Analysis and Design of Closed Loop Manufacturing Systems

by

Loren M. Werner

B.S. Operations Research, US Air Force Academy, CO(USA), 1999

Submitted to the Sloan School of Management
in partial fulfillment of the requirements for the degree of

Master of Science in Operations Research

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2001

© Massachusetts Institute of Technology 2001. All rights reserved.

Author .....
Sloan School of Management
June 4, 2001

Certified by .....
Stanley B. Gershwin
Senior Research Scientist, Department of Mechanical Engineering
Thesis Supervisor

Accepted by .....
James B. Orlin
Co-director, Operations Research Center

Analysis and Design of Closed Loop Manufacturing Systems

by

Loren M. Werner

Submitted to the Sloan School of Management
on June 4, 2001, in partial fulfillment of the
requirements for the degree of
Master of Science in Operations Research

Abstract

This thesis introduces an approximation method for evaluating the performance of closed loop manufacturing systems with unreliable machines and finite buffers. The method involves transforming an arbitrary loop into one without thresholds and then evaluating the transformed loop using a new set of decomposition equations. It is more accurate than existing methods and is effective for a wider range of cases. The convergence reliability, and speed of the method are also discussed. In addition, observations are made on the behavior of closed loop systems under various conditions. Finally, the method is used in a case study to determine the in-process inventory required to meet a specified production rate for a system operating according to a CONWIP control policy.

Thesis Supervisor: Stanley B. Gershwin
Title: Senior Research Scientist, Department of Mechanical Engineering

Acknowledgments

This thesis is dedicated to my parents. Without their support I would never have made it this far.

Special thanks to Dr Stan Gershwin for all of his teaching and guidance and to the Lean Aerospace Initiative for their financial support. I would also like to thank Youichi Nonaka for helping me with the case study, and Meow Seen Yong, Young Jae Jang, Nicola Maggio, and Francis de Vericourt for keeping things interesting.

Thanks to Guy Barth, Jeff Moffitt, and Steve Clark for their help and friendship.

Last, but certainly not least, I would like to thank Ban, Lily, and Daisy for making it all worthwhile.

Executive Summary

A closed-loop production system or *loop* is a system in which a constant amount of material flows through a set of work stations and storage buffers alternately in a fixed sequence, and when the material leaves the last buffer, it reenters the first machine. Figure 0-1 represents a K-machine loop. This type of system occurs frequently in manufacturing. Processes that utilize pallets or fixtures can be viewed as loops since the number of pallets/fixtures that are in the system remains constant. Similarly, control policies such as CONWIP and Kanban create conceptual loops by imposing a limit on the number of parts that can be in the system an any given time.

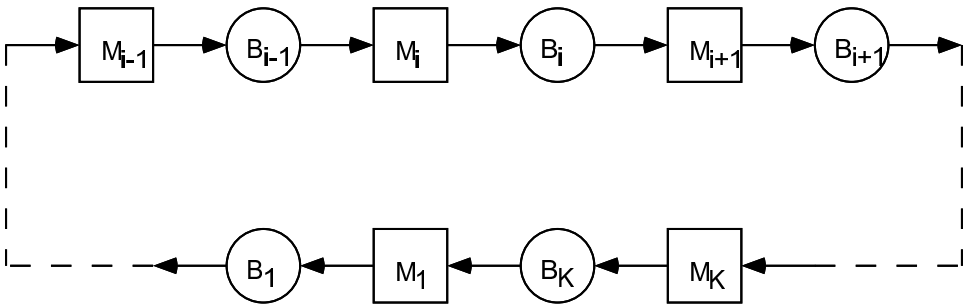

Figure 0-1: Illustration of a closed-loop production system

Performance measures such as average production rate and the distribution of in-process inventory cannot be expressed in closed form. Simulation provides accurate results for these quantities, but it can be time consuming. Some analytical methods have been developed, but they can only be used in a limited class of cases. The purpose of this thesis is to present a more versatile analytical method for evaluating these performance measures of closed-loop production systems.

We describe our model of a manufacturing system and review the existing techniques for evaluating open production lines. We then explain why the characteristics of loops make these techniques inadequate and propose a solution method designed specifically for closed-loop systems.

The accuracy, speed, and convergence reliability of the method are discussed in detail. In addition, several observations are made on the behavior of closed-loop systems.

We conclude with a case study involving the production of a network connection device. The company operates the production facility according to a CONWIP control policy. They anticipate an increase in demand that will require additional capacity in the factory and are considering the option of additional overtime or purchasing additional machinery. We use the loop evaluation method to determine the in-process

inventory required to meet the specified demand rate for each of the two options.

Contents

1 Introduction **15**
- 1.1 Problem Statement . . . 16
- 1.2 Literature Review . . . 16
- 1.3 Overview . . . 16

2 Closed-Loop Production Systems **19**
- 2.1 Basic Model . . . 19
- 2.2 Transfer Line Decomposition Techniques . . . 19
- 2.3 Special Characteristics of Closed-Loop Systems . . . 20
- 2.4 Thresholds . . . 21
- 2.5 Loop Transformation . . . 23
  - 2.5.1 Special Cases . . . 25
- 2.6 Fixed Population Considerations . . . 25
  - 2.6.1 Simultaneous Blocking and Starvation . . . 26

3 Loop Decomposition **27**
- 3.1 The Building Block Parameters . . . 27
- 3.2 Decomposition Equations . . . 28

4 Implementing the Loop Transformation and Decomposition **29**
- 4.1 The Transformation Algorithm . . . 29
- 4.2 The Decomposition Algorithm . . . 29

5 Performance of the Method **31**
- 5.1 Cases Studied . . . 31
- 5.2 Accuracy . . . 31
  - 5.2.1 Population Range . . . 31
  - 5.2.2 Three-machine Loops . . . 32
  - 5.2.3 Six-machine Loops . . . 32
  - 5.2.4 Ten-machine Loops . . . 32
  - 5.2.5 The Batman Effect . . . 41
- 5.3 Convergence Reliability . . . 42

5.4 Speed . . . 43

6 Observations on Loop Behavior **45**
6.1 Flatness . . . 45
6.1.1 Transfer Line Flatness . . . 45
6.1.2 Near Flatness and Non-Flatness . . . 47
6.2 Changes in Loop Parameters . . . 48
6.2.1 Machine Parameters . . . 51
6.2.2 Buffer Size . . . 53

