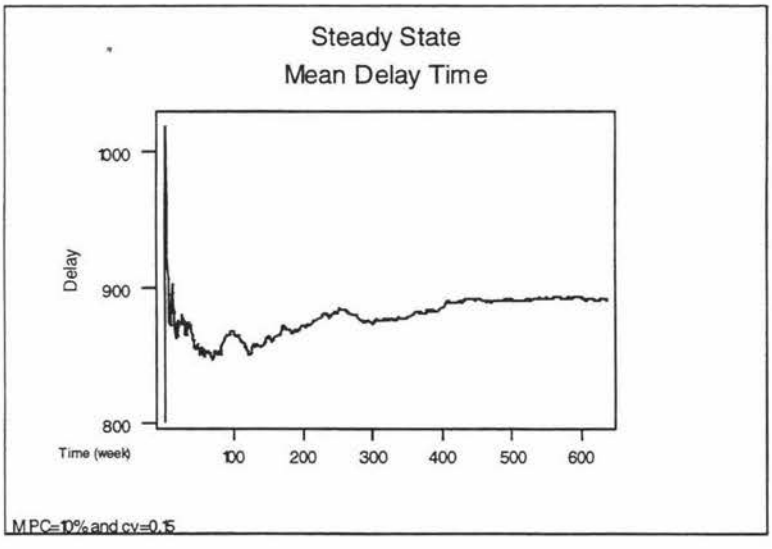


Figure 6.9: Mean Delay Time

MRP Driven Job-Shop Model

As with the DBR model, the delay time and work-station utilisation was sampled every 100 minutes and plotted in Figures 6.10 to 6.14. The value for l was then determined graphically using Welch's method (w=1).

The material release policy under MRP attempts to maximises the utilisation of the gate-way work-stations. Utilisation of work-stations one and two enter steady state conditions quickly - well in advance on the remaining work-stations. Accordingly, the time series plot these works-stations are not included in the analysis.

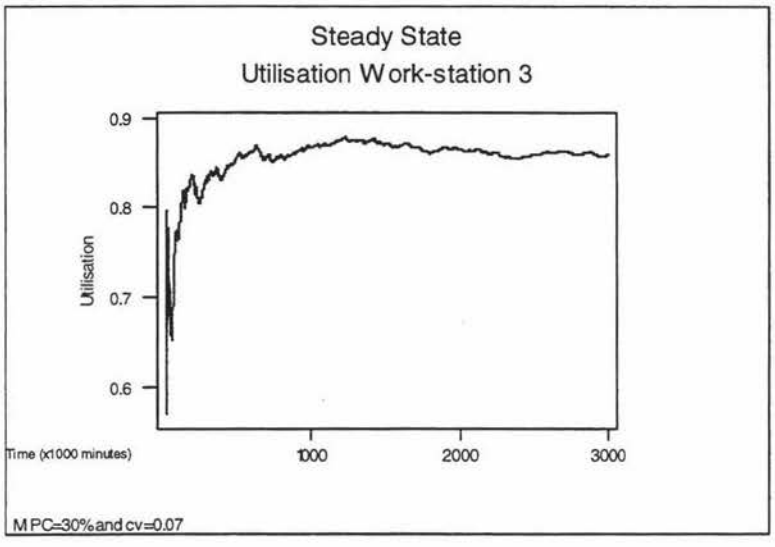


Figure 6.10: Mean Utilisation of Work-station Three

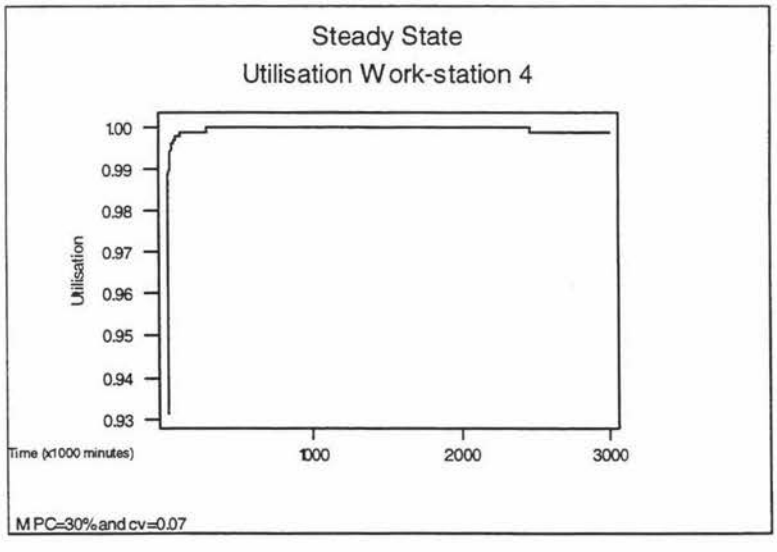


Figure 6.11: Mean Utilisation of Work-station Four

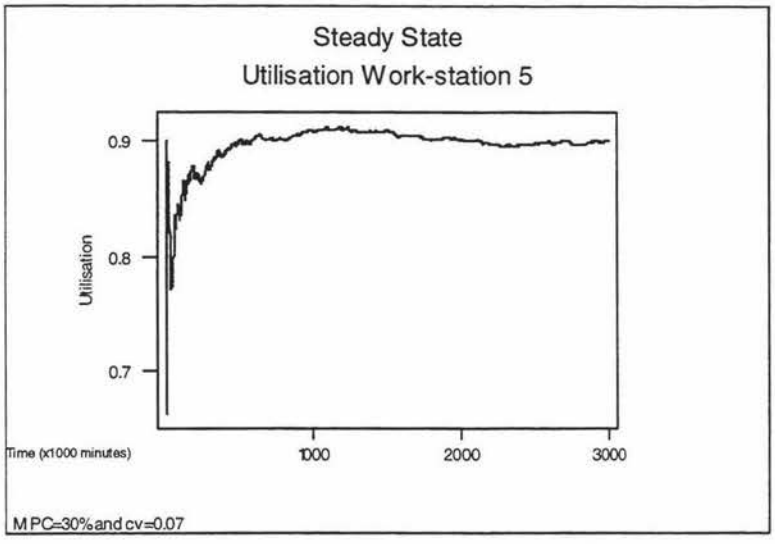


Figure 6.12: Mean Utilisation of Work-station Five

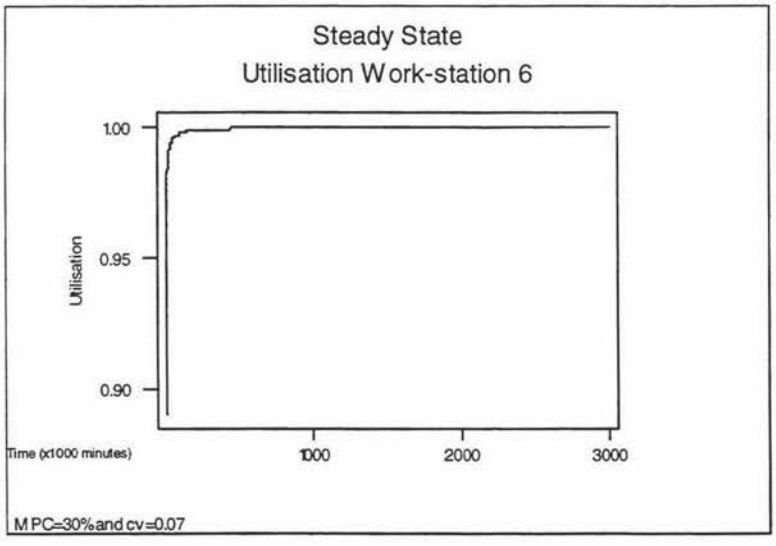


Figure 6.13: Mean Utilisation of Work-station Six

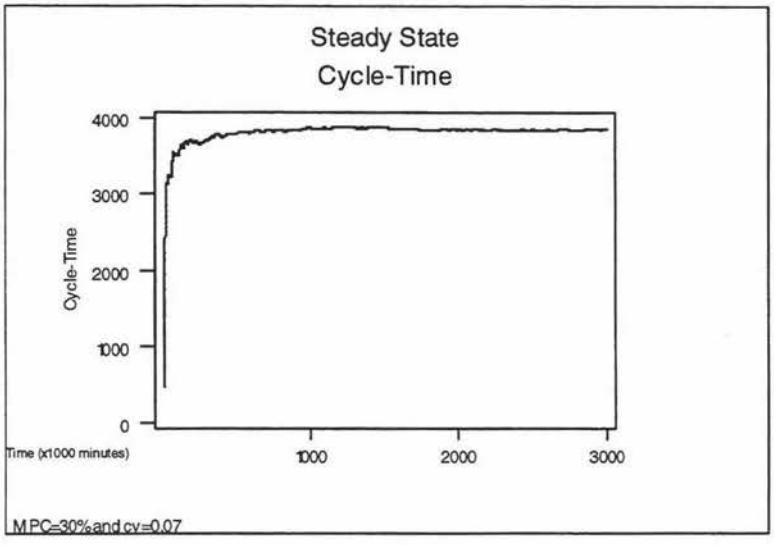


Figure 6.14: Mean Cycle-Time

It is apparent from Figures 6.10 to 6.14 that the model is in steady state conditions at time 3,000,000 minutes (or ≈ 298 weeks).

Epoch Length and Number

Each epoch length needs to be long enough to capture a detailed representation of the manufacturing facility. This length depends on the complexity of the manufacturing model. The manufacturing model used in this research consists of a constraint work-station producing a work-order on average every 117 minutes. It is desirable to "sample" at least 5 work-orders from each of the 5 product types. It seems reasonable to set each epoch length at least 5x5x117, or 2925 minutes. Instead, the epoch length was set at a pessimistic 7500 minutes since the product type in each work-order is randomly sampled.

10 epochs per replications - though more than enough - was adopted by this research since the relative cost of computing is small.

Number of Replications

Each series of epochs was replicated 10 times. As each replication consists of 10 epoch means and epoch means are averaged over each replication to produce a correlation free mean, the number of replications is 10.

For 10 replications, Table 6.5 shows the relative precision of each output statistic (MPC=10% and cv=0.15). The desired level of precision is 5%.

Measure	Relative Precision (%)
Overall cycle time	0.3
Overall delay time	4.1
Constraint Utilisation	≈ 0.0

Table 6.5: Relative Precision Achieved with 10 Replications (α=0.05)

6.7 Experimental Design

Two factors, mean protective capacity (MPC) and coefficient of variation (cv) are varied over three levels (low, medium and high) in order to simulate a variety of manufacturing facilities or environments. Factor levels are denoted by -, 0 and +.

To investigate the research hypothesis, a replicated 3x3 ANOVA design is utilised[2] in order to sample from the delay time distribution which is then used to determine the appropriate buffer size. For each simulation run, the buffer size was set at 1000 minutes. Experimental treatments can be found in Table 6.6.

[2] There is little utility is using a 3^m3^n F design as the number of simulation runs is small.

Variability	Mean Protective Capacity
-	-
-	0
-	+
0	-
0	0
0	+
+	-
+	0
+	+

Table 6.6: 3x3 Experimental Design

An important experimental design decision is the choice of the levels. Levels should be varied as much as practical. Considering the variability and mean protective capacity factors:

- Experimental results show that cv is limited to the range [0, 0.15] as higher values of cv may cause the sixth work-station (constraint) to become a non-constraint.
- MPC is limited to the range [0, 50]. The utilisation of a non-constraint work-station is bounded at 0 (see equation 6.3) and a manufacturing facility with a MPC of 50% or more was thought to be unrealistic.

Many researchers arbitrarily assign factor levels but generally represent the researchers intuitive understanding of the effect of each factor. The manufacturing model used in this research is hypothetical and as a result there is no intuitive understanding of what the appropriate factor levels are. Further it was important that each level had a statistically different effect. To address this, pre-experimental experimentation was used.

6.7.1 *Pre-experimental Design Experimentation*

In order to assess what value of mean protective capacity, low, medium and high were representative of, the mean protective capacity levels were varied in 10% increments over

the range [10, 50]. cv was held at a constant level (0.10) and the buffer size was set at 1000 minutes.

Delay times are calculated and recorded once the model had entered steady state. These delays times for epoch and replication are combined to form an empirical delay time cummulative density function. Equations 5.30 and 5.31 were then used to calculate the likely interval that contains the 95% delay time quartile - the appropriate buffer size adopted for the purposes of this research. The effect of changing the MPC on the appropriate buffer size is shown in Figure 6.15.

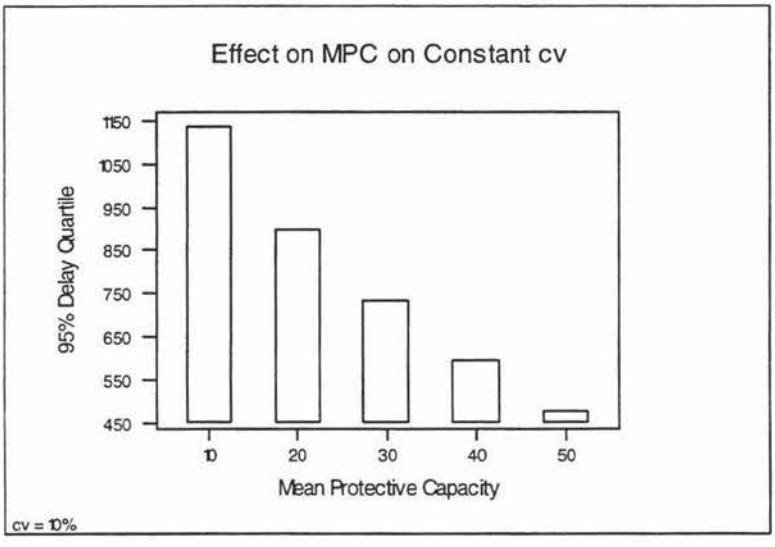


Figure 6.15: Effect of Mean Protective Capacity on 95% Delay Time Quartile

Three levels were chosen: low mean protective capacity at 10%; medium protective capacity at 30%; and high protective capacity at 50%.

Likewise, determination of what value of cv represented low, medium and high was investigated. cv was varied in 0.05 increments over the range [0.01, 0.15]. Mean protective capacity level was held at a constant level (50%) and the buffer size was set at 1000 minutes.

The 95% delay time quartile was calculated from data when the model entered steady state. Figure 6.16 shows the effect of changing cv on the appropriate buffer size.

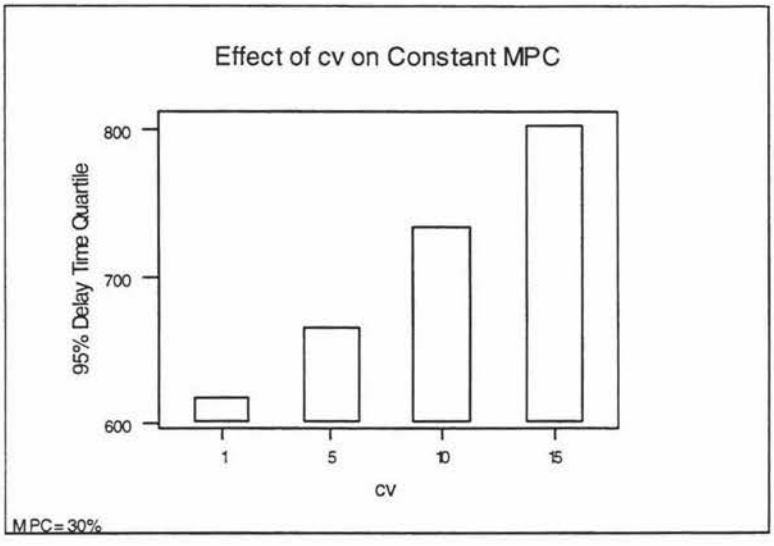


Figure 6.16: Effect of cv on 95% Delay Time Quartile

Accordingly, three levels were chosen: low mean protective capacity at 10%; medium mean protective capacity at 30%; and high mean protective capacity at 50%.

6.7.2 *Pre-Experimental Design Experimentation: MRP Driven Job-Shop*

The Umble buffer size is calculated from half the manufacturing lead time in a MRP driven job-shop. For the purposes of this research, a manufacturing environment is adopted to calculate a representative Umble buffer size: MPC = 30% and cv = 0.07.

This is done for two reasons: (1) MPC has no real meaning in a MRP environment (that attempts to maximise work-station utilisation); and (2) the determining factor of the manufacturing lead time is the material release policy. Average values for MPC and cv are adopted.

Chapter Seven

Experimentation and Analysis

7.1 Introduction

A fuzzy logic buffer sizing model was developed and outlined in chapter four using DBR's vague and non-specific buffer sizing advice. An important step in determining whether fuzzy logic is applicable to the BMP is a demonstration of its ability to effectively size buffers. In particular, a fuzzy solution should size buffers consistently and significantly better than a commonly used "quick and dirty" technique such as Umble's heuristic.

To test this, the simulation models detailed in chapter six were used to determine the Umble buffer size and the appropriate buffer size (defined by the 95% delay time quartile) for a given manufacturing environment. Against these, the effectiveness of the fuzzy buffer sizing model was gauged.

This chapter presents the results and discusses the applicability of the fuzzy solution approach to buffer sizing. In particular, sections 7.2 and 7.3 contrast the effectiveness of fuzzy logic model and Umble's heuristic. Section 7.4 examines the practical significance of the results for three representative manufacturing environments. This chapter finally concludes with a discussion of the practical significance of the fuzzy logic solution.

7.2 Buffer Sizing and Effectiveness

As previously mentioned, buffer size is dependent on Mean Protective Capacity (MPC) and the coefficient of variation of processing times (cv) that define a given manufacturing environment. Nine hypothetical manufacturing environments based on 3 combinations of high, medium and low MPC and cv (detailed in section 6.7.1) were simulated to estimate the effectiveness of each buffer sizing technique.

Effectiveness is dependent on the how closely each technique sizes the buffer to the appropriate buffer size. For the purposes of this research, the appropriate buffer size is defined by the 95% delay time quartile.

In other words, a buffer sizing technique is said to be effective if it minimises:

$$BE = | \text{Estimated Buffer Size} - 95\% \text{ Delay Time Quartile} | \quad \textbf{(7.1)}$$

which forms the basis of research hypothesis proposed in section 5.1. That is,

H_O: $BE_{(\text{Fuzzy logic})} \geq BE_{(\text{Umble})}$

H_A: $BE_{(\text{Fuzzy logic})} < BE_{(\text{Umble})}$

7.2.1 *Estimated Buffer Size*

For each manufacturing environment, the estimated buffer size is determined using the fuzzy logic model and Umble's heuristic. Results can be found in Table 7.1.

The Umble buffer size is calculated from half the manufacturing lead time in a MRP driven job-shop, while the fuzzy logic model was used to generate fuzzy buffer estimates using the cv and MPC as buffer sizing inputs. A worked example, illustrating the process of fuzzification, implication, aggregation and defuzzification is given in section 4.6.

Manufacturing Environment	cv	MPC (%)	Umble Buffer (minutes)	Fuzzy Logic Buffer (minutes)
1	-	-	1926	754
2	-	0	1926	669
3	-	+	1926	463
4	0	-	1926	1250
5	0	0	1926	714
6	0	+	1926	615
7	+	-	1926	1430
8	+	0	1926	1190
9	+	+	1926	783

Table 7.1: Estimated Buffer Sizes for Umble's Heuristic and the Fuzzy Logic Model

7.2.2 *Appropriate Buffer Size*

For each manufacturing environment, the appropriate buffer size was calculated using the methodology presented in sections 5.5 and 5.5.1.

The delay times of each work-order for every data collection epoch is recorded and combined to form a dataset from which 1000 delay times were randomly sampled. The delay times were then ranked and the confidence interval $[D_{i:n}, D_{j:n}]$ was calculated and used to determine the appropriate buffer size, $D_{0.95}$. Results are given in Table 7.2.

Manufacturing Environment	cv	MPC (%)	95% Delay Time Quartile (minutes)
1	-	-	916 [889, 936]*
2	-	0	620 [602, 644]
3	-	+	418 [408, 432]
4	0	-	1060 [1037,1100]
5	0	0	692 [666,715]
6	0	+	460 [446,471]
7	+	-	1527 [1488,1572]
8	+	0	791 [766, 831]
9	+	+	522 [508, 536]

* Estimated 95% delay time and 95% Confidence Interval.

Table 7.2: Appropriate (95% Delay Time Quartile) Buffer Size

7.2.3 *Buffer Effectiveness*

Using the estimated buffer size and appropriate buffer size, the buffer effectiveness can be calculated for each buffer sizing technique and manufacturing environment (see equation 7.1). Results are presented in their raw form in Table 7.3 and for ease of comparison are also expressed as the percentage error which is plotted in Figures 7.1 and 7.2. Note that Figures 7.1 and 7.2 have different scales.

Experimental Treatment	cv	MPC	Umble Buffer Effectiveness	Fuzzy Logic Model Effectiveness
1	-	-	1010 (110.26)	162 (17.69)*
2	-	0	1306 (210.65)	49 (7.90)
3	-	+	1507 (359.67)	44 (10.5)
4	0	-	866 (81.70)	190 (17.92)
5	0	0	1234 (178.32)	22 (3.18)
6	0	+	1467 (319.61)	156 (33.90)
7	+	-	399 (26.13)	97 (6.35)
8	+	0	1135 (143.49)	399 (50.44)
9	+	+	1404 (268.97)	261 (50.00)

* BE of 162 minutes representing an error of 17.69%

Table 7.3: Buffer Effectiveness

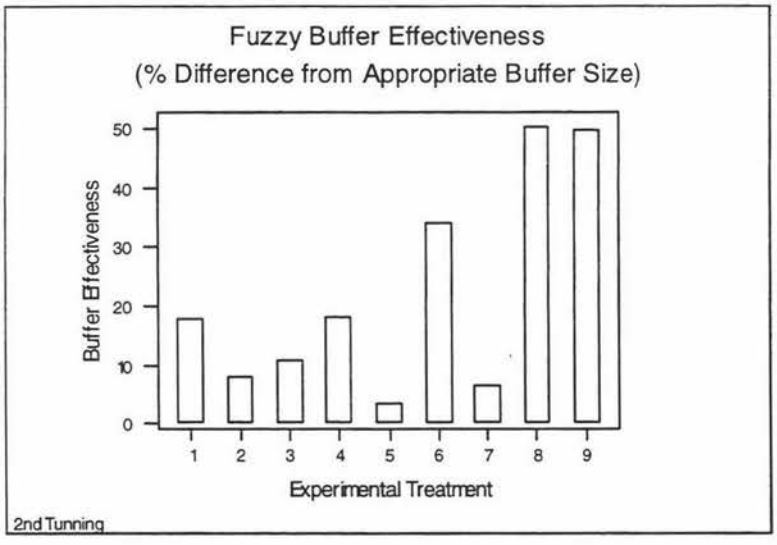


Figure 7.1: Buffer Effectiveness of the Fuzzy Logic Model

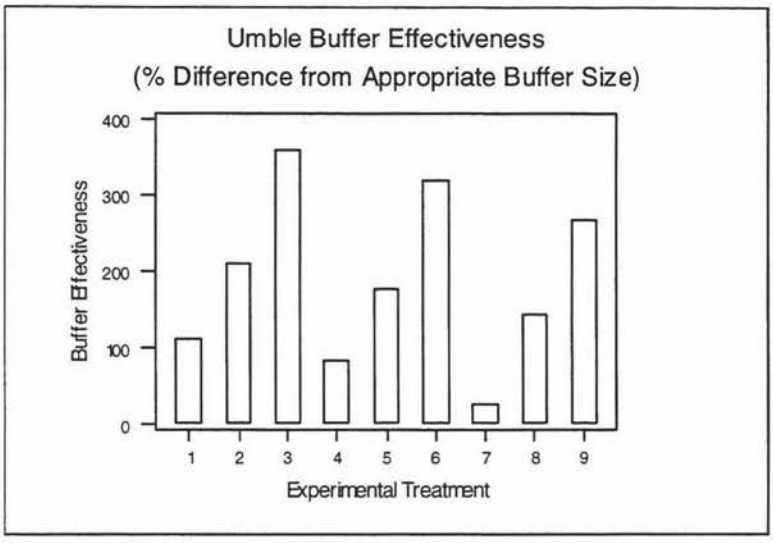


Figure 7.2: Buffer Effectiveness of Umble's Heuristic

7.3 Discussion

The simulation results show that fuzzy logic sizes buffers consistently and significantly better than Umble's heuristic. This is readily seen from Figure 7.1 where the effectiveness of the fuzzy logic model easily surpasses Umble's heuristic - more closely minimising equation 7.1. The research hypothesis is formally tested with a two sample t-test. The null hypothesis is rejected as expected, that is, the fuzzy logic model sizes buffers more effectively than Umble's heuristic (t=-8.03, p=0.000, df=9). Summary statistics are presented in Table 7.4.

Sizing Technique	Sample Size	Mean BE	Standard Deviation	Minimum	Maximum
Fuzzy Logic	9	153	121	22	399
Umble	9	1148	351	399	1507

Table 7.4: Summary Statistics

Despite the obvious effectiveness of the fuzzy logic solution, examination of Table 7.3 reveals two problems with the fuzzy logic model. Firstly, while Umble's heuristic may

overstate the protection to maintain a given output rate, it does not understate it as the fuzzy logic model does. This may cause a potential loss of output. Secondly, the fuzzy logic model performs poorly when the cv is low. However, the fuzzy logic model can be tuned with this information, and as such is not major problem.

Another point to note from Table 7.3 (and Figure 7.2) is that Umble's heuristic performs relatively well for an environment of low protective capacity and high variability (MPC=10%; cv=0.15). Umble's result is similar - but not as effective - as the fuzzy buffer size. Low protective capacity and high variability often characterise a job-shop operating under MRP-style decision rules.

7.4 Practical Significance

To develop practical solutions to the BMP this thesis has adopted the philosophy that the buffer sizing technique need not be optimal but offer a better solution than what is currently available. This section interprets the practical significance of the results presented in section 7.2.3 and 7.3.

As discussed, the goal of the BMP research is an implementable solution that enhances the time-based competitiveness of a manufacturing facility, maximising due date performance and minimising the manufacturing lead time. Practical significance is perhaps best interpreted in terms of mean constraint utilisation and cycle time.

Three of the nine simulation experiments were selected as representative manufacturing environments. The manufacturing environments were:

- high protective capacity (MPC=50%) and low variability (cv=0.01)
- low protective capacity (MPC=10%) and high variability (cv=0.15); and

- medium protective capacity (MPC=30%) and medium variability (cv=0.07).

which for the purposes of this research represent a well-behaved, an out of control, and a normal manufacturing facility.

For each of the three environments the effectiveness of the buffer sizing techniques were contrasted using mean constraint utilisation and cycle time. Results are presented in Table 7.5.

The utilisation of the constraint was protected in all instances. The cycle times alone are used to demonstrate the practical significance of the fuzzy logic model. As expected, the fuzzy logic model outperforms Umble's heuristic. The most obvious instance is for a well behaved manufacturing facility, where the cycle time for the fuzzy buffer is 747.66 minutes compared to 2255.66 minutes for Umble's heuristic. The average cycle time under a fuzzy buffer is 1001 minutes smaller than Umble's heuristic. This roughly corresponds to a difference 8.6 work-orders.

The difference in cycle times between fuzzy logic and the appropriate buffer size is practically insignificant. The fuzzy logic model the average cycle time is 22 minutes longer than the appropriate buffer size. This corresponds to a difference of 0.2 work-orders - at a practical level, insignificant. One must conclude that the fuzzy solution sizes buffers effectively in terms of constraint utilisation and cycle time compared to either the 95% delay time quartile or Umble's buffers.

Sizing Technique	Manufacturing Environment	Mean Constraint Utilisation (%)	Mean Cycle-Time (minutes)
95% Delay Statistic	3	100	747.66 (743.77,751.56)*
	5	100	1106.60 (1101.49, 1111.72)
	7	100	2067.08 (2060.94, 2073.22)
Umble's Heuristic	3	100	2255.66 (2251.77, 2259.56)
	5	100	2340.60 (2335.49, 2345.72)
	7	100	2425.53 (2419.18, 2437.87)
Fuzzy Logic	3	100	792.66 (788.77, 796.56)
	5	100	1128.60 (1123.49, 1133.72)
	7	100	2067.12 (2060.98, 2073.26)

* Mean value with 95% Confidence Interval

Table 7.5: Simulation Results

7.5 Summary

The simulation results show that the fuzzy logic model performs significantly better than Umble's heuristic. Moreover, effectiveness is practically significant. Further, the fuzzy logic model's performance approaches the performance of the 95% delay time statistic. These results are encouraging and reinforce the argument that fuzzy logic is likely to produce implementable and effective solutions to the BMP. While simulation allows experimental conditions to be tightly controlled, any result is still tenuous and the ultimate test of the applicability of fuzzy logic rests on a practical implementation in a manufacturing facility. This issue and others relating to extensions to this research are addressed in chapter eight.

Chapter Eight

Future Work

8.1 Introduction

The BMP is a rich area for manufacturing systems research as it is concerned with the two basic phenomena of manufacturing: statistical fluctuations and random events in chains of dependent events. A number of extensions to the work are possible and are detailed in this chapter. For convenience they are categorised into one of five areas:

1. improvements to the fuzzy logic model;
2. methodological aspects;
3. role of protective capacity;
4. buffer management technologies; and
5. quality in research.

8.2 Improvements to the Fuzzy Logic Model

This thesis presents a simple fuzzy buffer sizing model developed from intuition and general advice given by Cox (1994).

The fuzzy model is deliberately kept simple by utilising unoptimised membership functions and simple and commonly used implication, aggregation and defuzzification strategies. A number of improvements are possible, in particular:

- estimation of protective capacity;
- shape of the membership functions;
- optimisation of the rule confidences;
- implication and defuzzification strategy;
- tuning the fuzzy membership functions; and
- testing and implementation.

8.2.1 *Estimation of Protective Capacity*

As discussed, the operational definition of protective capacity based on equation 4.3 overestimates a manufacturing facilities protective capacity. The MPC metric fails to reflect

the intuitive understanding of the practitioner. An accurate representation of protective capacity may aid in setting good initial bounds on the protective capacity membership functions, reducing the need to tune each fuzzy term set.

The fuzzy rule base can be augmented to capture the intended meaning of equation 4.3. An unconditional fuzzy *if then* production rule is proposed:

Protective Capacity is *somewhat* in the *vicinity below* MPC

where the fuzzy hedges, *somewhat* and in the *vicinity below*, transform MPC into a fuzzy space, modelling the semantic interpretation of equation 4.3 as an absolute upper bound. For a discussion of fuzzy hedges see Cox (1994)

This allows protective capacity to be represented as a fuzzy space rather than a real-valued input. The relative truth of *high*, *medium* or *low* protective capacity can then be determined from the intersection with protective capacity membership functions and the fuzzy protective capacity space. See Cox (1994) for a worked example of using hedges in this way.

8.2.2 *Membership Function Shapes*

Triangular and trapezoidal membership functions are often adopted by practitioners and researchers as they are easy to visualise and simple to use. However, a large number of membership functions exist and for the BMP the most appropriate membership function is an open research question.