7 Applying the Method **57**
7.1 Background . . . 57
7.1.1 The Network Device . . . 57
7.1.2 Production Process . . . 57
7.2 Problem Statement . . . 58
7.2.1 Objective . . . 58
7.2.2 Limiting the Scope . . . 58
7.3 Transforming the Data . . . 59
7.3.1 Original Data . . . 59
7.3.2 Parallel Machines . . . 59
7.3.3 Overtime . . . 60
7.4 Reducing the System . . . 60
7.4.1 Identifying the Bottlenecks . . . 60
7.4.2 Zero-buffer Approximation . . . 61
7.4.3 Processing Times . . . 63
7.5 Solution Methodology . . . 64
7.6 Conclusions . . . 65
7.6.1 Case Conclusions . . . 65
7.6.2 Needed Research . . . 66

8 Conclusions and Future Work **69**

A Three-Machine Loop Parameters **71**

B Six-Machine Loop Parameters **75**

C Ten-machine Loop Parameters **81**

List of Figures

0-1 Illustration of a closed-loop production system 7

1-1 Illustration of a closed-loop production system 15

2-1 Example of a Loop with Thresholds 22
2-2 Illustration of a transformed loop 24

5-1 Throughput error (three-machine loops) 33
5-2 Average buffer level error (three-machine loops) 34
5-3 Throughput error versus population (Loop 3.2) 35
5-4 Throughput error (six-machine loops) 36
5-5 Average buffer level error (six-machine loops) 37
5-6 Average buffer level error (six-machine loops) 38
5-7 Throughput error (ten-machine loops) 39
5-8 Average buffer level error (ten-machine loops) 40
5-9 Illustration of the batman effect for a three-machine loop 41
5-10 Illustration of the batman effect for a six-machine loop 42
5-11 Computation time versus loop size 43

6-1 Example of loop with transfer line flatness 45
6-2 Analytical throughput as a function of population 46
6-3 Analytical average buffer level as a function of population 47
6-4 Loop 6.4 ($\sigma_{buffers} = 7.97$) 48
6-5 Loop 6.1 ($\sigma_{buffers} = 11.80$) 49
6-6 Loop 6.2 ($\sigma_{buffers} = 19.12$) 50
6-7 Average Throughput as a Function of r_1 51
6-8 Average Buffer Level as a Function of r_1 52
6-9 Average Throughput as a Function of r 53
6-10 Average Throughput as a Function of e 54
6-11 Analytical Throughput as a Function of Buffer Size 55

7-1 Illustration of reduced system 62
7-2 Representation of a three-parameter machine as two two-parameter machines 63

7-3 Illustration of final model 64
7-4 Production Rate as a Function of CONWIP Limit (Max Overtime) . 65
7-5 Production Rate as a Function of CONWIP Limit (Buy Additional Machine) 66

List of Tables

I Parameters of reduced loop (max overtime) 62
II Parameters of reduced loop (additional machine) 62

I Loop 3.1 71
II Loop 3.2 71
III Loop 3.3 71
IV Loop 3.4 72
V Loop 3.5 72
VI Loop 3.6 72
VII Loop 3.7 72
VIII Loop 3.8 72
IX Loop 3.9 73
X Loop 3.10 73
XI Loop 3.11 73
XII Loop 3.12 73
XIII Loop 3.13 73
XIV Loop 3.14 74
XV Loop 3.15 74

I Loop 6.1 75
II Loop 6.2 75
III Loop 6.3 76
IV Loop 6.4 76
V Loop 6.5 76
VI Loop 6.6 76
VII Loop 6.7 77
VIII Loop 6.8 77
IX Loop 6.9 77
X Loop 6.10 77
XI Loop 6.11 78
XII Loop 6.12 78
XIII Loop 6.13 78
XIV Loop 6.14 78

XV Loop 6.15 79

I Loop 10.1 81
II Loop 10.2 82
III Loop 10.3 82
IV Loop 10.4 83
V Loop 10.5 83
VI Loop 10.6 84
VII Loop 10.7 84
VIII Loop 10.8 85
IX Loop 10.9 85
X Loop 10.10 86
XI Loop 10.11 86
XII Loop 10.12 87
XIII Loop 10.13 87
XIV Loop 10.14 88
XV Loop 10.15 88

Chapter 1

Introduction

A closed-loop production system or *loop* is a system in which a constant amount of material flows through a set of work stations and storage buffers alternately in a fixed sequence, and when the material leaves the last buffer, it reenters the first machine. Figure 1-1 represents a K-machine loop. This type of system occurs frequently in manufacturing. Processes that utilize pallets or fixtures can be viewed as loops since the number of pallets/fixtures that are in the system remains constant. Similarly, control policies such as CONWIP and Kanban create conceptual loops by imposing a limit on the number of parts that can be in the system an any given time.

In a typical loop, raw parts entering the system are placed onto pallets at a loading station. The pallet-part assembly then visits the fixed sequence of machines and buffers. After receiving all of the operations, the finished part is removed from the pallet at an unloading station and exits the system. The empty pallet returns to the loading station.

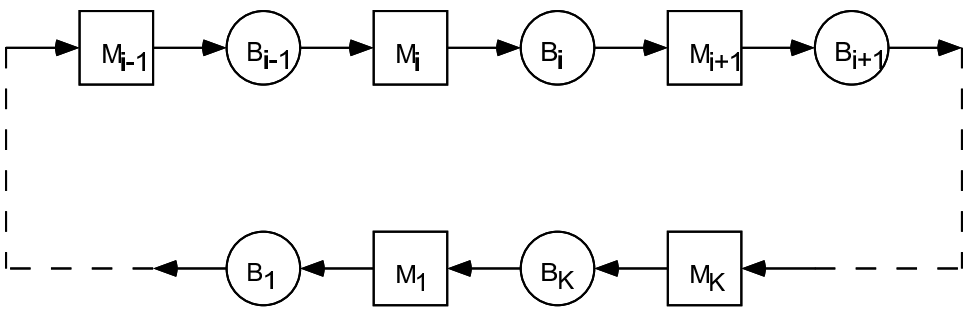

Figure 1-1: Illustration of a closed-loop production system

1.1 Problem Statement

Performance measures such as average production rate and the distribution of in-process inventory cannot be expressed in closed form. Simulation provides accurate results for these quantities, but it can be time consuming. Some analytical methods have been developed, but they can only be used in a limited class of cases. (See Section 1.2.) The purpose of this thesis is to present a more versatile analytical method for evaluating these performance measures of closed-loop production systems. Specifically, we are concerned with closed-loop systems where the number of parts in the system is larger than the number of machines and the size of the largest buffer and less than the total buffer capacity minus the the number of machines and the size of the largest buffer.

1.2 Literature Review

Compared to open transfer lines, relatively little work has been done on closed-loop production systems with finite buffers and unreliable machines. Onvural and Perros [OP90] demonstrated that the production rate of a closed-loop system is a function of the number of parts in the system. In addition, they showed that the throughput versus population curve is symmetric when blocking occurs before service and processing time is exponential. To avoid the complication that finite buffers create in closed-loop systems, Akyildiz [Aky88] approximated production rate by reducing the population and evaluating the same system with infinite buffers. Bouhchouch, Frein, and Dallery [BFD92] used a closed-loop queuing network with finite capacities to model a closed-loop system with finite buffers. For a more detailed listing of previous work dealing with closed-loop systems, see [Mag00].

The first analytical method for evaluating the performance of closed-loop systems with finite buffers and unreliable machines was proposed by Frein-Commault-Dallery in 1994 [FCD94]. This method is an extension of the decomposition method developed by [Ger87a]. It is important to note that this method does not account for the correlation between number of parts in each buffer and the probability of blocking and starvation. As a result, the method is only accurate for large loops.

[Mag00] presents a new decomposition method which does account for the correlation between population and the probability of blocking and starvation. However, the model is more complex and is not practical for loops with more than three machines.

1.3 Overview

Chapter 2 gives a detailed description of the closed-loop systems and introduces our transformation method and Chapter 3 discusses the loop decomposition. The algorithms for implementing the transformation and decomposition are discussed in

Chapter 4. In Chapter 5 we comment on the performance of the method. Chapter 6 conveys some of the observations we made on the behavior of loops. In Chapter 7 we describe a real-world application of the loop evaluation method. Chapter 8 concludes the paper and offers some possible areas for future research.

Chapter 2

Closed-Loop Production Systems

We first describe our model of a manufacturing system. Next, we review the existing techniques for evaluating open production lines and explain why the characteristics of loops make these techniques inadequate. Finally, we propose a transformation and decomposition method designed specifically for closed-loop systems.

2.1 Basic Model

Throughout this analysis, we extend the deterministic processing time model presented in [Ger94] to closed-loop systems. More specifically, we use the version of the model presented by [TM98], which allows machines to fail in more than one mode. This feature is critical to our method and is discussed in Section 2.6. Processing times for all machines are assumed to be deterministic and identical. In addition, all *operational* (i.e. not failed) machines start their operations at the same time. For simplicity, we scale the processing time to one time unit. Parts in the machines are ignored, as is travel time between machines. Machine failure and repair times are geometrically distributed.

M_i refers to Machine i. B_i is its downstream buffer and has capacity N_i. A machine is *blocked* if its downstream buffer is full and *starved* if its upstream buffer is empty. When M_i is *working* (operational and neither blocked nor starved) it has a probability p_{ij} of failing in mode j in one time unit. If M_i is down in mode j, it is repaired in a given time unit with probability r_{ij}. By convention, machine failures and repairs take place at the beginnings of time units and changes in buffer levels occur at the ends of time units.

2.2 Transfer Line Decomposition Techniques

Although it is possible to obtain an exact analytical solution for a two-machine line, the problem becomes intractable for longer lines. However, accurate decomposition

methods have been developed for evaluating long transfer lines [Ger94]. These methods decompose a K-machine transfer line into $K-1$ two-machine lines or *building blocks*. In each building block $L(i)$, the buffer $B(i)$ corresponds to B_i in the original transfer line. The upstream machine $M^u(i)$ represents the collective behavior of the line upstream of B_i and the downstream machine $M^d(i)$ represents the behavior downstream.

To an observer sitting in $B(i)$, $M^u(i)$ appears to be down when M_i is either down or starved by some upstream machine. $M^u(i)$ is said to have *real* failure modes corresponding to those of M_i and *virtual* failure modes corresponding to each of the upstream machines [TM98]. Likewise, $M^d(i)$ has *real* failure modes corresponding to those of M_{i+1} and *virtual* failure modes corresponding to each of the downstream machines.

In order to estimate the system performance, we must find values of the virtual failure probabilities, $p^u_{k,j}(i)$ and $p^d_{k,j}(i)$, the parameters of $M^u(i)$ and $M^d(i)$. The probability $p^u_{k,j}(i)$ is the observer's estimate of the probability of machine $M^u(i)$ failing in mode (k,j). Although the observer does not know this, mode (k,j) corresponds to mode j of machine M_k. If M_k is upstream of M_i, this is a virtual mode. If $k=i$ this is a real mode. (Similarly for $M^d(i)$.) The concept of range of starvation eliminates the ambiguity of "upstream" and "downstream" in a loop[1].

The goal of the decomposition method is to find the parameters of $M^u(i)$ and $M^d(i)$ such that the flow of parts through $B(i)$ mimics that through B_i. Accomplishing this for all building blocks gives approximate values for average throughput and buffer levels in the original transfer line.

2.3 Special Characteristics of Closed-Loop Systems

In a transfer line, blocking and starvation can propagate throughout the entire system. If the first machine fails, it is possible for all of the downstream machines to become starved. Similarly, if the last machine fails, all upstream machines can become blocked.

This is not the case in loops. Whether or not a machine can be starved or blocked by the failure of another machine depends on the number of parts in the system and the total buffer space between the two machines. For ease of notation, we define all subscripts to be modulo K. In particular, we define the set of integers (i,j) as:

$$(i,j) = \begin{cases} (i, i+1, ..., j) & \text{if } i < j \\ (i, i+1, ..., K, 1, ..., j) & \text{if } i > j \end{cases} \tag{2.1}$$

We define N^p to be the total number of parts in the system and $\Psi(v,w)$ as the total buffer capacity between M_v and M_w in the direction of flow [MMGT00]. More

[1] See Section 2.6.

formally,

$$\Psi(v,w) = \begin{cases} \sum_{z=v}^{w-1} N_z & \text{if } v \neq w \\ 0 & \text{if } v = w \end{cases} \tag{2.2}$$

Note that if $v \neq w$, the total buffer space is given by

$$N^{total} = \Psi(v,w) + \Psi(w,v) \tag{2.3}$$

If $N^p < \Psi(v,w)$, then the failure of M_w can never cause M_v to become blocked because there are not enough parts in the system to fill all buffers between M_v and M_w simultaneously. Conversely, if $N^p > \Psi(v,w)$, M_w cannot starve M_v.