8.2.3 *Optimisation of Rule Confidences*

Rule confidences play an important part in the performance of a fuzzy system. Marsh *et al* (1995) have proposed a simple technique that combines expert knowledge with input-output

data to produce a more accurate estimate of rule confidences. They suggest that altering the rule confidences may provide a simpler way of optimising fuzzy systems than tuning the membership functions.

8.2.4 *Implication and Defuzzification Strategies*

While Zadeh min. and max. operators are commonly used a number of other operators exist. Examples include bounded sum and product and a variety of Yager type compensatory operators. Likewise, the centroid approach is one of a large number of defuzzification strategies. What the most appropriate implication and defuzzification strategy is, is also an open research question.

8.2.5 *Tuning the Fuzzy Membership Functions*

For the purposes of this research, the overlap and domain of each fuzzy set is determined using experience and intuition gained from the simulation experiments detailed in chapter seven. This is similar to how practitioners would set the membership functions in practice. The membership functions are not "optimal" and it is possible to do better. Some tuning may be necessary.

The genetic algorithm (G.A) has commonly applied to the problem of tuning the membership functions (Brown *et al*, 1995). However, the mathematical sophistication of the genetic algorithm is likely to be beyond the mathematical acumen of most practitioners who are not trained operations research analysts. A future area for research is the development of "practitioner-friendly" heuristics for tuning membership functions. An illustrative example is detailed below.

Example

If the constraint work-station is idle more than β% of the time, the buffer size membership functions can be adjusted, to increase the protection available. This is achieved by shifting the domain of the *large* buffer size (for example) by ε units along the universe of discourse. The dotted membership function, in Figure 8.1, offers ε more time units protection than the original membership defined by the solid line.

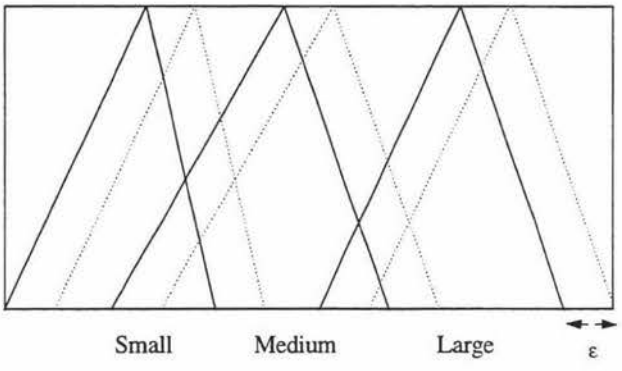


Figure 8.1: Buffer Size is too Small

A possible expression for ε is given by equation 8.1.

$$\varepsilon = (1 - (\eta + \beta)) \text{ x Previous Buffer Size} \qquad \textbf{(8.1)}$$

where β is the maximum acceptable idleness of the constraint and η is the actual utilisation of the constraint. β is management driven, while η can be estimated from shop floor records. Equation 8.1 is a linear approximation, as buffer sizing is characterised by diminishing returns (Baral, 1993).

Further, the effectiveness of intuition also needs to be investigated. Intuition, however, may be a suitable comparison for assessing the relative utility of any tuning heuristic.

8.2.6 *Testing and Implementation*

A limited number of manufacturing environments were used to test the effectiveness of the fuzzy buffer sizing model. More extensive testing is needed before a definitive conclusion can be reached about the fuzzy model's effectiveness. Preliminary results, however, are encouraging and certainly warrant further investigation into a fuzzy logic buffer sizing approach.

The ultimate aim of manufacturing system research should be at least one solution to a real world problem. Accordingly, there is a need to implement the fuzzy logic model in practice. This is perhaps the ultimate and final test of a buffer sizing technique's effectiveness. This would also provide an unique opportunity to examine other issues including a spread-sheet representation of a fuzzy buffer sizing model, setting of membership functions and the practical "how to" associated with the BMP.

8.3 Methodological Issues

8.3.1 *Further Simplification of the Manufacturing Model*

As mentioned, an important first step in demonstrating the applicability of a fuzzy buffer sizing solution is simulation (under controlled conditions). As a result, much effort has centred around the construction and verification of the simulation models detailed in chapter six. This process has been time intensive and due to the relative complexity of each model, prone to errors.

Simulation modelling is an important part of BMP research. Without simulation the effectiveness of a buffer sizing technique is likely to remain a matter of conjecture. Further simplification of the manufacturing model is desirable as it would reduce development time and enhance the simulation model's credibility. It is important to note that any

simplifications should be justifiable in terms of realism rather than development time and ease of model verification.

BMP research is interested in the delay time distribution which is a function of protective capacity and variability. The exact form of this function is poorly understood and an empirical description of this delay time distribution is desirable. With such an understanding non-constraint work-stations could be replaced with a "delay time" generator that would simulate the arrival of a given work-order at the constraint's buffer. As well as simplifying the manufacturing model (thus reducing development time and improving credibility) this simplification has a number of advantages. First, steady state conditions exist from the outset of the simulation (given that the buffer has been built). Second, it might be possible to apply queuing theory to produce an analytical solution to the BMP. This would aid researchers. However, the utility of this avenue of research is likely to be limited as the mathematics required is likely to be complex.

An alternative simplification, which has a similar result, is modelling the buffer size distribution. It may be possible to fit a general distribution, such as the Weibull and relate its parameters to the cv and MPC levels.

8.3.2 *Testing the Validity of Manufacturing Model Simplifications*

An interesting area of research is the application of the Goldratt (1995) thinking processes (particularly the future reality tree) to show that the proposed simplifications to the manufacturing model are valid by replicating key DBR results. For example, deterministic processing times do not model the phenomena of statistical fluctuations. Application of the thinking processes departs from the traditional approach to model building where the simplified model is embellished until it resembles "reality".

A structured approach makes it possible to rigorously assess the effect of model assumptions and pass judgement on whether proposed simplifications are restrictive or not.

8.3.3 *Processing Time Distributions*

Section 5.4.5 presented a methodology for modelling processing time distributions using the triangular distribution defined by a modal value. A number of issues need to be examined. First, the argument for the use of modal value is based on observation. There is a need to carry out experimental work to strengthen this argument adding weight to the appropriateness of calculating the release and scheduled starting time from the modal rather than the mean value.

Further as the material release time is calculated from the theoretical lead time to the constraint's buffer based on Σm_{jk}, the mean may be an appropriate measure due to the central limit theorem. This needs to be investigated.

Second, the performance of the proposed triangular distribution needs to be compared with a commonly used processing time distribution such as the gamma or the lognormal distribution. If the triangular distribution performs well, the ideas could be extended to a more complex 3 parameter distribution such as the pearson vi.

8.3.4 *Delay Time Order Statistic*

The methodology in section 5.5 derives an optimistic and pessimistic bound on the appropriate buffer size based on the delay time distribution. Calculation of the exact appropriate buffer size is not possible since the probability of a work-order's arrival the constraint buffer, p, is assumed to be constant. This is clearly a stylised assumption as the delay times are auto-correlated. More work is needed to address this issue.

The proposed methodology also assumes that the manufacturing model can achieve steady state conditions and the delay time distribution is stable. This limits the usefulness of the technique to simulation-based studies though it may be possible to extend the methodology to improving populations which model the dynamic and ever-changing nature of

manufacturing. A discussion of improving population applied to the prediction of olympic records can be found in Arnold *et al* (1992).

8.3.5 *Practical Significance*

Of critical importance to the development of implementable solutions is the issue of practical significance. What practical significance means is an open research question for the BMP. For the purposes of this research, constraint utilisation and cycle time were used to assess the practical significance of the effectiveness of each buffer sizing technique.

It seems reasonable that practical significance is driven by the economic trade-off between constraint utilisation ("an hour lost at the constraint is an hour lost for the system") and a larger than "necessary" lead time. This, as before, is another area of research.

8.4 The Role of Protective Capacity

Protective capacity is a subtle and often overlooked factor in the BMP. As a result, protective capacity is poorly understood and more research needs to be done to quantify its effect in manufacturing facilities.

While protective capacity can be intuitively understood using DBR, its function and meaning has not being rigorously investigated. Only Atwater (1991) has appeared to examined protective capacity. A number of unanswered questions remain:

- how to model the APICS (1990) definition of protective capacity discussed in section 3.4.2;
- determination of the optimal arrangement of protective capacity in a manufacturing facility; and
- determination of the relationship and economic trade-off between protective capacity, effect of variability and buffer size.

A structured understanding of protective capacity is likely to motivate the development of effective buffer sizing heuristics. Further, such an understanding would provide the economic justification for quality improvement in a manufacturing facility (coupled with buffer profiles - see chapter two). Decreasing variability or increasing non-constraint capacity is likely to increase protective capacity or excess capacity. In this case, the buffer size could be decreased or excess capacity sold for a profit (both outcomes may increase throughput). Quality improvement, however, may be expensive and any cost should be balanced with increased revenue or enhanced time-based competitiveness.

8.5 Buffer Management Technologies

In its broader context, the BMP results from the third step in Goldratt's (1990) five step processing for implementing DBR: *subordination* (Hurley *et al*, 1996). Buffer management performs the important function of ensuring that work-orders arrive at the constraint's buffer before their scheduled starting time at the constraint work-station. In this regard, buffer sizing plays an important part in *subordination*. BMP also entails other functions such as constraint-focused quality improvement and expediting tardy work-orders. Further, the BMP is also concerned with implementation issues. These techniques and methods fall under the heading of buffer management technologies.

8.5.1 *Constraint Focused Quality Improvement*

Solvable problems represent potential gains to the manufacturing facility. Removing a source of variability lessens the need for protection. As discussed in chapter two the size of the buffer will influence whether problems will be found. If few production problem are found, the buffer size could be reduced. This needs to be investigated. A discussion on constraint focused quality improvement can be found in Hurley *et al* (1996).

8.5.2 *Expediting Tardy Work-Orders*

Schragenheim *et al* (1990) divides a buffer into 3 zones; (1) expediting zone; (2) on-going improvement zone; and (3) sizing zone. A hole (or missing work-order) in the first zone warrants immediate managerial attention. In all probability, the "tardy" work-order may have to be expedited in order to avoid loss of output. Despite such an intuitive understanding, there is no structured advice for determining the length of the expediting zone. Another approach is Hurley's (1996) who addresses the problem of expediting with his "slack time to the constraint" sequencing heuristic. The heuristic, however, is currently being tested.

Another approach is to make use of the auto-correlation between delay times. A time series model (such as ARIMA) could be used to forecast delay times of work-orders. If the delay time is greater than the buffer size for a given work-order, then the work-order could be expedited thus ensuring that the output rate is maintained. Further, since the delay time distribution relates to the loadings of the non-constraint work-stations, a forecast may enable dynamic buffering - where work-orders are released into the facility a "buffer time before" based on the delay time forecasts.

8.5.3 *The "How To" of Buffer Management*

The practicalities of buffer management are rarely discussed in the literature. Chapter two identified 3 areas which need further research:

1. How to change the buffer size?
2. When the change the buffer size?
3. The means to practically implement a buffer?

How to Change the Buffer Size

Increasing the buffer size will cause the material release times to be brought forward. This is likely to coincide with the material release times for older work-order calculated from the

previous and inadequate buffer size. A one off spike in the material release rate will increase the shop-floor congestion, decreasing non-constraint protective capacity and possibly overloading non-constraint workstations. This proposes an interesting problem since inadequate protective capacity may be the very reason that the buffer size needs to be increased in the first place!

The buffer size can be gradually increased in the buffer size over time by releasing work-orders at a rate that does not overwhelm the non-constraint protective capacity. This is likely to be the most acceptable way of increasing the buffer size. Research needs to be conducted to develop structured heuristics.

When to Change the Buffer

A change in the buffer size is needed when the p^{th} delay time quartile changes by a significant amount. What exactly is meant by *significant* is dependent on the economic trade-off between due date protection and having a competitive lead time and the effort involved in changing the size of the buffer.

There are three possible ways of identifying when the buffer size needs to be changed:

1. Monitoring the delay time distribution for significant changes. Possibly a control chart for correlated data could be developed.
2. If the expediting rate is significantly larger or less than $\beta\%$.
3. Too few or too many quality problems are found.

Research needs to be conducted in order to develop simple and readily implementable heuristics.

The Means to Implement a Buffer

Research needs to be conducted on how to physically implement a buffer on the shop floor. For example, flags of different colours (yellow, orange and red) may be used to represent the progression of a hole through the sizing, on-going improvement and expediting zone.

8.6 Quality in Research

The understanding of authors such as Hurley (1996) and Huysmans (1994) about the limitations of OR techniques is based on a number of industrial case studies that are assumed to be representative. There is a need to survey (or alternatively work closely with) manufacturers to assess the perceived relevance and usefulness of OR-based solutions.

To answer questions such as these, a quality assurance model such as SERV QUAL (Zeithaml *et al*, 1990) could be applied. This could be used to develop a research agenda for BMP research that would provide the impetus for developing practical and implementable techniques that would enhance the competitiveness of manufacturers.

Chapter Nine

Conclusions

9.1 Introduction

This project has examined a number of issues associated with the BMP. In particular a practical buffer sizing solution based on fuzzy logic is researched. This chapter presents the conclusions drawn from the research.

9.2 The Buffer Management Problem (BMP)

The output rate of any manufacturing facility is limited by the constraint work-station. This realisation has a number of important ramifications for the BMP. It reduces considerably the problem of combinatorial complexity eliminating the need for complex combinatorial optimisation techniques.

A DBR solution to the BMP is not complete as determining the correct size of the buffer is still an open research question. In practice, DBR sizes buffers using intuition and experience which can be time consuming. A more structured approach to buffer sizing is desirable. This thesis has presented a fuzzy model, which is both effective and implementable.

The fuzzy approach presented has:

- addressed the needs of the mathematical unsophisticated production manager as it is intuitive and easy to understand;
- is capable of modelling the inherent vagueness associated with protective capacity and the effect of up-stream variability; and
- avoids the need to develop over-stylised and restrictive research models for dealing with analytical intractability.

A fuzzy buffer sizing model was developed in MATLAB using protective capacity and variability as input variables and the absolute buffer size as the output variable. The intuition and experience of the researcher was used to set the membership functions.

Simulation is the research methodology of choice for buffer management as it can cope with the inherent complexity of manufacturing systems. This complexity results from two basic manufacturing phenomena: dependent events and variability. Modelling the existance of dependent events on a manufacturing-wide basis with a simulation approach is time intensive and prone to errors. To address the need for improved model creditability and reduced development time, a simplified manufacturing model was developed based on a single constraint work-station and buffer.

Variability was modelled by processing time variation. Problems with commonly used processing time distributions including the lognormal and gamma were highlighted. A new methodology for modelling processing time distributions has been proposed based on the use of the mode and the triangular distribution. It was found to be effective and easy to implement producing similar distributions to more complex distributions such as the pearson vi.

By specifying the minimum amount work required in the buffer, the appropriate buffer size was determined using the p^{th} delay time quartile. This represents an optimistic, or lower bound, on the appropriate buffer size. An upper bound, or pessimistic estimate can be derived from the 95% delay time quartile. A simple measure of buffer effectiveness was developed, based on the absolute difference between the fuzzy or Umble estimate and the p^{th} delay time quartile.

This approach is limited to simulation based studies as the delay time distribution for a real manufacturing system is likely to be non-stationary moreover unlikely to reach steady state.

9.3 Results

The results of the simulation study help establish the applicability of a fuzzy solution approach to buffer sizing. The fuzzy solution is a substantial improvement over Umble's

heuristic and is practically no different to the appropriate or "optimal" result derived from the 95% delay time order statistic. The fuzzy logic solution results in a substantially smaller lead time for a given level of constraint work-station utilisation than Umble's heuristic.

Perhaps the real success of the fuzzy solution lies not with the simulation results but with the stringent nature of the fuzzy logic modelling process. As the components of the fuzzy logic model are context dependent and drawn from expert knowledge, considerable attention should be paid to the meaning and significance of input and output variables. This again, highlights the need for a sound substantive and theoretic basis which is addressed by DBR.