2.4 Thresholds

The issue of blocking and starvation is more complicated still. In some cases, whether or not a machine can ever be starved or blocked by the failure of a specific other machine depends on the number of parts in an adjacent buffer. This is the concept of *thresholds* introduced in [MMGT00].

Consider the case where 7 parts are traveling through a three-machine loop with buffers of size 5 (see Figure 2-1). If M_2 fails, parts begin to build up in B_1 and eventually M_1 becomes blocked. However, we know that M_1 cannot be blocked if the number of parts in its upstream buffer, B_3, remains greater than 2. This would mean that the number of parts in its downstream buffer must be less than 5 since there are only 7 parts in the system. Conversely, we know that if the number of parts in B_3 remains less than 2 then the number of parts in B_2 must be greater than zero and M_3 cannot become starved.[2] Therefore, we say that B_3 has a threshold of 2.

In general, we define the threshold $l_k(i)$ to be the maximum level of B_i such that all buffers between M_{i+1} and M_k can become full at the same time. Alternately, we can think of $l_k(i)$ as the maximum level of B_i such that the failure of M_k can cause M_{i+1} to become blocked. It is calculated as:

$$l_k(i) = N^p - \Psi(i+1,k) \tag{2.4}$$

Note that $l_k(i)$ can assume values ranging from less than zero to greater than N_i depending on the population and buffer sizes. Values less than zero indicate that the failure of M_k cannot cause M_{i+1} to become blocked, regardless of the number of parts in B_i. Conversely, values greater than N_i indicate that M_{i+1} can become blocked by M_k independently of the level of B_i. Special cases arise when $l_k(i)$ is equal to zero or N_i which are discussed in Section 2.6.

[2] Maggio shows that blocking thresholds and starving thresholds are the same in [MMGT00]. In this paper, we determine the thresholds from the perspective of blocking.

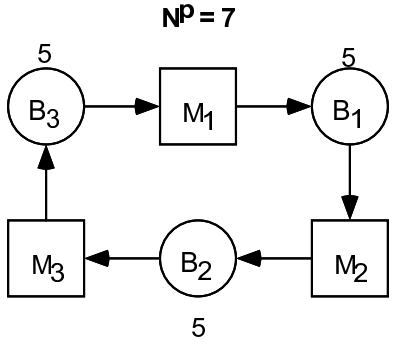

Figure 2-1: Example of a Loop with Thresholds

[MMGT00] is limited to cases in which $0 < l_k(i) < N_i$. Maggio introduces a new, more detailed, building block and a new set of decomposition equations. This approach is very accurate and has been implemented for three-machine loops with certain restrictions on population and buffer sizes. However, Maggio's building block can only take one threshold into account. To extend the method to larger loops, the building block would have to become very complex in order to deal with multiple thresholds. Because the propagation of starvation and blocking is limited, the virtual failure probabilities, $p^u_{k,j}(i)$ and $p^d_{k,j}(i)$ may be zero. Whether they are zero or not depends on the level of the buffer $B(i)$, ie, whether it is above or below a threshold.

Assume that machine M_{i+1} can be blocked by machine M_k. This means that $M^d(i)$ has a virtual failure of type $p^d_{kj}(i)$ (where j indicates one of the failure modes of machine M_k, $j = 1, \ldots, F_k$). Because of the population constraint, if buffer B_i has too many parts, the remaining parts cannot fill all the buffers between M_{i+1} and M_k. Therefore, if the level of the buffer is greater than a threshold, indicated by $l^d_k(i)$, then a failure on M_k cannot cause blocking on machine M_{i+1} so $p^d_{kj}(i)$ is equal to 0. Generalizing, machine M_k could produce blocking on machine M_{i+1} and therefore it could affect $M^d(i)$ only if the level of buffer $B(i)$ is lower than or equal to the threshold indicated by $l^d_k(i)$.

A similar argument holds for the starvation of M_i. Suppose that machine M_i can be starved by machine M_z. This means that $M^u(i)$ has a virtual failure probability $p^u_{zj}(i)$ (where j indicates one of the failure modes of the real machine M_z, $j = 1, \ldots, F_z$). If buffer B_i has too few parts, the remaining parts are too many to be contained in all the buffers space between M_{i+1} and M_z. Therefore M_i cannot be starved due to a failure of machine M_z, since the buffers between M_z and M_i cannot all be empty. More precisely buffer B_{i-1} cannot be empty due to machine M_z being failed for a time long enough to make all the buffers between M_z and M_i empty. This observation leads us to say that $p^u_{zj}(i) = 0$ if the number of parts in $B(i)$ is less

than a specific threshold indicated by $l_z^u(i)$. This symbol represents the threshold for the upstream machine $M^u(i)$ related to a failure of machine M_z. Machine M_z could produce starvation at machine M_i (and therefore it could affect $M^u(i)$) only if the level of buffer $B(i)$ is greater than or equal to $l_z^u(i)$.

The threshold $l_k^d(i)$ represents the largest number of parts in buffer B_i that allows all the buffers between M_{i+1} and M_k to be full at the same time. In that case all the buffers between M_k and M_i would be empty. Similarly, threshold $l_k^u(i)$ represents the smallest number of parts that allows all the buffers between M_k and M_i to be empty. Therefore, since machine M_{i+1} cannot be blocked by M_i and M_i cannot be starved by M_{i+1} (due to the assumption on the number of parts we made), the two thresholds represent the same number of parts and therefore the same condition: buffers between M_{i+1} and M_k full and buffers between M_k and M_i empty. Thus we can simplify the notation:

$$l_k^u(i) = l_k^d(i) = l_k(i) \tag{2.5}$$

Let $n(t)$ indicate the number of parts in buffer $B(i)$ at time t. Then,

- if $n(t) < l_k(i)$ then $M^u(i)$ cannot be down in virtual mode due to a failure j of machine M_k. Therefore: $p_{kj}^u(i) = 0$.