This research has found that the rigour of the fuzzy modelling process does enhance understanding of the factors that influence buffer size helps to shift the focus away from applying stylised solution approaches.

The BMP is a rich area for future research that has important implications for manufacturing time-based competitiveness.

9.4 Publications from this Research

1. Foote, J.L., Hurley, S.F., Evans, A.N., (1996). *Fuzzy logic applied to the management of production buffers*. Proceedings of the Third World Automation Congress, Montpellier, France, June, 1996.

2. Foote, J.L., and Hurley, S.F., (1995). *A research agenda for effective buffer management: a theory of constraints approach*. Proceedings of the Second New Zealand Postgraduate Conference for Engineering and Technology, Auckland, New Zealand, August, 1995.

3. Hurley, S.F., Evans, A.N., and Foote, J.L., (1997). *A fuzzy logic approach to the sizing of production buffers*. Submitted for inclusion in the proceedings of Decision Sciences Institute, Hawaii, March, 1997.

Other publications from this research are currently being written. These include an examination of a TOC-based research agenda for the BMP, the use of the lognormal, the gamma and the proposed triangular distributions for modelling processing times and further simplification of the manufacturing model by examining the structure of the delay time distribution and the buffer size distribution.

9.5 Contribution of this Research

In summary this thesis has presented:

- an initial fuzzy buffer sizing model that effectively sizes buffers in a simulated DBR environment - establishing the validity of a fuzzy logic approach to sizing buffers;
- a research agenda for effective buffer management based on TOC;
- a simplified manufacturing model that captures the dependency found in a DBR implementation;
- a methodology for describing realistic processing time distributions based on the triangular distribution and a single modal processing time;
- a technique for determining the lower bound and upper bound of appropriate buffer based on the delay time distribution; and
- a variety of directions for future research in buffer management.

References

1. American Production and Inventory Control Society., (1990) Cited by Atwater, J., (1991).

2. Anderson, D. and Moodie, C., (1969). Optimal buffer storage capacity in production line systems. *International Journal of Production Research, 7* (3), 233-240.

3. Arnold, B., Balakrishnan, N., and Nagaraja, H., (1992). *A first course in order statistics.* John Wiley & Sons: New York.

4. Ashcroft, S., (1989). Applying the principles of optimised production technology in a small manufacturing company, *Engineering Costs and Production Economics, 17* (1-4), 79-88.

5. Atwater, J., (1991). *The impact of protective capacity on the output of a typical unblocked flow shop*, (PhD Thesis). Georgia: University of Georgia.

6. Atwater, J. and Chakravorty, S., (1994). Does protective capacity assist managers in competing along time-based dimensions. *Production and Inventory Management Journal, 35* (3), 53-59.

7. Baker, K., (1974). *Introduction to sequencing and scheduling.* John Wiley & Sons: New York.

8. Baker, K., (1992). Tightly-coupled production systems: models, analysis, and insights. *Journal of Manufacturing Systems, 11* (6), 385-400.

9. Baker, K., Powell, S. and Pyke, D., (1993). Optimal allocation of work in assembly systems. *Management Science, 29* (1), 101-108.

10. Baral, S., (1993). Probabilistic modelling of the states of a buffer in a production flow system. *IEEE Transactions on Engineering Management, 40* (4), 381-389.

11. Blackburn, J., (1991). *Time-based competition: the next battleground in American manufacturing.* Business One Irwin: New York.

12. Blumenfeld, D., (1990). Cited by Baker, K., (1992).

13. Billesbach, T., (1991). A study of implementation of JIT in United States, *Production and Inventory Management Journal, 32* (3), 1-4.

14. Brown, M. and Harris, C. (1995). *Neurofuzzy adaptive modelling and control.* Pentice Hall: Hertfordshire.

15. Bulgak, A., (1992). Impact of quality improvement on optimal buffer designs and productivity in automatic assembly. *Journal of Manufacturing Systems, 11* (2), 124-135.

16. Bunday, B., (1984). *Basic optimisation methods.* Edward Arnold: London.

17. CACI, 1993, SimFactory, CACI Products Company, California.

18. CACI, 1996, SIMSCRIPT II.5, CACI Products Company, California.

19. Cheng, T. and Podolsky, S., (1993). *Just-in-time manufacturing: an introduction.* Chapman and Hall: New York.

20. Cheser, R., (1994). Kaizen is more than continuous improvement. *Quality Progress,* April, 23-25.

21. Chow, W., (1987). Buffer capacity analysis for sequential production lines with variable process times. *International Journal of Production* Research, *25* (8), 1183-1196.

22. Conway, R., Maxwell, W., McClain, J. and Thomas, L., (1988). The role of work-in-progress inventory in serial production lines. *Operations Research, 36* (2), 229-241.

23. Cox, E., (1994). *The fuzzy system handbook: a practitioners guide to building, using, and maintaining fuzzy systems.* Academic Press: Massachusetts.

24. Crandall, R. and Burwell, T., (1993). The effect of work-in-progress inventory levels on throughput and lead times. *Production and Inventory Management Journal, 34* (1), 6-12.

25. Crawford, K. and Cox, J., (1991). Addressing manufacturing problems through the implementation of just-in-time. *Production and Inventory Management* Journal, *32* (1), 33-36.

26. Custódio, L., Sentieiro, J., and Bispo, C., (1994). Production planning and scheduling using a fuzzy decision system. *IEEE Transactions on Robotics and Automation,* 10 (2), 160-167.

27. DeKok, A., (1990). Cited by Baker, K., (1992).

28. Evans, G., Karwowski, W., and Wilhelm, M., (1989). An introduction to fuzzy set methodologies for industrial and systems engineering. In Evans, G., Karwowski, W., and Wilhelm, M. (eds.), *Applications of fuzzy set methodologies in industrial engineering,* (pp. 3-11). Elsevier Science Publishers: Amsterdam.

29. Fawcett, S. and Pearson, J., (1991). Understanding and applying constraint management in today's manufacturing environments. *Production and Inventory Management Journal, 32* (3), 46-55.

30. Fawcett, S. and Birous, L., (1993). Just-in-time sourcing techniques: current state of adoption and performance benefits. *Production and Inventory Management Journal, 34* (1), 18-24.

31. Freud S., (1920). *Beyond the pleasure principle*, Standard Edition, Vol XVIII. Hogarth Press: London.

32. Freund, J., (1988). *Modern elementary statistics*. Prentice-Hall: New York.

33. Fry, T., (1990). Controlling input: the real key to shorter lead times. *International Journal of Logistics Management, 1* (1), 7-12.

34. Funk and Wagnalls, 1994. Ockham, Microsoft Encarta.

35. Gardiner, S., Blackstone, J. and Gardiner, L., (1993). Drum-buffer-rope and buffer management: impact on production management study and practices. *International Journal of Operations and Production Management, 13* (6), 68-78.

36. Ginting, D., (1995). *Design, implementation and simulation of a hybrid architecture for The control Of scheduling activities in an FMS*. (MTech Thesis). Palmerston North: Massey University.

37. Goldberg, D., (1989). *Genetic algorithms in search, optimisation, and machine learning*. Addison-Wesley Publishing Company: Massachusetts.

38. Goldratt, E., and Fox, R., (1986). *The race*. North River Press: New York.

39. Goldratt, E., (1990). *The haystack syndrome: sifting information out of the data ocean*. North River Press: New York.

40. Goldratt, E., (1995). *It's not luck*. North River Press: New York.

41. Goldratt, E., (1996). *Production: the TOC way*. Avraham Y. Goldratt Institute. New Haven: Connecticut

42. Hedin, S. and Russell, G., (1992). JIT implementation: interactions between production and cost accounting function. *Production and Inventory Management Journal, 33* (3), 68-73.

43. Hendricks, K., (1992). The output processes of serial production lines of exponential machines with finite buffers. *Operations Research, 40* (6), 1139-1147.

44. Hess, R. and Woolsey, G., (1980). *Applied management science: a quick and dirty approach*. Science Research Associates: Chicago.

45. Hillier, F., So, K. and Boling, R., (1993). Notes: towards characterising the optimal allocation of storage space in production line systems with variable processing times. *Management Science, 39* (1), 126-133.

46. Hillier, F. and So, K., (1991). The effect of the coefficient of variation of operations times on the allocation of storage space in production line systems. *IIE Transactions, 23* (2), 198-206.

47. Ho, Y., Eyler, M. and Chien, T., (1979). A gradient technique for general buffer storage design in a production line. *International Journal of Production Research, 17* (6), 557-580.

48. Hunt, G., (1956). Cited by Anderson *et al* (1969).

49. Hurley, S., (1993). *Development of a generic simulation-based critical resource scheduler for batch manufacturing environments.* (PhD Thesis). Liverpool: University of Liverpool.

50. Hurley, S., (1996). A practical heuristic for effective buffer management. To be published in the *International Journal of Operations and Production Management, 16* (8).

51. Hurley, S., Wright, A., and Foote, J., (1996). A theory of constraints approach to common sense continuous improvement. Chapter of a book to be published early 1996, titled "Advanced manufacturing systems: management and implementation".

52. Huysmans, J., (1994). Using the systems approach to increase management science impact on business. *Interfaces, 24* (5), 152-164.

53. Im, J., Hartman, S., and Bondi, P., (1994). How do JIT systems affect human resource management, *Production and Inventory Management Journal, 35* (1), 1-4.

54. Ishibuchi, H., Yamamoto, N., Misaki, S., and Tanaka, H., (1994). Local search algorithms for flow shop scheduling with fuzzy due-dates. *International Journal of Production Economics, 33*, 53-66.

55. Iyama, T. and Odawara, T., (1989). Two allocation methods for buffer storage in split automatic transfer lines. *International Journal of Production Research, 27* (10), 1705-1714.

56. Jafari, M. and Shanthikumar, J., (1989). Determination of optimal buffer storage capacities and optimal allocation in multistage automatic transfer lines. *IIE Transactions, 21* (2), 130-135.

57. Joo, S. and Wilhelm, W., (1993). A review of quantitative approaches in JIT manufacturing. *Production Planning and Control, 4* (3), 207-222.

58. Juran , J., (1992). *Juran on quality by design: new steps for planning quality into goods and services*. Free Press: New York.

59. Kobu, B. and Greenwood, F., (1991). Continuous improvement in a competitive global economy, *Production and Inventory Management Journal, 32* (4), 58-63.

60. Koenisberg, E., (1959). Cited by Conway *et al* (1988).

61. Koulamas, C., (1993). Single-stage and serial production line systems with unreliable machines and general probability distributions. *Decision Sciences, 24* (2), 343-369.

62. Kubat, P. and Sumita, U., (1985). Buffers and backup machines in automatic transfer lines. *International Journal of Production Research, 23* (6), 1259-1270.

63. Law, A. and Kelton, W., (1991). *Simulation modeling and analysis* (2nd ed.). McGraw-Hill Inc: New York.

64. Law, A., McComas, M., and Vincent, S., (1994). The crucial role of input modeling in successful simulation studies. *Industrial Engineering*, July, 55-59.

65. Lewis, F. and Liu, K., (1996). A paradigm for fuzzy logic control. *Automatica, 32* (2c), 167-181.

66. Lindley, D., (1987). The probability approach to the treatment of uncertainty in artificial intelligence and expert systems. *Statistical Science, 2* (1), 17-24.

67. Marks II, R. (ed.), (1994). *Fuzzy logic technology and applications*. Institute of Electrical and Electronics Engineers: New York.

68. Marsh, C. and McGowam, C., (1995). Amalgamation of knowledge and data through fuzzy modelling. Procedings of ANNES, NZ Computer Society, Dunedin, November, pp 269-273.

69. MathWorks., (1992). MATLAB v.4.2. MathWorks Inc: Massachusetts.

70. MathWorks., (1995). Fuzzy Logic Tool Box Mannual. MathWorks Inc: Massachusetts.

71. McKay, K., Safayeni, R. and Buzacott, J., (1988). Job-shop scheduling theory: what is relevant? *Interfaces, 18* (4), 84-90.

72. Mendenhall, W., Reinmuth, J., Beaver, R. and Duhan, D., (1982). *Statistics for management and economics*. Duxbury Press: Boston.

73. Minitab., (1994). Minitab v.10. Minitab Inc: Pennsylvania.

74. Monden, Y., (1983). *Toyota production system: practical approach to production management.* Industrial Engineering and Management Press: Institute of Industrial Engineers.

75. Moras, R. and Dieck, J., (1992). Industrial applications of just-in-time: lessons to be learned. *Production and Inventory Management Journal, 33* (3), 25-28.

76. Morton, T. and Pentico, D., (1993). *Heuristic scheduling systems: with applications to production systems and project management.* John Wiley & Sons: New York.

77. Mosteller, F. and Rourke, R., (1973). *Sturdy statistics.* Addison-Wesley Publishing Company: Massachusetts.

78. Muralidhar, K., Swenseth, S., and Wilson, R., (1992). Describing processing times when simulating JIT environments. *International Journal of Production Research, 30* (1), 1-11.

79. Ohmi., (1992), cited by Baker, K., (1992).

80. Park, T., (1993). A two-phase heuristic algorithm for determining buffer sizes of production lines. *International Journal of Production Research, 31* (3), 613-631.

81. Reimer, G., (1991). Material requirements planning and theory of constraints: can they coexist? A case study. *Production and Inventory Management Journal, 32* (4), 46-52.

82. Schermerhorn, J., (1996). (5th ed.). *Management for productivity.* John Wiley & Sons: New York.

83. Schonberger, J., (1986). *World class manufacturing: the lessons of simplicity applied.* Free Press: London.

84. Schragenheim, E. and Ronen, B., (1990). Drum-buffer-rope shop floor control. *Production and Inventory Management Journal, 31* (3), 18-23.

85. Schragenheim, E. and Ronen, B., 1991, Buffer management: a diagnostic tool for production control, *Production and Inventory Management Journal, 32* (2), 74-79.

86. Slany, W., Stary, C. and Dorn, J., (1992). Vague data management in production process scheduling applied to high-grade steelmaking. Proceedings of the First International Conference of Artificial Intelligence Planning Systems, June 15-19, pp 214-217.

87. Simon, H., (1969). Cited by Goldberg, D., (1989).

88. Smith, J. and Daskalaki, S., (1988). Buffer space allocation in automated assembly lines. *Operations Research, 36* (2), 343-358.

89. Spencer, M. and Cox, J., (1995). Master production scheduling development in a theory of constraints environment. *Production and Inventory Management Journal, 36* (1), 18-14.

90. So, K. and Pinault, S., (1988). Allocating buffer storage in a pull system. *International Journal of Production Research, 26* (12), 1959-1980.

91. Stalk, G., (1988). Time based competition, *Harvard Business Review*, July/August, 41-51.

92. Tompkins, J., (1990). *Winning manufacturing: the how-to book of successful manufacturing*. McGraw-Hill: New York.

93. Umble, M. and Srikanth, M., (1990). *Synchronous manufacturing: principles for world class manufacturing*. South-Western Publishing Co: Cincinnati.

94. van der Walde, E., (1991). Computer simulation in manufacturing. *Production and Inventory Management Journal, 32* (2), 80-83.

95. Wang, C., (1993). *Sense and nonsense of statistical inference: controversy, misuse, and subtlety*. Marcel Dekker Inc: New York.

96. Ward, T., Ralston, P. and Davis, J., (1992). Fuzzy logic control of aggregate production planning, *Computers and Industrial Engineering, 23* (1-4), 137-140.

97. Webber, A., (1993). What's so new about the new economy?. *Harvard Business Review*, Jan-Feb, 24-42.

98. Welch, P., (1981). *On the problem of the initial transient in steady-state simulation.* IBM Watson Research Center: New York.