  In other words, if too few parts are in $B(i)$, the total capacity of the buffers between M_{i+1} and M_k is not enough to contain all the remaining parts. In this case M_i cannot be starved by M_k.

- if $n(t) \geq l_k(i)$, the probability $p_{kj}^u(i)$ is an unknown constant failure probability that has to be determined.

- if $n(t) > l_k(i)$ then $M^d(i)$ cannot be down in virtual mode due to a failure j of machine M_k. Therefore: $p_{kj}^d(i) = 0$

  In other words if too many parts are in $B(i)$, then all the buffers between M_{i+1} and M_k cannot be filled. Therefore in this case M_{i+1} cannot be blocked by M_k.

- if $n(t) \leq l_k(i)$, the probability $p_{kj}^d(i)$ is an unknown constant failure probability that has to be determined.

The value of the threshold does not depend on the failure type but only on the buffer capacities between machines M_{i+1} and M_k and on the number of parts in the system, i.e. N^p.

2.5 Loop Transformation

It is possible to eliminate the complications in the two-machine building blocks due to thresholds by using a transformation procedure. The transformation allows us to

evaluate much larger loops for a wider range of population levels and buffer sizes than is possible using the method presented in [MMGT00].

Consider a two-machine line with unreliable machines and a buffer of size 10. Into the buffer, we insert an infinitely fast (i.e. with an operation time equal to zero) and perfectly reliable machine. The result is a three-machine line with two buffers having a combined buffer capacity of 10. The performance of the new line is identical to the original.

In reality, infinitely fast machines do not exist. Our model is based on an operation time of one, not zero. However, this concept provides the motivation behind our transformation method. Instead of dealing with the thresholds directly, we transform the loop into one without thresholds that behaves in almost the same way. The resulting loop is relatively easy to analyze.

Consider again the three-machine loop with buffers of size 5 and population 7. Into each of the three buffers, we insert a perfectly reliable machine so that the buffer of size 5 is replaced by an upstream buffer of size 3 and a downstream buffer of size 2 (see Figure 2-2)). The performance of this new six-machine loop is approximately the same as the original three-machine loop, but we have eliminated all thresholds between zero and N_i.

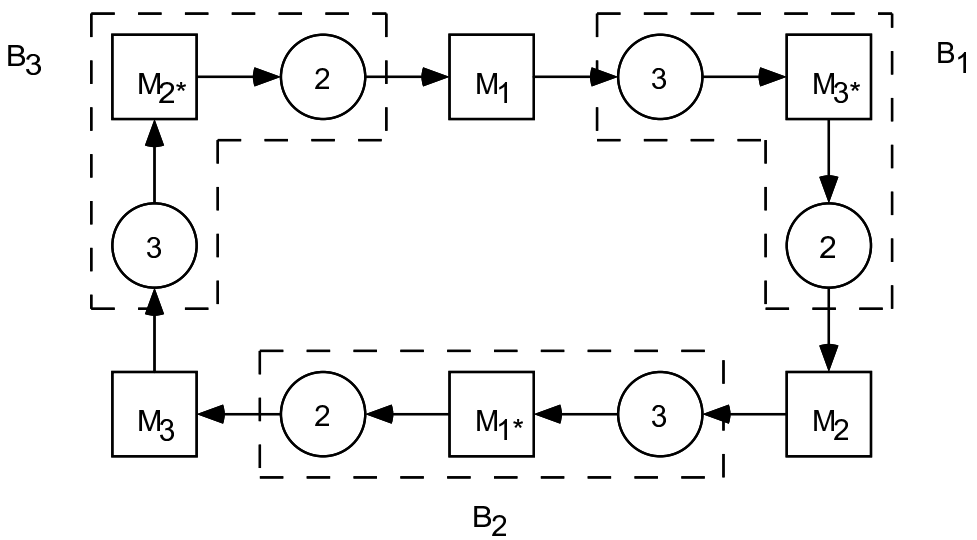

Figure 2-2: Illustration of a transformed loop

We can extend this approach to any K-machine loop. For each threshold $0 < l_k(i) < N_i$, we insert a perfectly reliable machine M_{k^*} into buffer B_i such that $\Psi(k^*, k) = N^p$. B_i is now represented as a buffer of size $N_i - l_k(i)$ followed by M_{k^*} followed by a buffer of size $l_k(i)$. Since each unreliable machine can cause at most

one threshold between zero and N_i, the transformed loop will consist of at most $2K$ machines. Although the loop is larger, we can now use the same building block that is used in Tolio's transfer line decomposition. More importantly, the transformation allows us to analyze a much wider range of buffer sizes and population levels.

2.5.1 Special Cases

When the population of the loop is less than the size of one or more of the buffers, it is not necessary to insert a reliable machine for all $0 < l_k(i) < N_i$. If $N_i > N^p$ there will be a threshold $l_{i+1}(i) = N^p$. Since we know the level of B_i can never exceed N^p, the threshold has a different meaning than in other cases. Instead of adding a reliable machine, we truncate the capacity of B_i so that $N_i = N^p$.

Due to symmetry, the same argument applies when $N^{total} - N^p$, or the number of *holes*, is less than one or more of the buffer capacities.

Buffers of Size One It is possible that the original loop may contain a buffer of size one. Depending on the population of the loop and the size of the buffers, the transformation may also create a buffer of size one. According to our model of the two-machine building block, the line is starved when the buffer is empty and blocked when the buffer is full. Since a buffer of size one must always be empty or full by definition, our model gives artificially low values for average throughput in this case.

To alleviate this effect, we replace all buffers of size one with buffers of size two. While this approach is somewhat arbitrary, it results in a better approximation of average throughput and has only a negligible impact on average buffer levels.