99. White, R., (1993). An empirical assessment of JIT in U.S manufacturers. *Production and Inventory Management Journal, 34* (2), 38-42.

100. Winston, W., (1987). *Operations research: applications and algorithms*. Duxbury Press: California.

101. Zeithaml, V., Parasuraman, A., and Berry, L., (1990). *Delivering quality service: balancing customer perceptions and expectations*. The Free Press: New York.

102. Zimmerman, H., (1985). *Fuzzy set theory and its applications*, Kluwer Academic Publishers: Boston.

Appendices

Appendix A: Triangular Processing Time Distribution Calculations

The manufacturing operation described in section 5.4.6 is characterised by two levels of variability (cv of 0.1 and cv of 0.2) and a single modal processing time, 10 minutes.

Following the procedure detailed in Figure 5.8, the low level of cv is set equal to 0.1.

From equation 5.6, μ_o= 10.1 minutes.

The value of a is calculated using μ_o and c and equation 5.16.

$$3a + k_1 a + k_2 = 0$$

where:

$k = 20.3$

$k_1 = -60.9$

$k_2 = 290.73$

That is, $a = 7.68$ minutes.

The value for b is calculated by substituting a and c into equation 5.23:

$$b = \frac{(a+c) + \sqrt{(a+c)^2 - 4(a^2 + c^2 - ac - 18\sigma^2)}}{2}$$

That is, $b = 12.62$ minutes.

The first processing time distribution is fully specified as:

Triangular (7.68,10,12.62) minutes

Taking the square root of equation 5.9 yields the standard deviation of the above distribution as 1.01 minutes. From equation 5.24, the cv is 0.10.

The value for a is adopted as the lower bound. The second example processing time distribution has a cv of 0.2.

$\sigma = \mu_o$ x cv = 10.1 x 0.2 = 2.02 minutes

Given a = 7.68, c = 10 and σ = 2.02, b is calculated from equation 5.23, that is, b = 17.17 minutes

The second processing time distribution is fully specified as:

Triangular (7.68, 10, 17.17) minutes

As before the standard deviation and the cv of the above distribution can be calculated as 2.01 minutes and 0.20 respectively.

Appendix B: Verification of Input Probability Distributions

Section 6.5 emphasised the need for verification. Without this, researchers can have little confidence in the final results and the conclusions that are reached. This appendix details the procedures used to verify the input probability distributions that were used to generate batch size, product type and processing time of each work-order.

The chi-square test was used to confirm "post facto" whether product type and batch size were uniformly distributed. In addition, the Kolmogorov-Smirnov test was used to test whether the processing times were triangularly distributed.

Product type

Table B.1 shows the distribution of product types for 4000 work-orders.

Product Type	Count	Percentage
1	807	20.17
2	781	19.53
3	840	21.00
4	784	19.60
5	788	19.70
Total	4000	100.00

Table B.1: Tally Sheet for Product Type

The hypothesis tested is:

H_O: $p_i = 0.2$ (i = 1 to 5)

H_A: $p_i \neq 0.2$

$\chi^2_{computed} = 0.08$

$\chi^2_{(4,0.95)} = 9.49$

There is insufficient evidence to reject H_O at 5% level of significance.

Batch size

Table B.2 shows the distribution of batch sizes for 4000 work-orders.

Batch Size	Count	Percentage
10	386	9.65
11	357	8.93
12	375	9.37
13	373	9.32
14	352	8.80
15	339	8.48
16	365	9.12
17	351	8.77
18	387	9.68
19	346	8.65
20	369	9.23
Total	4000	100.00

Table B.2: Tally Sheet for Batch Size

The hypothesis tested is:

H_O: $p_i = 0.10$ (i = 1 to 10)

H_A: $p_i \neq 0.10$

$\chi^2_{computed} = 1.07$

$\chi^2_{(9,0.95)} = 16.96$

There is insufficient evidence to reject H_O at 5% level of significance.

Processing Times

60 processing times were recorded for a randomly selected product type at a given work-station. A histogram of the processing times of product 1 at work-station 1 is displayed in Figure B.1.

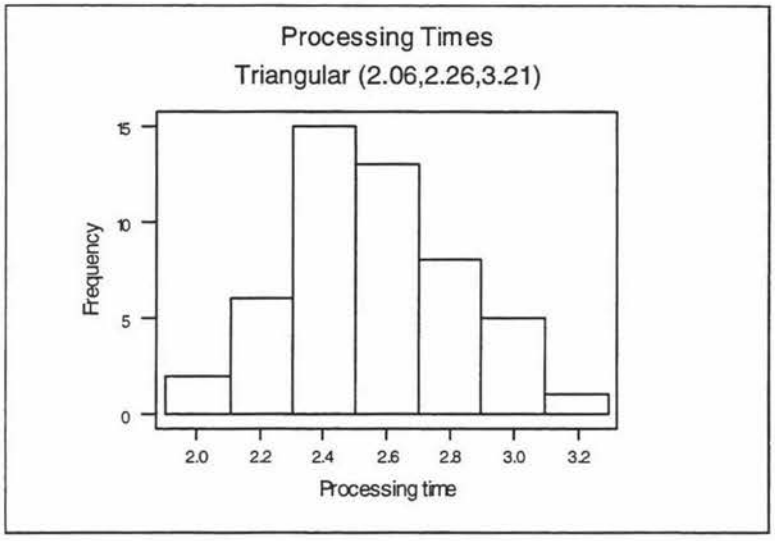


Figure B.1: Processing Times

The hypothesised distribution is a triangular (2.06, 2.26, 3.21).

The hypothesis to be tested is:

H_O: triangular (2.06, 2.26, 3.21)

H_A: not triangular (2.06, 2.26, 3.21)

The Kolmogorov-Smirnov test was used to test the hypothesised processing time distribution. It yielded a KS statistic of 0.1257. When all the parameters of the hypothesised distribution are known, the critical value of K-S statistic can easily be calculated. The critical value of KS for the 5% level of significance is 1.358. H_O can not be rejected.