2.6 Fixed Population Considerations

Once the loop is transformed to eliminate all thresholds between zero and N_i, we must account for the limited propagation of blocking and starvation due to a fixed population level. To do this, we define the *range of starvation* and *range of blocking*, indexed on the buffer number. The range of starvation of B_i is the set $\{M_{s(i)}, M_{s(i)+1}, ..., M_i\}$, where $M_{s(i)}$ is the machine farthest upstream which can cause B_i to become empty if it is failed for a long period of time. Similarly, the range of blocking of B_i is the set $\{M_{i+1}, M_{i+2}, ..., M_{b(i)}\}$, where $M_{b(i)}$ is the machine farthest downstream which can cause B_i to become full. We calculate $M_{s(i)}$ and $M_{b(i)}$ as follows:[3]

$$M_{s(i)} = \min_j M_{i+j} \qquad s.t. \;\; \Psi(i, i+j) \quad > N^p \tag{2.6}$$

[3]Note that the inequalities are strict. We use this convention to deal with the situation of simultaneous blocking and starvation, which is discussed in Section 2.6.1.

$$M_{b(i)} = \max_j M_{i+j+1} \quad s.t. \ \Psi(i+1, i+j+1) < N^p \tag{2.7}$$

The loop population is incorporated into the model by including in the building blocks only those virtual failure modes related to machines within the range of blocking and range of starvation. $M^u(i)$ has virtual failure modes corresponding only to the failure modes of $M_{s(i)}$ through M_{i-1}. Likewise, $M^d(i)$ has virtual failure modes corresponding to M_{i+2} through $M_{b(i)}$.

2.6.1 Simultaneous Blocking and Starvation

If $\Psi(v, w) = N^p$ then machine M_v can become simultaneously blocked and starved when M_w is down for a long period of time. This is the case where the threshold $l_w(v-1) = 0$ and $l_w(v) = N_v$. In transformed loops, this situation can occur at each reliable machine M_{k^*} when M_k fails since the buffer sum between the two machines $\Psi(k^*, k) = N^p$ by construction.

The two-machine building block developed in [TG96] does not account for the states where both machines are down and the buffer level is either zero or N. Rather than modifying the building block, we associate the zero buffer level case with an upstream failure and the N buffer level case with a downstream failure.

To help justify this convention, we define the *state* of the building block $L(i)$ to be $(n(t), \alpha_u(t), \alpha_d(t))$, the buffer level, the state of M^u, and the state of M^d at time t. M^u can be up in state 1, down in real mode λ_{ij}, or down in virtual mode λ_{kj}. Similarly, M^d can be up in state 1, down in real mode $\lambda_{i+1,j}$, or down in virtual mode λ_{kj}.

There are two cases of simultaneous blocking and starvation that we must consider, $(N, \lambda_{kj}, \lambda_{kj})$ and $(0, \lambda_{kj}, \lambda_{kj})$. The first state corresponds to the case where M_i is both blocked and starved and the second to the case where M_{i+1} is both blocked and starved.

We first consider the state $(N, \lambda_{kj}, \lambda_{kj})$. Here, both M^u and M^d are down in a virtual mode due to the failure of M_k. However, even if M^u is repaired, it still cannot perform an operation because it is blocked by M^d. Our model assumes that as soon as M_k is repaired, M^u and M^d both become operational. Therefore, under the assumptions of our model, the state $(N, \lambda_{kj}, \lambda_{kj})$ behaves exactly the same as the state $(N, 1, \lambda_{kj})$ and we label λ_{kj} as a downstream failure mode.

We use similar reasoning to account for the state $(0, \lambda_{kj}, \lambda_{kj})$. Here, whether or not M^d is up has no effect on the production rate or buffer level since it is starved by M^u. In this case, we label λ_{kj} as an upstream failure mode.

Chapter 3

Loop Decomposition

In order to decompose closed-loop systems, we must first establish a building block and then find a way to relate these building blocks to one another. This section discusses the required parameters and the equations that we use to find them. Since it is always possible to transform a loop into one in which thresholds are not needed, we restrict our attention to loops without thresholds.

3.1 The Building Block Parameters

As in the Tolio transfer line decomposition, we evaluate the loop by breaking it up into a series of two-machine building blocks. Each building block $L(i)$ is associated with the buffer B_i in the original loop. The upstream machine $M^u(i)$ has real failure modes corresponding to those of M_i and virtual failure modes corresponding to those of machines $M_{s(i)}$ through M_{i-1}. Similarly, $M^d(i)$ has real failure modes corresponding to those of M_{i+1} and virtual failure modes corresponding to those of machines M_{i+2} through $M_{b(i)}$.

As shown in [MMGT00], the failure and repair probabilities for the real failure modes are equal to the probabilities of the corresponding modes of the machines in the loop. Therefore, we have

$$p^u_{ij}(i) = p_{ij} \tag{3.1}$$

$$r^u_{ij}(i) = r_{ij} \tag{3.2}$$

$$p^d_{i+1,j}(i) = p_{i+1,j} \tag{3.3}$$

$$r^d_{i+1,j}(i) = r_{i+1,j} \tag{3.4}$$

In addition, we know that the probability of repair when a machine is down in virtual failure mode λ_{kj} is equal to the probability that machine M_k is repaired when

it is down in failure mode j. This gives us

$$r^u_{kj}(i) = r_{kj} \tag{3.5}$$

$$r^d_{kj}(i) = r_{kj} \tag{3.6}$$

To evaluate the performance measure of the loop, we must find the virtual failure probabilities $p^u_{kj}(i)$ and $p^d_{kj}(i)$ for each $L(i)$. This is the objective solving the decomposition equations.

3.2 Decomposition Equations

The decomposition equations are nearly identical to the transfer line decomposition equations presented in [TM98]. In fact, we need only modify the indices to account for the range of blocking and starvation and the fact that loops contain as many buffers as machines.

We define $P^{st}_{kj}(i)$ as the probability that $B(i)$ is empty due to $M^u(i)$ being down in virtual failure mode λ_{kj}. Likewise, $P^{bl}_{kj}(i)$ is the probability that $B(i)$ is full due to $M^d(i)$ being down in virtual failure mode λ_{kj}. Finally, we define $E(i)$ to be the average throughput of building block $L(i)$. Using this notation, we write the decomposition equations. Recall from Section 2.6 that $s(i)$ is the number of the machine furthest upstream which falls within the range of starvation of buffer B_i. Similarly, $b(i)$ is the number of the machine furthest downstream which falls within the range of blocking of B_i. Then,

For $i = 1$ to K', $k = s(i)$ to $i - 1$, $\forall j$:

$$p^u_{kj}(i) = \frac{P^{st}_{kj}(i-1)}{E(i)} r_{kj} \tag{3.7}$$

For $i = 1$ to K', $k = i + 2$ to $b(i - 1)$, $\forall j$:

$$p^d_{kj}(i) = \frac{P^{bl}_{kj}(i+1)}{E(i)} r_{kj} \tag{3.8}$$

We know the values of r_{kj} from the parameters of the machines in the transformed loop. By solving the building block transition equations presented in [TM98] for each $L(i)$, we can find the values of $P^{st}_{kj}(i)$, $P^{bl}_{kj}(i)$, and $E(i)$, which are functions of $p^u_{kj}(i)$ and $p^d_{kj}(i)$. The decomposition equations (3.7) and (3.8) represent a system of $2K'$ independent equations in $2K'$ unknowns.

Chapter 4

Implementing the Loop Transformation and Decomposition

This section provides a step-by-step procedure for evaluating a loop. First, we transform an arbitrary loop into one without thresholds. Next, we introduce a set of decomposition equations and an algorithm which is a slight modification of that of [TM98] for solving them.

4.1 The Transformation Algorithm

(a) For all $N_i > N^p$, set $N_i = N^p$. For all $N_i > N^{total} - N^p$, set $N_i = N^{total} - N^p$ (In this case we add $N_i - N^{total} - N^p$ to the resulting average buffer level in order to recover the true average buffer level for B_i).

(b) Insert a perfectly reliable machine M_{i*} for every unreliable machine M_i such that $\Psi(i*, i) = N^p$ (unless a machine M_j such that $\Psi(j, i) = N^p$ already exists). The new loop consists of K' machines separated by K' buffers. Note that the size of the buffers may be different from the original buffers, but the total buffer space in the transformed loop is equal to that of the original.

(c) Re-number the machines and buffers from 1 to K'.

(d) For all $N_i = 1$, set $N_i = 2$ (see Section 2.5.1).

4.2 The Decomposition Algorithm

(a) Calculate the range of starvation and range of blocking for each $L(i)$ using (2.6) and (2.7) to determine all valid failure modes λ^u_{kj} and λ^d_{kj}.

(b) Initialize $p_{ij}^u(i)$, $r_{ij}^u(i)$, $p_{i+1,j}^d(i)$, $r_{i+1,j}^d(i)$, $r_{kj}^u(i)$, and $r_{kj}^d(i)$ for all valid failure modes using equations (3.1), (3.2), (3.3), (3.4), (3.5), and (3.6). Set $p_{kj}^u(i) = p_{kj}$ and $p_{kj}^d(i) = p_{kj}$.

(c) For $i = 1$ to K':
– Calculate $E(i)$ and $P_{kj}^{st}(i)$.
– Update $p_{kj}^u(i+1)$ using (3.7).

(d) For $i = K'$ to 1:
– Calculate $E(i)$ and $P_{kj}^{bl}(i)$.
– Update $p_{kj}^d(i-1)$ using (3.8).

(e) Repeat (c) and (d) until the parameters converge to an acceptable tolerance.

(f) Record performance measures.
– Use $\max_i E(i)$ as an estimate of the overall average throughput E. The maximum has proven to be the most robust estimator of average throughput. In addition, it allows us to avoid complications that result from buffers of size one. This is discussed further in section 2.5.1
– Calculate the average buffer level $\overline{n}(i)$ as the sum of all average buffer levels between unreliable machine i and unreliable machine $i+1$.

Chapter 5

Performance of the Method

5.1 Cases Studied

The algorithm was tested extensively on three-, six-, and ten-machine loops with machine parameters and buffer sizes generated randomly using Microsoft Excel. Repair probabilities for each machine in the loop were drawn from a uniform distribution between 0.05 and 0.2 and were specified to be of the same order of magnitude. Failure probabilities were randomly generated from a uniform distribution such that the isolated efficiency $r/(r+p)$ of each machine in the loop was between 75 and 99 percent. Buffer sizes were drawn from a uniform distribution between $1/5r$ and $5/r$. For five loops of each size, the decomposition and simulation were performed for all possible population levels. Additional cases were studied where the population was randomly generated from a uniform distribution between the size of the largest buffer N^{max} and $N^{total} - N^{max}$, the total buffer capacity minus the size of the largest buffer. For a partial listing of the loops studied, see Appendices A, B, and C. Here, we examine the accuracy, convergence reliability, and speed of the method.

5.2 Accuracy

In this section, we compare the analytical results to those obtained through simulation. The measures of interest are average throughput and average buffer levels.

5.2.1 Population Range

One of the assumptions of our decomposition approximation is that travel time for parts is negligible. However, when the number of parts in the system is less than the number of machines, travel time can not be ignored. Consider a three-machine loop with buffers of size 5 and perfectly reliable machines. If there is only one part in the system, the maximum throughput is 1/3 since the operation time at each machine is

one. In this situation, ignoring travel time has a significant impact on the throughput. Due to symmetry, this same argument applies when the number of *holes* (i.e. the total buffer capacity minus the number of parts) is less than the number of machines.

In testing the method for accuracy, we chose to focus on loops with populations where $N^{max} \leq N^p \leq N^{total} - N^{max}$. While in many cases the algorithm performs well outside this range, this provides a conservative estimate of the capability of the algorithm. Additionally, this range seems acceptable for dealing with real-world cases.

5.2.2 Three-machine Loops

The method was observed to be accurate in evaluating three-machine loops. In all cases where the population was within the specified range, the relative throughput error is less than 1% (see Figure5-1).[1]

Buffer level error was calculated as

$$Error = |\frac{2(\bar{n}(i)_{decomposition} - \bar{n}(i)_{simulation})}{N_i}| \tag{5.1}$$

The mean for buffer level error for the 48 three-machine cases is 2.9%, but the error reaches as high as 9.58% (see Figure 5-2).

For three-machine loops, the method is still accurate outside the specified population range. As long as the number of parts (or holes) is greater than 2, the throughput error is generally less than 2%. Figure 5-3 shows the throughput error in this range for a typical three-machine loop.

5.2.3 Six-machine Loops

For six-machine loops, the mean throughput error increased to 1.1% with a maximum of 2.7% for the 287 cases tested. The mean buffer level error is 5.04%. Although the maximum error is 21.0%, 87.3% of the cases have buffer level errors of less than 10.0%. Figures 5-4, 5-5, and 5-6 give a graphical representation of the results.