Appendix C: Simulation Code

Drum Buffer Rope

Preamble

```
" defines data structures and global variables

 processes include start, release, job, final.station, steady.state.stats and stop.sim
 resources include nonconstraint and constraint

 define name3 as real, 1-dimensional array
 define epoch.length, epoch.counter and no.epoch as real variables
 define counter, no.replications, rep.counter, token and number.job as integer variables
 define c.v, mean.protective.capacity, s.buffer.size, throughput, s.time and s.constraint.processing.time
        as real variables
 define s.job.name, s.batch.size, max.batch.size, output and min.batch.size as integer variables

 permanent entities
 every product.type has
   a lead.time.to.constraint,
   a no.stages,
   a constraint.processing.time,
   and a buffer.length
   and owns a routing

 define no.stages, constraint.processing.time, lead.time.to.constraint, buffer.size, mu_d.f,
        mu_cycle.time, mu_delay.time and buffer.length as real variable

 temporary entities
  every task has
  a processing.time and a nonconstraint.id
  and belongs to a routing

  every store has
  a lateness and a epoch.number
  and belongs to a storage

 define processing.time and delay as real variables
 define nonconstraint.id and epoch.number as integer variables

 the system owns the storage
 the system owns the buffer
 the system owns the final.assembly.schedule

 temporary entities
 every job has
        a job.name,
        a release.time,  " - at the gateway resource
        a starting.time, " - at the constraint
        a batch.size,
        a delay.time,    " - actual time arrival at buffer - planned.time
        a cycle.time,    " - completion time
        a resource.id,   " - work-station id
        a nonconstraint.processing.time,
        a mode,          " - modal processing time
        a d.f, " -disruption factor
        and a p.type     " - product type
        and belongs to a final.assembly.schedule
        and belongs to a buffer

 define release.time, starting.time, delay.time, cycle.time, nonconstraint.processing.time, and d.f
        as real variables
 define batch.size, job.name, resource.id and p.type as integer variables

  define buffer as a FIFO set
  define final.assembly.schedule as a FIFO set
  define routing as a FIFO set
```

```
  define storage as a FIFO set

" - collect and collate statistics

  accumulate utilisation.nc as the mean of n.x.nonconstraint
  accumulate utilisation.c as the mean of n.x.constraint

  tally mean.delay.time as the mean, max.delay.time as the maximum and
        min.delay.time as the minimum of mu_delay.time
  tally mean.cycle.time as the mean and
        max.cycle.time as the maximum of mu_cycle.time
  tally disruption.f as the mean of mu_d.f
  tally mean.buffer.size as the mean of buffer.size

  define c.mean.e1.cu, c.mean.e2.cu, c.mean.e3.cu, c.mean.e4.cu, c.mean.e5.cu,
         c.mean.e6.cu, c.mean.e7.cu, c.mean.e8.cu, c.mean.e9.cu
         and c.mean.e10.cu as real variables

  define c.mean.e1.nu1, c.mean.e2.nu1, c.mean.e3.nu1, c.mean.e4.nu1, c.mean.e5.nu1,
         c.mean.e6.nu1, c.mean.e7.nu1, c.mean.e8.nu1, c.mean.e9.nu1 and c.mean.e10.nu1
         as real variables

  define c.mean.e1.nu2, c.mean.e2.nu2, c.mean.e3.nu2, c.mean.e4.nu2, c.mean.e5.nu2,
         c.mean.e6.nu2, c.mean.e7.nu2, c.mean.e8.nu2, c.mean.e9.nu2 and c.mean.e10.nu2
         as real variables

  define c.mean.e1.nu3, c.mean.e2.nu3, c.mean.e3.nu3, c.mean.e4.nu3, c.mean.e5.nu3,
         c.mean.e6.nu3, c.mean.e7.nu3, c.mean.e8.nu3, c.mean.e9.nu3 and c.mean.e10.nu3
         as real variables

  define c.mean.e1.nu4, c.mean.e2.nu4, c.mean.e3.nu4, c.mean.e4.nu4, c.mean.e5.nu4,
         c.mean.e6.nu4, c.mean.e7.nu4, c.mean.e8.nu4, c.mean.e9.nu4 and c.mean.e10.nu4
         as real variables

  define c.mean.e1.nu5, c.mean.e2.nu5, c.mean.e3.nu5, c.mean.e4.nu5, c.mean.e5.nu5,
         c.mean.e6.nu5, c.mean.e7.nu5, c.mean.e8.nu5, c.mean.e9.nu5 and c.mean.e10.nu5
         as real variables

 tally mean.e1.cu as the mean of c.mean.e1.cu
 tally mean.e2.cu as the mean of c.mean.e2.cu
 tally mean.e3.cu as the mean of c.mean.e3.cu
 tally mean.e4.cu as the mean of c.mean.e4.cu
 tally mean.e5.cu as the mean of c.mean.e5.cu
 tally mean.e6.cu as the mean of c.mean.e6.cu
 tally mean.e7.cu as the mean of c.mean.e7.cu
 tally mean.e8.cu as the mean of c.mean.e8.cu
 tally mean.e9.cu as the mean of c.mean.e9.cu
 tally mean.e10.cu as the mean of c.mean.e10.cu

 tally mean.e1.nu1 as the mean of c.mean.e1.nu1
 tally mean.e2.nu1 as the mean of c.mean.e2.nu1
 tally mean.e3.nu1 as the mean of c.mean.e3.nu1
 tally mean.e4.nu1 as the mean of c.mean.e4.nu1
 tally mean.e5.nu1 as the mean of c.mean.e5.nu1
 tally mean.e6.nu1 as the mean of c.mean.e6.nu1
 tally mean.e7.nu1 as the mean of c.mean.e7.nu1
 tally mean.e8.nu1 as the mean of c.mean.e8.nu1
 tally mean.e9.nu1 as the mean of c.mean.e9.nu1
 tally mean.e10.nu1 as the mean of c.mean.e10.nu1
 tally mean.e1.nu2 as the mean of c.mean.e1.nu2
 tally mean.e2.nu2 as the mean of c.mean.e2.nu2
 tally mean.e3.nu2 as the mean of c.mean.e3.nu2
 tally mean.e4.nu2 as the mean of c.mean.e4.nu2
 tally mean.e5.nu2 as the mean of c.mean.e5.nu2
 tally mean.e6.nu2 as the mean of c.mean.e6.nu2
 tally mean.e7.nu2 as the mean of c.mean.e7.nu2
 tally mean.e8.nu2 as the mean of c.mean.e8.nu2
 tally mean.e9.nu2 as the mean of c.mean.e9.nu2
 tally mean.e10.nu2 as the mean of c.mean.e10.nu2

 tally mean.e1.nu3 as the mean of c.mean.e1.nu3
```

```
tally mean.e2.nu3 as the mean of c.mean.e2.nu3
tally mean.e3.nu3 as the mean of c.mean.e3.nu3
tally mean.e4.nu3 as the mean of c.mean.e4.nu3
tally mean.e5.nu3 as the mean of c.mean.e5.nu3
tally mean.e6.nu3 as the mean of c.mean.e6.nu3
tally mean.e7.nu3 as the mean of c.mean.e7.nu3
tally mean.e8.nu3 as the mean of c.mean.e8.nu3
tally mean.e9.nu3 as the mean of c.mean.e9.nu3
tally mean.e10.nu3 as the mean of c.mean.e10.nu3
tally mean.e1.nu4 as the mean of c.mean.e1.nu4
tally mean.e2.nu4 as the mean of c.mean.e2.nu4
tally mean.e3.nu4 as the mean of c.mean.e3.nu4
tally mean.e4.nu4 as the mean of c.mean.e4.nu4
tally mean.e5.nu4 as the mean of c.mean.e5.nu4
tally mean.e6.nu4 as the mean of c.mean.e6.nu4
tally mean.e7.nu4 as the mean of c.mean.e7.nu4
tally mean.e8.nu4 as the mean of c.mean.e8.nu4
tally mean.e9.nu4 as the mean of c.mean.e9.nu4
tally mean.e10.nu4 as the mean of c.mean.e10.nu4

tally mean.e1.nu5 as the mean of c.mean.e1.nu5
tally mean.e2.nu5 as the mean of c.mean.e2.nu5
tally mean.e3.nu5 as the mean of c.mean.e3.nu5
tally mean.e4.nu5 as the mean of c.mean.e4.nu5
tally mean.e5.nu5 as the mean of c.mean.e5.nu5
tally mean.e6.nu5 as the mean of c.mean.e6.nu5
tally mean.e7.nu5 as the mean of c.mean.e7.nu5
tally mean.e8.nu5 as the mean of c.mean.e8.nu5
tally mean.e9.nu5 as the mean of c.mean.e9.nu5
tally mean.e10.nu5 as the mean of c.mean.e10.nu5

define c.mean.e1.ct, c.mean.e2.ct, c.mean.e3.ct, c.mean.e4.ct, c.mean.e5.ct,
    c.mean.e6.ct, c.mean.e7.ct, c.mean.e8.ct, c.mean.e9.ct and c.mean.e10.ct as real variables

define c.mean.e1.df, c.mean.e2.df, c.mean.e3.df, c.mean.e4.df, c.mean.e5.df,
    c.mean.e6.df, c.mean.e7.df, c.mean.e8.df, c.mean.e9.df and c.mean.e10.df as real variables

define c.mean.e1.bs, c.mean.e2.bs, c.mean.e3.bs, c.mean.e4.bs, c.mean.e5.bs,
    c.mean.e6.bs, c.mean.e7.bs, c.mean.e8.bs, c.mean.e9.bs and c.mean.e10.bs as real variables

define c.mean.e1.no.changes, c.mean.e2.no.changes, c.mean.e3.no.changes, c.mean.e4.no.changes,
    c.mean.e5.no.changes, c.mean.e6.no.changes, c.mean.e7.no.changes, c.mean.e8.no.changes,
    c.mean.e9.no.changes and c.mean.e10.no.changes as real variables

define c.mean.e1.delay, c.mean.e2.delay, c.mean.e3.delay, c.mean.e4.delay, c.mean.e5.delay,
    c.mean.e6.delay, c.mean.e7.delay, c.mean.e8.delay, c.mean.e9.delay and c.mean.e10.delay as real
    variables

define c.mean.e1.output, c.mean.e2.output, c.mean.e3.output, c.mean.e4.output, c.mean.e5.output,
    c.mean.e6.output, c.mean.e7.output, c.mean.e8.output, c.mean.e9.output and c.mean.e10.output
    as real variables

tally mean.e1.ct as the mean and v.mean.e1.ct as the variance of c.mean.e1.ct
tally mean.e2.ct as the mean and v.mean.e2.ct as the variance of c.mean.e2.ct
tally mean.e3.ct as the mean and v.mean.e3.ct as the variance of c.mean.e3.ct
tally mean.e4.ct as the mean and v.mean.e4.ct as the variance of c.mean.e4.ct
tally mean.e5.ct as the mean and v.mean.e5.ct as the variance of c.mean.e5.ct
tally mean.e6.ct as the mean and v.mean.e6.ct as the variance of c.mean.e6.ct
tally mean.e7.ct as the mean and v.mean.e7.ct as the variance of c.mean.e7.ct
tally mean.e8.ct as the mean and v.mean.e8.ct as the variance of c.mean.e8.ct
tally mean.e9.ct as the mean and v.mean.e9.ct as the variance of c.mean.e9.ct
tally mean.e10.ct as the mean and v.mean.e10.ct as the variance of c.mean.e10.ct

tally mean.e1.df as the mean and v.mean.e1.df as the variance of c.mean.e1.df
tally mean.e2.df as the mean and v.mean.e2.df as the variance of c.mean.e2.df
tally mean.e3.df as the mean and v.mean.e3.df as the variance of c.mean.e3.df
tally mean.e4.df as the mean and v.mean.e4.df as the variance of c.mean.e4.df
tally mean.e5.df as the mean and v.mean.e5.df as the variance of c.mean.e5.df
tally mean.e6.df as the mean and v.mean.e6.df as the variance of c.mean.e6.df
tally mean.e7.df as the mean and v.mean.e7.df as the variance of c.mean.e7.df
tally mean.e8.df as the mean and v.mean.e8.df as the variance of c.mean.e8.df
```

```
tally mean.e9.df as the mean and v.mean.e9.df as the variance of c.mean.e9.df
tally mean.e10.df as the mean and v.mean.e10.df as the variance of c.mean.e10.df

tally mean.e1.bs as the mean and v.mean.e1.bs as the variance of c.mean.e1.bs
tally mean.e2.bs as the mean and v.mean.e2.bs as the variance of c.mean.e2.bs
tally mean.e3.bs as the mean and v.mean.e3.bs as the variance of c.mean.e3.bs
tally mean.e4.bs as the mean and v.mean.e4.bs as the variance of c.mean.e4.bs
tally mean.e5.bs as the mean and v.mean.e5.bs as the variance of c.mean.e5.bs
tally mean.e6.bs as the mean and v.mean.e6.bs as the variance of c.mean.e6.bs
tally mean.e7.bs as the mean and v.mean.e7.bs as the variance of c.mean.e7.bs
tally mean.e8.bs as the mean and v.mean.e8.bs as the variance of c.mean.e8.bs
tally mean.e9.bs as the mean and v.mean.e9.bs as the variance of c.mean.e9.bs
tally mean.e10.bs as the mean and v.mean.e10.bs as the variance of c.mean.e10.bs

tally mean.e1.no.changes as the mean and v.mean.e1.no.changes as the variance of c.mean.e1.no.changes
tally mean.e2.no.changes as the mean and v.mean.e2.no.changes as the variance of c.mean.e2.no.changes
tally mean.e3.no.changes as the mean and v.mean.e3.no.changes as the variance of c.mean.e3.no.changes
tally mean.e4.no.changes as the mean and v.mean.e4.no.changes as the variance of c.mean.e4.no.changes
tally mean.e5.no.changes as the mean and v.mean.e5.no.changes as the variance of c.mean.e5.no.changes
tally mean.e6.no.changes as the mean and v.mean.e6.no.changes as the variance of c.mean.e6.no.changes
tally mean.e7.no.changes as the mean and v.mean.e7.no.changes as the variance of c.mean.e7.no.changes
tally mean.e8.no.changes as the mean and v.mean.e8.no.changes as the variance of c.mean.e8.no.changes
tally mean.e9.no.changes as the mean and v.mean.e9.no.changes as the variance of c.mean.e9.no.changes
tally mean.e10.no.changes as the mean and v.mean.e10.no.changes as the variance of
    c.mean.e10.no.changes

tally mean.e1.delay as the mean and v.mean.e1.delay as the variance of c.mean.e1.delay
tally mean.e2.delay as the mean and v.mean.e2.delay as the variance of c.mean.e2.delay
tally mean.e3.delay as the mean and v.mean.e3.delay as the variance of c.mean.e3.delay
tally mean.e4.delay as the mean and v.mean.e4.delay as the variance of c.mean.e4.delay
tally mean.e5.delay as the mean and v.mean.e5.delay as the variance of c.mean.e5.delay
tally mean.e6.delay as the mean and v.mean.e6.delay as the variance of c.mean.e6.delay
tally mean.e7.delay as the mean and v.mean.e7.delay as the variance of c.mean.e7.delay
tally mean.e8.delay as the mean and v.mean.e8.delay as the variance of c.mean.e8.delay
tally mean.e9.delay as the mean and v.mean.e9.delay as the variance of c.mean.e9.delay
tally mean.e10.delay as the mean and v.mean.e10.delay as the variance of c.mean.e10.delay

tally mean.e1.output as the mean and v.mean.e1.output as the variance of c.mean.e1.output
tally mean.e2.output as the mean and v.mean.e2.output as the variance of c.mean.e2.output
tally mean.e3.output as the mean and v.mean.e3.output as the variance of c.mean.e3.output
tally mean.e4.output as the mean and v.mean.e4.output as the variance of c.mean.e4.output
tally mean.e5.output as the mean and v.mean.e5.output as the variance of c.mean.e5.output
tally mean.e6.output as the mean and v.mean.e6.output as the variance of c.mean.e6.output
tally mean.e7.output as the mean and v.mean.e7.output as the variance of c.mean.e7.output
tally mean.e8.output as the mean and v.mean.e8.output as the variance of c.mean.e8.output
tally mean.e9.output as the mean and v.mean.e9.output as the variance of c.mean.e9.output
tally mean.e10.output as the mean and v.mean.e10.output as the variance of c.mean.e10.output

End
```

```
Main

  call set.up
  call initial
  for i=1 to 50,
  do
    call generate.job - " generate an initial FAS of 50 work-orders (jobs)
  loop
  activate a start now
  start simulation

end
```

```
Routine set.up

 define ind, count, index and i as integer variables
 define mu_processing, store and mu_dummy as real variables

" - create the work-stations (nonconstraints and constraint)

 n.nonconstraint=5
 create every nonconstraint

 for ind=1 to n.nonconstraint
  let u.nonconstraint(ind)=1

 n.constraint=1
 create every constraint
 u.constraint=1

 n.product.type=5
 create every product.type

" - read the input data from disk

 reserve name3(*) as 60
 open unit 1 for input, name is "data"
 use unit 1 for input
   let eof.v=0
   for index = 1 to 36, read name3(index)
   let eof.v=1
   let index = 0
   let index = index + 1
   let c.v = name3(index)
   let index = index + 1
   let mean.protective.capacity = name3(index)
   let index = index + 1
   let min.batch.size= name3(index)
   let index = index + 1
   let max.batch.size = name3(index)
   let index = index + 1
   let no.stages(1) = name3(index)
   let index = index + 1
   let no.stages(2) = name3(index)
   let index = index + 1
   let no.stages(3) = name3(index)
   let index = index + 1
   let no.stages(4) = name3(index)
   let index = index + 1
   let no.stages(5) = name3(index)
   let index = index + 1
   let constraint.processing.time(1) = name3(index)
   let index = index + 1
   let constraint.processing.time(2)  = name3(index)
   let index = index + 1
   let constraint.processing.time(3)  = name3(index)
   let index = index + 1
   let constraint.processing.time(4)  =  name3(index)
   let index = index + 1
   let constraint.processing.time(5)  = name3(index)
   let index = index + 1
   let buffer.length(1) = name3(index)
   let index = index + 1
   let buffer.length(2) = name3(index)
   let index = index + 1
   let buffer.length(3) = name3(index)
   let index = index + 1
   let buffer.length(4) = name3(index)
   let index = index + 1
   let buffer.length(5) = name3(index)
 for each product.type,
 do
   for i=1 to no.stages(product.type),
     do
```

```
      create a task " - data structure used to store processing and routing information
      let index = index + 1
      let nonconstraint.id(task) = name3(index)
      mu_processing = constraint.processing.time(product.type)*(1 - (mean.protective.capacity/100))
      mu_dummy = triang.f(0.85*mu_processing, mu_processing, 1.2*mu_processing, 10)
      processing.time(task) = mu_dummy
      file task in routing(product.type)
      add mu_dummy to lead.time.to.constraint(product.type)
    loop
  loop
  close unit 1

 print 2 lines thus
Drum-Buffer-Rope Simulation of a Job-Shop
-----------------------------------------
  skip 1 line
  print 1 line thus
Experimental conditions
  print 1 line with mean.protective.capacity thus
Mean protective capacity level: ** %
  print 1 line with c.v thus
Coefficient of variation: *.**
  print 1 line with max.batch.size thus
Maximum batch size: **
  print 1 line with min.batch.size thus
Minimum batch size: **
  skip 1 line
  print 1 line thus
Protection
   print 1 line with buffer.length(product.type) thus
Time buffer of **.** minutes
   skip 1 line

return

end
```

```
process start

 define index, index1 and index2 as integer variables

 let epoch.length = 7500
 let no.epoch = 10
 let no.replications = 10
 let rep.counter = 0
 let epoch.counter = 0
 let token = 0
 let counter = 0
 let time.after.warm.up = 4000000000

 for index = 1 to 120000,
  do
   activate a release now
  loop

 dummy = 5
 while dummy <> 4
 do
  if time.v*24*60 > 500*24*60*7
   call reset.statistics
   let output = 0
   for index1 = 1 to no.replications
    do
     for index2 = 1 to no.epoch
      do
       add 1 to counter
       activate a steady.state.stats in (counter*epoch.length) minutes
      loop
    loop
    let dummy = 4
  else
   wait 10 minutes
  always
 loop

end
```

```
Process release

  define job.to.be.released as an integer variable
  define min.release.time and wait.to.release as real variables

  call generate.job " - top up FAS

  for each job in the final.assembly.schedule, with release.time(job) >= 0
  compute min.release.time as the minimum of release.time(job)

  for each job in the final.assembly.schedule, with release.time(job) = min.release.time
    find job.to.be.released = the first job
    remove this job from the final.assembly.schedule
    let wait.to.release = release.time(job.to.be.released) - 24*60*time.v
    activate this job called job.to.be.released in wait.to.release minutes

end
```

```
Process job

 define k, k1, k2, constant, constant1, constant2, a, b, mode job.work, delay and processing as real
        variables

 if job.name(job) <> 0
  let product.type = p.type(job)
  for each task in routing(product.type),
   do
    let resource.id = nonconstraint.id(task)
    let nonconstraint.processing.time = processing.time(task)
    let mode = processing.time(task)
    let mean.nonconstraint.processing.time = mode/0.9999
    request 1 nonconstraint(resource.id)
    let constant = 3*mean.nonconstraint.processing.time - mode
    let constant1 = - 3*constant
    let constant2 = 3*(constant**2) - 9*constant*mean.nonconstraint.processing.time
                  + 9*(mean.nonconstraint.processing.time**2)
                   - 18*((0.01*mean.nonconstraint.processing.time)**2)
    let a = (- constant1 - sqrt.f( (constant1)**2 - 4*3*constant2))/(2*3)
    let k = (a + mode)
    let k2 = (c.v*mean.nonconstraint.processing.time)**2
    let  k1 = 4*(a**2 + mode**2 - a*mode - 18*k2)
    let b = (+k + sqrt.f(k**2 - k1))/2
    let processing = triang.f(a, mode, b, resource.id)
    work (processing*batch.size(job)) minutes
    relinquish 1 nonconstraint(resource.id)
   loop
 else
  wait (starting.time(job) - buffer.length(p.type(job))) minutes
  request 1 unit of constraint
  work (buffer.length(p.type(job)) + constraint.processing.time(p.type(job))*batch.size(job)) minutes
  relinquish 1 unit of constraint
  reset the totals of n.x.constraint
 always

 if job.name(job) <>0,
  file job in the buffer
  let job.work = (constraint.processing.time(p.type(job))*batch.size(job))
  let buffer.size = s.buffer.size + job.work
  let s.buffer.size = buffer.size
 always

 let delay = (time.v*24*60 - release.time(job)) - lead.time.to.constraint(p.type(job))
 if delay >= 0,
  let delay.time(job) = delay
 else
  let delay.time(job) = 0
 always

 let mu_delay.time = delay.time(job)
 if epoch.counter ge 1,
  create a store
  let epoch.number(store) = epoch.counter
  let lateness(store) = mu_delay.time
  file store in storage
 always

 let d.f(job)= delay.time*constraint.processing.time(p.type(job))*batch.size(job)
 let mu_d.f = d.f(job)
 if job.name(job) =1
  activate a final.station now
 always

 suspend

end
```

```
Process final.station

 define min.starting.time and job.work as real variables
 define dummy and job.to.be.processed as an integer variable

 let dummy=5
 while dummy <> 4,
  do
    if u.constraint = 1 and the buffer is not empty
     for each job in the buffer,
     compute min.starting.time as the minimum of starting.time(job)
     for each job in the buffer,
     with starting.time(job) = min.starting.time
     find job.to.be.processed = the first job
     if time.v*24*60 ne starting.time(job.to.be.processed),
      add 1 to no.changes.order
     always
     remove job.to.be.processed from the buffer
     let job.work = batch.size(job.to.be.processed)*constraint.processing.time(p.type(job.to.be.processed))
     request 1 unit of constraint
     let buffer.size = s.buffer.size - job.work
     let s.buffer.size = buffer.size
     work (batch.size(job.to.be.processed)*constraint.processing.time(p.type(job.to.be.processed)))
          minutes
     relinquish 1 unit of constraint
     let cycle.time(job.to.be.processed) = 24*60*time.v - release.time(job.to.be.processed)
     let mu_cycle.time = cycle.time(job.to.be.processed)
     add 1 to output
    else
     wait 0.01 minutes
    always
 loop

end
```

```
Process steady.state.stats

 add 1 to epoch.counter
 if epoch.counter = 1,
  let c.mean.e1.ct = mean.cycle.time
  let c.mean.e1.df = disruption.f
  let c.mean.e1.bs = mean.buffer.size
  let c.mean.e1.no.changes = no.changes.order
  let c.mean.e1.output = output
  let c.mean.e1.delay = mean.delay.time
  let c.mean.e1.cu = utilisation.c
  let c.mean.e1.nu1 = utilisation.nc(1)
  let c.mean.e1.nu2 = utilisation.nc(2)
  let c.mean.e1.nu3 = utilisation.nc(3)
  let c.mean.e1.nu4 = utilisation.nc(4)
  let c.mean.e1.nu5 = utilisation.nc(5)
 always

 if epoch.counter = 2,
  let c.mean.e2.ct = mean.cycle.time
  let c.mean.e2.df = disruption.f
  let c.mean.e2.bs = mean.buffer.size
  let c.mean.e2.no.changes = no.changes.order
  let c.mean.e2.output = output
  let c.mean.e2.delay = mean.delay.time
  let c.mean.e2.cu = utilisation.c
  let c.mean.e2.nu1 = utilisation.nc(1)
  let c.mean.e2.nu2 = utilisation.nc(2)
  let c.mean.e2.nu3 = utilisation.nc(3)
  let c.mean.e2.nu4 = utilisation.nc(4)
  let c.mean.e2.nu5 = utilisation.nc(5)
 always

 if epoch.counter = 3,
  let c.mean.e3.ct = mean.cycle.time
  let c.mean.e3.df = disruption.f
  let c.mean.e3.bs = mean.buffer.size
  let c.mean.e3.no.changes = no.changes.order
  let c.mean.e3.output = output
  let c.mean.e3.delay = mean.delay.time
  let c.mean.e3.cu = utilisation.c
  let c.mean.e3.nu1 = utilisation.nc(1)
  let c.mean.e3.nu2 = utilisation.nc(2)
  let c.mean.e3.nu3 = utilisation.nc(3)
  let c.mean.e3.nu4 = utilisation.nc(4)
  let c.mean.e3.nu5 = utilisation.nc(5)
 always

 if epoch.counter = 4,
  let c.mean.e4.ct = mean.cycle.time
  let c.mean.e4.df = disruption.f
  let c.mean.e4.bs = mean.buffer.size
  let c.mean.e4.no.changes = no.changes.order
  let c.mean.e4.output = output
  let c.mean.e4.delay = mean.delay.time
  let c.mean.e4.cu = utilisation.c
  let c.mean.e4.nu1 = utilisation.nc(1)
  let c.mean.e4.nu2 = utilisation.nc(2)
  let c.mean.e4.nu3 = utilisation.nc(3)
  let c.mean.e4.nu4 = utilisation.nc(4)
  let c.mean.e4.nu5 = utilisation.nc(5)
 always

 if epoch.counter = 5,
  let c.mean.e5.ct = mean.cycle.time
  let c.mean.e5.df = disruption.f
  let c.mean.e5.bs = mean.buffer.size
  let c.mean.e5.no.changes = no.changes.order
  let c.mean.e5.output = output
  let c.mean.e5.delay = mean.delay.time
  let c.mean.e5.cu = utilisation.c
```

```
 let c.mean.e5.nu1 = utilisation.nc(1)
 let c.mean.e5.nu2 = utilisation.nc(2)
 let c.mean.e5.nu3 = utilisation.nc(3)
 let c.mean.e5.nu4 = utilisation.nc(4)
 let c.mean.e5.nu5 = utilisation.nc(5)
always

if epoch.counter =6,
 let c.mean.e6.ct = mean.cycle.time
 let c.mean.e6.df = disruption.f
 let c.mean.e6.bs = mean.buffer.size
 let c.mean.e6.no.changes = no.changes.order
 let c.mean.e6.output = output
 let c.mean.e6.delay = mean.delay.time
 let c.mean.e6.cu = utilisation.c
 let c.mean.e6.nu1 = utilisation.nc(1)
 let c.mean.e6.nu2 = utilisation.nc(2)
 let c.mean.e6.nu3 = utilisation.nc(3)
 let c.mean.e6.nu4 = utilisation.nc(4)
 let c.mean.e6.nu5 = utilisation.nc(5)
always

 if epoch.counter = 7,
 let c.mean.e7.ct = mean.cycle.time
 let c.mean.e7.df = disruption.f
 let c.mean.e7.bs = mean.buffer.size
 let c.mean.e7.no.changes = no.changes.order
 let c.mean.e7.output = output
 let c.mean.e7.delay = mean.delay.time
 let c.mean.e7.cu = utilisation.c
 let c.mean.e7.nu1 = utilisation.nc(1)
 let c.mean.e7.nu2 = utilisation.nc(2)
 let c.mean.e7.nu3 = utilisation.nc(3)
 let c.mean.e7.nu4 = utilisation.nc(4)
 let c.mean.e7.nu5 = utilisation.nc(5)
always

if epoch.counter = 8,
 let c.mean.e8.ct = mean.cycle.time
 let c.mean.e8.df = disruption.f
 let c.mean.e8.bs = mean.buffer.size
 let c.mean.e8.no.changes = no.changes.order
 let c.mean.e8.output = output
 let c.mean.e8.delay = mean.delay.time
 let c.mean.e8.cu = utilisation.c
 let c.mean.e8.nu1 = utilisation.nc(1)
 let c.mean.e8.nu2 = utilisation.nc(2)
 let c.mean.e8.nu3 = utilisation.nc(3)
 let c.mean.e8.nu4 = utilisation.nc(4)
 let c.mean.e8.nu5 = utilisation.nc(5)
always

if epoch.counter = 9,
 let c.mean.e9.ct = mean.cycle.time
 let c.mean.e9.df = disruption.f
 let c.mean.e9.bs = mean.buffer.size
 let c.mean.e9.no.changes = no.changes.order
 let c.mean.e9.output = output
 let c.mean.e9.delay = mean.delay.time
 let c.mean.e9.cu = utilisation.c
 let c.mean.e9.nu1 = utilisation.nc(1)
 let c.mean.e9.nu2 = utilisation.nc(2)
 let c.mean.e9.nu3 = utilisation.nc(3)
 let c.mean.e9.nu4 = utilisation.nc(4)
 let c.mean.e9.nu5 = utilisation.nc(5)
always

 if epoch.counter = 10,
 let c.mean.e10.ct = mean.cycle.time
 let c.mean.e10.df = disruption.f
 let c.mean.e10.bs = mean.buffer.size
```

```
  let c.mean.e10.no.changes = no.changes.order
  let c.mean.e10.output = output
  let c.mean.e10.delay = mean.delay.time
  let c.mean.e10.cu = utilisation.c
  let c.mean.e10.nu1 = utilisation.nc(1)
  let c.mean.e10.nu2 = utilisation.nc(2)
  let c.mean.e10.nu3 = utilisation.nc(3)
  let c.mean.e10.nu4 = utilisation.nc(4)
  let c.mean.e10.nu5 = utilisation.nc(5)
 always

 if epoch.counter = no.epoch,
  add 1 to rep.counter
  let epoch.counter = 0
 always

 call reset.statistics
 let output = 0
 let no.changes.order = 0

if rep.counter = no.replications
 activate a stop.sim now
always

end
```

```
Routine initial

 " creates a dummy job that builds the buffer to its required size

 create a job
 let p.type(job) =3 " arbitrary
 let batch.size(job) =15 " arbitrary
 let s.batch.size = batch.size(job)
 let starting.time(job) = 10000 " arbitrary
 let s.time = starting.time(job)
 let release.time(job) = 0
 let s.constraint.processing.time = constraint.processing.time(p.type(job))
 let job.name(job) = 0
 let s.job.name = job.name(job)
 file job in final.assembly.schedule

return

end
```

```
Routine generate.job

 create a job
 let job.name(job) = s.job.name + 1
 let s.job.name = job.name(job)
 let p.type(job) = randi.f(1,n.product.type,8)
 let batch.size(job) = randi.f(min.batch.size, max.batch.size,9)
 let starting.time(job) = s.time + s.constraint.processing.time*s.batch.size
 let s.time = starting.time(job)
 let s.constraint.processing.time = constraint.processing.time(p.type(job))
 let s.batch.size = batch.size(job)
 let release.time(job) = starting.time(job) - buffer.length(p.type(job)) -
    lead.time.to.constraint(p.type(job))*batch.size(job)
 file job in final.assembly.schedule

return

end
```

```
Routine reset.statistics

 define index as an integer variable

 for index =1 to n.nonconstraint
  do
   reset the totals of n.x.nonconstraint(index)
  loop
  reset the totals of n.x.constraint
  reset the totals of mu_cycle.time
  reset the totals of mu_delay.time
  reset the totals of mu_d.f
  reset the totals of buffer.size

return

end
```

```
Process stop.sim

  open unit 3 for output, name is "out.txt"
  use unit 3 for output

  print 1 line thus
Collected Statistics by Epoch
  skip 1 line
  print 1 line thus
Experimental conditions:
  print 1 line with c.v thus
cv = ****.****
  print 1 line with mean.protective.capacity thus
MPC = ****.***
  skip 1 line
  print 1 line with buffer.length thus
Planned buffer length = ******.****
  print 1 line thus
Statistics for Epoch 1: Means
  print 1 line with mean.e1.output thus
Output = *****.**
  print 1 line with mean.e1.ct thus
Cycle time = ******.**
  print 1 line with mean.e1.delay thus
Delay time = ******.**
  print 1 line with mean.e1.df thus
Disruption factor = ******.**
  print 1 line with mean.e1.cu thus
Contraint utilisation = *****.****
  print 1 line with mean.e1.nu1 thus
Work-station 1 utilisation = *****.***
  print 1 line with mean.e1.nu2 thus
Work-station 2 utilisation = *****.***
  print 1 line with mean.e1.nu3 thus
Work-station 3 utilisation = *****.***
  print 1 line with mean.e1.nu4 thus
Work-station 4 utilisation = *****.***
  print 1 line with mean.e1.nu5 thus
Work-station 5 utilisation = *****.***
  print 1 line with mean.e1.bs thus
Buffer size = ******.**
skip 1 line
  print 1 line thus
Statistics for Epoch 2: Means
  print 1 line with mean.e2.output thus
Output = *****.**
  print 1 line with mean.e2.ct thus
Cycle time = ******.**
  print 1 line with mean.e2.delay thus
Delay time = ******.**
  print 1 line with mean.e2.df thus
Disruption factor = ******.**
  print 1 line with mean.e2.cu thus
Contraint utilisation = *****.****
  print 1 line with mean.e2.nu1 thus
Work-station 1 utilisation = *****.***
  print 1 line with mean.e2.nu2 thus
Work-station 2 utilisation = *****.***
  print 1 line with mean.e2.nu3 thus
Work-station 3 utilisation = *****.***
  print 1 line with mean.e2.nu4 thus
Work-station 4 utilisation = *****.***
  print 1 line with mean.e2.nu5 thus
Work-station 5 utilisation = *****.***
  print 1 line with mean.e2.bs thus
Buffer size = ******.**
 skip 1 line
  print 1 line thus
Statistics for Epoch 3: Means
  print 1 line with mean.e3.output thus
Output = *****.**
```

```
 print 1 line with mean.e3.ct thus
Cycle time = ******.**
 print 1 line with mean.e3.delay thus
Delay time = ******.**
 print 1 line with mean.e3.df thus
Disruption factor = ******.**
 print 1 line with mean.e3.cu thus
Contraint utilisation = *****.****
 print 1 line with mean.e3.nu1 thus
Work-station 1 utilisation = *****.***
 print 1 line with mean.e3.nu2 thus
Work-station 2 utilisation = *****.***
 print 1 line with mean.e3.nu3 thus
Work-station 3 utilisation = *****.***
 print 1 line with mean.e3.nu4 thus
Work-station 4 utilisation = *****.***
 print 1 line with mean.e3.nu5 thus
Work-station 5 utilisation = *****.***
 print 1 line with mean.e3.bs thus
Buffer size = ******.**
 print 1 line with mean.e3.no.changes thus
Number of changes to schedule = *****
 skip 1 line
 print 1 line thus
Statistics for Epoch 4: means
 print 1 line with mean.e4.output thus
Output = *****.**
 print 1 line with mean.e4.ct thus
Cycle time = ******.**
 print 1 line with mean.e4.delay thus
Delay time = ******.**
 print 1 line with mean.e4.df thus
Disruption factor = ******.**
 print 1 line with mean.e4.cu thus
Contraint utilisation = *****.****
 print 1 line with mean.e4.nu1 thus
Work-station 1 utilisation = *****.***
 print 1 line with mean.e4.nu2 thus
Work-station 2 utilisation = *****.***
 print 1 line with mean.e4.nu3 thus
Work-station 3 utilisation = *****.***
 print 1 line with mean.e4.nu4 thus
Work-station 4 utilisation = *****.***
 print 1 line with mean.e4.nu5 thus
Work-station 5 utilisation = *****.***
 print 1 line with mean.e4.bs thus
Buffer size = ******.**
 print 1 line with mean.e4.no.changes thus
Number of changes to schedule = *****
 skip 1 line
 print 1 line thus
Statistics for Epoch 5: means
 print 1 line with mean.e5.output thus
Output = *****.**
 print 1 line with mean.e5.ct thus
Cycle time = ******.**
 print 1 line with mean.e5.delay thus
Delay time = ******.**
 print 1 line with mean.e5.df thus
Disruption factor = ******.**
 print 1 line with mean.e5.cu thus
Contraint utilisation = *****.****
 print 1 line with mean.e5.nu1 thus
Work-station 1 utilisation = *****.***
 print 1 line with mean.e5.nu2 thus
Work-station 2 utilisation = *****.***
 print 1 line with mean.e5.nu3 thus
Work-station 3 utilisation = *****.***
 print 1 line with mean.e5.nu4 thus
Work-station 4 utilisation = *****.***
 print 1 line with mean.e5.nu5 thus
```

```
Work-station 5 utilisation = *****.***
  print 1 line with mean.e5.bs thus
Buffer size = ******.**
  print 1 line with mean.e5.no.changes thus
Number of changes to schedule = *****
  skip 1 line
  print 1 line thus
Statistics for Epoch 6: means
  print 1 line with mean.e6.output thus
Output = *****.**
  print 1 line with mean.e6.ct thus
Cycle time = ******.**
  print 1 line with mean.e6.delay thus
Delay time = ******.**
  print 1 line with mean.e6.df thus
Disruption factor = ******.**
  print 1 line with mean.e6.cu thus
Contraint utilisation = *****.****
  print 1 line with mean.e6.nu1 thus
Work-station 1 utilisation = *****.***
  print 1 line with mean.e6.nu2 thus
Work-station 2 utilisation = *****.***
  print 1 line with mean.e6.nu3 thus
Work-station 3 utilisation = *****.***
  print 1 line with mean.e6.nu4 thus
Work-station 4 utilisation = *****.***
  print 1 line with mean.e6.nu5 thus
Work-station 5 utilisation = *****.***
  print 1 line with mean.e6.bs thus
Buffer size = ******.**
  print 1 line with mean.e6.no.changes thus
Number of changes to schedule = *****
  skip 1 line
  print 1 line thus
Statistics for Epoch 7: means
  print 1 line with mean.e7.output thus
Output = *****.**
  print 1 line with mean.e7.ct thus
Cycle time = ******.**
  print 1 line with mean.e7.delay thus
Delay time = ******.**
  print 1 line with mean.e7.df thus
Disruption factor = ******.**
  print 1 line with mean.e7.cu thus
Contraint utilisation = *****.****
  print 1 line with mean.e7.nu1 thus
Work-station 1 utilisation = *****.***
  print 1 line with mean.e7.nu2 thus
Work-station 2 utilisation = *****.***
  print 1 line with mean.e7.nu3 thus
Work-station 3 utilisation = *****.***
  print 1 line with mean.e7.nu4 thus
Work-station 4 utilisation = *****.***
  print 1 line with mean.e7.nu5 thus
Work-station 5 utilisation = *****.***
  print 1 line with mean.e7.bs thus
Buffer size = ******.**
  print 1 line with mean.e7.no.changes thus
Number of changes to schedule = *****
  skip 1 line
  print 1 line thus
Statistics for Epoch 8: means
  print 1 line with mean.e8.output thus
Output = *****.**
  print 1 line with mean.e8.ct thus
Cycle time = ******.**
  print 1 line with mean.e8.delay thus
Delay time = ******.**
  print 1 line with mean.e8.df thus
Disruption factor = ******.**
  print 1 line with mean.e8.cu thus
```

```
Contraint utilisation = *****.****
 print 1 line with mean.e8.nu1 thus
Work-station 1 utilisation = *****.***
 print 1 line with mean.e8.nu2 thus
Work-station 2 utilisation = *****.***
 print 1 line with mean.e8.nu3 thus
Work-station 3 utilisation = *****.***
 print 1 line with mean.e8.nu4 thus
Work-station 4 utilisation = *****.***
 print 1 line with mean.e8.nu5 thus
Work-station 5 utilisation = *****.***
 print 1 line with mean.e8.bs thus
Buffer size = ******.**
 print 1 line with mean.e8.no.changes thus
Number of changes to schedule = *****
 skip 1 line
 print 1 line thus
Statistics for Epoch 9: means
 print 1 line with mean.e9.output thus
Output = *****.**
 print 1 line with mean.e9.ct thus
Cycle time = ******.**
 print 1 line with mean.e9.delay thus
Delay time = ******.**
 print 1 line with mean.e9.df thus
Disruption factor = ******.**
 print 1 line with mean.e9.cu thus
Contraint utilisation = *****.****
 print 1 line with mean.e9.nu1 thus
Work-station 1 utilisation = *****.***
 print 1 line with mean.e9.nu2 thus
Work-station 2 utilisation = *****.***
 print 1 line with mean.e9.nu3 thus
Work-station 3 utilisation = *****.***
 print 1 line with mean.e9.nu4 thus
Work-station 4 utilisation = *****.***
 print 1 line with mean.e9.nu5 thus
Work-station 5 utilisation = *****.***
 print 1 line with mean.e9.bs thus
Buffer size = ******.**
 print 1 line with mean.e9.no.changes thus
Number of changes to schedule = *****
 skip 1 line
 print 1 line thus
Statistics for Epoch 10: means
 print 1 line with mean.e10.output thus
Output = *****.**
 print 1 line with mean.e10.ct thus
Cycle time = ******.**
 print 1 line with mean.e10.delay thus
Delay time = ******.**
 print 1 line with mean.e10.df thus
Disruption factor = ******.**
 print 1 line with mean.e10.cu thus
Contraint utilisation = *****.****
 print 1 line with mean.e10.nu1 thus
Work-station 1 utilisation = *****.***
 print 1 line with mean.e10.nu2 thus
Work-station 2 utilisation = *****.***
 print 1 line with mean.e10.nu3 thus
Work-station 3 utilisation = *****.***
 print 1 line with mean.e10.nu4 thus
Work-station 4 utilisation = *****.***
 print 1 line with mean.e10.nu5 thus
Work-station 5 utilisation = *****.***
 print 1 line with mean.e10.bs thus
Buffer size = ******.**
 print 1 line with mean.e10.no.changes thus
Number of changes to schedule = *****
 close unit 3
```

```
 open unit 2 for output, name is "delay"
 use unit 2 for output
  for each store in storage,
   print 1 line with lateness(store) and epoch.number(store) thus
*****.****          *****.****
 close unit 2

stop

end
```

MRP Driven Job-Shop

```
Preamble

 processes include start, job, steady.state.stats, release and stop.sim
 resources include workstation

 define name3 as real, 1-dimensional array .
 define epoch.length, epoch.counter and no.epoch as real variables
 define counter, no.replications, rep.counter, token and number.job as integer variables
 define c.v, mean.protective.capacity, s.buffer.size, s.time and s.constraint.processing.time
        as real variables
 define s.job.name, s.batch.size, max.batch.size, output and min.batch.size as integer variables

 permanent entities
 every product.type has
   a no.stages,
   a constraint.processing.time,
   and owns a routing

  temporary entities
  every task has
         a processing.time, a nonconstraint.id and a next.nonconstraint.id
         and belongs to a routing

  define no.stages, constraint.processing.time, processing.time and mu_cycle.time as real variables
  define nonconstraint.id and epoch.number as integer variables

 temporary entities
 every job has
        a job.name,
        a batch.size,
        a release.time,
        a cycle.time,    " - completion time
        a resource.id,
        a nonconstraint.processing.time,
        a mode,
        and a p.type

 define cycle.time, release.time and nonconstraint.processing.time as real variables
 define batch.size, job.name and p.type as integer variables

 define routing as a LIFO set

 accumulate utilisation as the mean of n.x.workstation
 tally mean.cycle.time as the mean and max.cycle.time as the maximum of mu_cycle.time

 define c.mean.e1.ct, c.mean.e2.ct, c.mean.e3.ct, c.mean.e4.ct, c.mean.e5.ct,
        c.mean.e6.ct, c.mean.e7.ct, c.mean.e8.ct, c.mean.e9.ct and c.mean.e10.ct as real variables

  tally mean.e1.ct as the mean and v.mean.e1.ct as the variance of c.mean.e1.ct
  tally mean.e2.ct as the mean and v.mean.e2.ct as the variance of c.mean.e2.ct
  tally mean.e3.ct as the mean and v.mean.e3.ct as the variance of c.mean.e3.ct
  tally mean.e4.ct as the mean and v.mean.e4.ct as the variance of c.mean.e4.ct
  tally mean.e5.ct as the mean and v.mean.e5.ct as the variance of c.mean.e5.ct
  tally mean.e6.ct as the mean and v.mean.e6.ct as the variance of c.mean.e6.ct
  tally mean.e7.ct as the mean and v.mean.e7.ct as the variance of c.mean.e7.ct
  tally mean.e8.ct as the mean and v.mean.e8.ct as the variance of c.mean.e8.ct
  tally mean.e9.ct as the mean and v.mean.e9.ct as the variance of c.mean.e9.ct
  tally mean.e10.ct as the mean and v.mean.e10.ct as the variance of c.mean.e10.ct

End
```

```
Main

  call set.up
  activate a start now
  start simulation

end
```

```
Routine set.up

define ind, count, index and i as integer variables
define mu_processing, store and mu_dummy as real variables

 n.nonconstraint=5
 create every nonconstraint

 for ind=1 to n.nonconstraint
  let u.nonconstraint(ind)=1

 n.constraint=1
 create every constraint
 u.constraint=1

 n.product.type=5
 create every product.type

 reserve name3(*) as 60
 open unit 1 for input, name is "data"
 use unit 1 for input
   let eof.v=0
   for index = 1 to 36, read name3(index)
   let eof.v=1
   let index = 0
   let index = index + 1
   let c.v = name3(index)
   let index = index + 1
   let mean.protective.capacity = name3(index)
   let index = index + 1
   let min.batch.size= name3(index)
   let index = index + 1
   let max.batch.size = name3(index)
   let index = index + 1
   let no.stages(1) = name3(index)
   let index = index + 1
   let no.stages(2) = name3(index)
   let index = index + 1
   let no.stages(3) = name3(index)
   let index = index + 1
   let no.stages(4) = name3(index)
   let index = index + 1
   let no.stages(5) = name3(index)
   let index = index + 1
   let constraint.processing.time(1) = name3(index)
   let index = index + 1
   let constraint.processing.time(2)  = name3(index)
   let index = index + 1
   let constraint.processing.time(3)  = name3(index)
   let index = index + 1
   let constraint.processing.time(4)  =  name3(index)
   let index = index + 1
   let constraint.processing.time(5)  = name3(index)
   let index = index + 1
   let buffer.length(1) = name3(index)
   let index = index + 1
   let buffer.length(2) = name3(index)
   let index = index + 1
   let buffer.length(3) = name3(index)
   let index = index + 1
   let buffer.length(4) = name3(index)
   let index = index + 1
   let buffer.length(5) = name3(index)
 for each product.type,
 do
    for i=1 to no.stages(product.type),
     do
       create a task
       let index = index + 1
       let nonconstraint.id(task) = name3(index)
       mu_processing = constraint.processing.time(product.type)*(1 - (mean.protective.capacity/100))
```

```
      mu_dummy = triang.f(0.85*mu_processing, mu_processing, 1.2*mu_processing, 10)
      processing.time(task) = mu_dummy
      file task in routing(product.type)
      add mu_dummy to lead.time.to.constraint(product.type)
    loop
  loop
  close unit 1

 print 2 lines thus
Drum-Buffer-Rope Simulation of a Job-Shop
-----------------------------------------
  skip 1 line
  print 1 line thus
Experimental conditions
  print 1 line with mean.protective.capacity thus
Mean protective capacity level: ** %
  print 1 line with c.v thus
Coefficient of variation: *.**
  print 1 line with max.batch.size thus
Maximum batch size: **
  print 1 line with min.batch.size thus
Minimum batch size: **
  skip 1 line
  print 1 line thus
Protection
   print 1 line with buffer.length(product.type) thus
Time buffer of **.** minutes
   skip 1 line

return

end
```

```
process start

 let epoch.length = 75000
 let no.epoch = 10
 let no.replications = 20
 let rep.counter = 0
 let epoch.counter = 0
 let token = 0
 let counter = 0
 let time.after.warm.up = 40000000

 activate a release now
 dummy = 3
 while dummy <> 4
  do

  if time.v*24*60 > 3000*1000,
  call reset.statistics
  for index1 = 1 to no.replications
  do
   for index2 = 1 to no.epoch
    do
     add 1 to counter
     activate a steady.state.stats in (counter*epoch.length) minutes
    loop
   loop
   let dummy = 4
  else
   wait 10 minutes
  always
 loop

end
```

```
process release

 define random and randoma as integer variables
 let rdummy = 1

 while rdummy <> 0
 do
  if n.x.workstation(1) = 0,
   create a job
   let job.name(job) = s.job.name + 1
   let batch.size(job) = randi.f(min.batch.size, max.batch.size, 9)
   let random = randi.f(1,3,8)
   if random =1
    let p.type(job) = 1
   always
   if random = 2
    let p.type(job) = 3
   always
   if random = 3
    let p.type(job) = 5
   always
   let s.job.name = job.name(job)
   activate this job now
  always

  if n.x.workstation(2) = 0,
   create a job
   let job.name(job) = s.job.name + 1
   let batch.size(job) = randi.f(min.batch.size, max.batch.size, 9)
   let randoma = randi.f(1,2,8)
   if randoma =1
    let p.type(job) = 2
   else
    let p.type(job) = 4
   always
   let s.job.name = job.name(job)
   activate this job now
  always
 wait 0.1 minute
 loop
end
```

```
process job

define processing as real variables
define guess, dummy1, dummy2, resource.id, mode, constant, constant1 and constant2 as real variables

 let guess = 0
 let product.type = p.type(job)

 for each task in routing(product.type),
 do
  let resource.id = nonconstraint.id(task)
  let next.resource.id = next.nonconstraint.id(task)
  let nonconstraint.processing.time = processing.time(task)
  let mode = processing.time(task)
  let mean.nonconstraint.processing.time = mode/0.99
  dummy1 = 0

  while dummy1 <> 1
  do
   if n.q.workstation(resource.id) < 10
     request 1 workstation(resource.id)
     let dummy1 = 1
     if guess = 0
      let release.time(job) = time.v*24*60
     always
     guess = 1
   else
     wait 1 minute
   always
  loop

  let constant = 3*mean.nonconstraint.processing.time - mode
  let constant1 = - 3*constant
  let constant2 = 3*(constant**2) - 9*constant*mean.nonconstraint.processing.time
              + 9*(mean.nonconstraint.processing.time**2)
              - 18*((0.1*mean.nonconstraint.processing.time)**2)
  let a = (- constant1 - sqrt.f( (constant1)**2 - 4*3*constant2))/(2*3)
  let k = (a + mode)
  let k2 = (c.v*mean.nonconstraint.processing.time)**2
  let  k1 = 4*(a**2 + mode**2 - a*mode - 18*k2)
  let b = (+k + sqrt.f(k**2 - k1))/2
  let processing = triang.f(a, mode, b, resource.id)
  work (processing*batch.size(job)) minutes

  if next.resource.id = 0
     relinquish 1 workstation(resource.id)
  else

    let dummy2 = 0
    while dummy2 <> 1
     do
       if next.resource.id <> 0 and n.q.workstation(next.resource.id) < 10
         relinquish 1 workstation(resource.id)
         let dummy2 = 1
       else
         wait 1 minute
       always
     loop
  always

  loop

  let cycle.time(job) = time.v*24*60 - release.time(job)
  let mu_cycle.time = cycle.time(job)
  add 1 to output


end
```

```
Process steady.state.stats

 add 1 to epoch.counter
 if epoch.counter = 1,
  let c.mean.e1.ct = mean.cycle.time
 always

 if epoch.counter = 2,
  let c.mean.e2.ct = mean.cycle.time
 always

 if epoch.counter = 3,
  let c.mean.e3.ct = mean.cycle.time
 always

 if epoch.counter = 4,
  let c.mean.e4.ct = mean.cycle.time
 always

 if epoch.counter = 5,
  let c.mean.e5.ct = mean.cycle.time
 always

 if epoch.counter =6,
  let c.mean.e6.ct = mean.cycle.time
 always

 if epoch.counter = 7,
  let c.mean.e7.ct = mean.cycle.time
 always

 if epoch.counter = 8,
  let c.mean.e8.ct = mean.cycle.time
 always

 if epoch.counter = 9,
  let c.mean.e9.ct = mean.cycle.time
 always

 if epoch.counter = 10,
  let c.mean.e10.ct = mean.cycle.time
 always

 if epoch.counter = no.epoch,
  add 1 to rep.counter
  let epoch.counter = 0
 always

 call reset.statistics
 let output = 0
 let no.changes.order = 0

 if rep.counter = no.replications
  activate a stop.sim now
 always

end
```

```
process stop.sim

  open unit 3 for output, name is "out.txt"
  use unit 3 for output
  print 1 line thus
Collected Statistics by Epoch
  skip 1 line
  print 1 line thus
Experimental conditions:
  print 1 line with c.v thus
cv = ****.****
  print 1 line with mean.protective.capacity thus
MPC = ****.***
  skip 1 line
  print 1 line thus
Statistics for Epoch 1: Mean and Variance
  print 1 line with mean.e1.ct and v.mean.e1.ct thus
Cycle time = ******.** (******.**)
  print 1 line thus
Statistics for Epoch 2: Mean and Variance
  print 1 line with mean.e2.ct and v.mean.e2.ct thus
Cycle time = ******.** (******.**)
  skip 1 line
  print 1 line thus
Statistics for Epoch 3: Mean and Variance
  print 1 line with mean.e3.ct and v.mean.e3.ct thus
Cycle time = ******.** (******.**)
  skip 1 line
  print 1 line thus
Statistics for Epoch 4: Mean and Variance
  print 1 line with mean.e4.ct and v.mean.e4.ct thus
Cycle time = ******.** (******.**)
  skip 1 line
  print 1 line thus
Statistics for Epoch 5: Mean and Variance
  print 1 line with mean.e5.ct and v.mean.e5.ct thus
Cycle time = ******.** (******.**)
  skip 1 line
  print 1 line thus
Statistics for Epoch 6: Mean and Variance
  print 1 line with mean.e6.ct and v.mean.e6.ct thus
Cycle time = ******.** (******.**)
  skip 1 line
  print 1 line thus
Statistics for Epoch 7: Mean and Variance
  print 1 line with mean.e7.ct and v.mean.e7.ct thus
Cycle time = ******.** (******.**)
  skip 1 line
  print 1 line thus
Statistics for Epoch 8: Mean and Variance
  print 1 line with mean.e8.ct and v.mean.e8.ct thus
Cycle time = ******.** (******.**)
  skip 1 line
  print 1 line thus
Statistics for Epoch 9: Mean and Variance
  print 1 line with mean.e9.ct and v.mean.e9.ct thus
Cycle time = ******.** (******.**)
  skip 1 line
  print 1 line thus
Statistics for Epoch 10: Mean and Variance
  print 1 line with mean.e10.ct and v.mean.e9.ct thus
Cycle time = ******.** (******.**)
  close unit 3

stop

end
```