5.2.4 Ten-machine Loops

In the ten-machine loops studied, the errors are slightly larger. For the 454 cases tested, the mean throughput error is 1.43% and the maximum is 4.04% (see Figure 5-7). Average buffer level errors range from 0.0% to 43.8% with a mean of 6.27%. 80.2% of the mean buffer errors are less than 10.0%.

[1] In all of the error graphs, cases are sorted in order of ascending error.

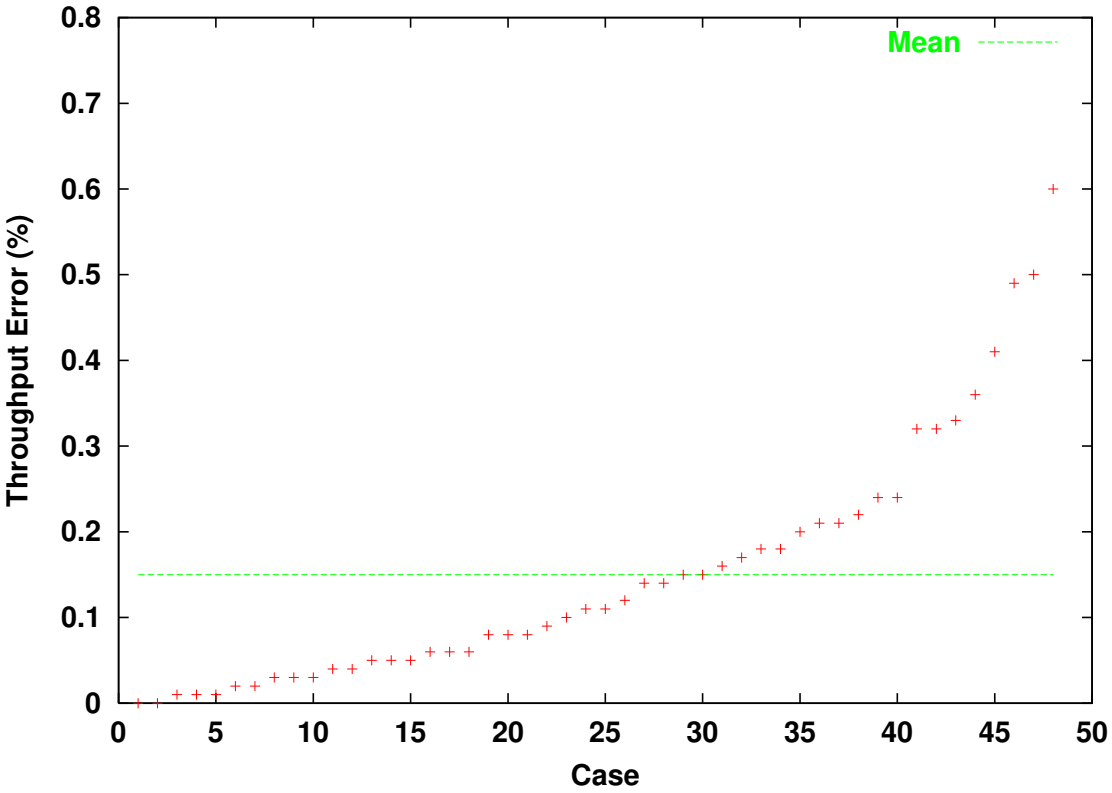

Figure 5-1: Throughput error (three-machine loops)

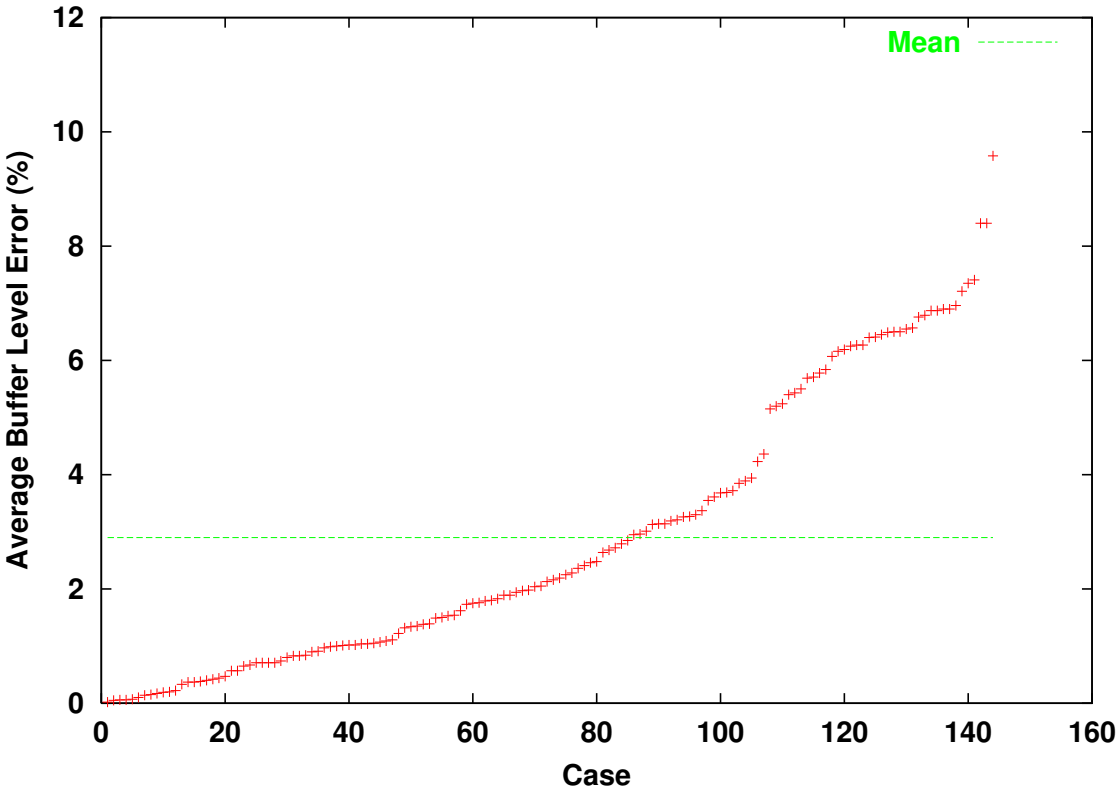

Figure 5-2: Average buffer level error (three-machine loops)

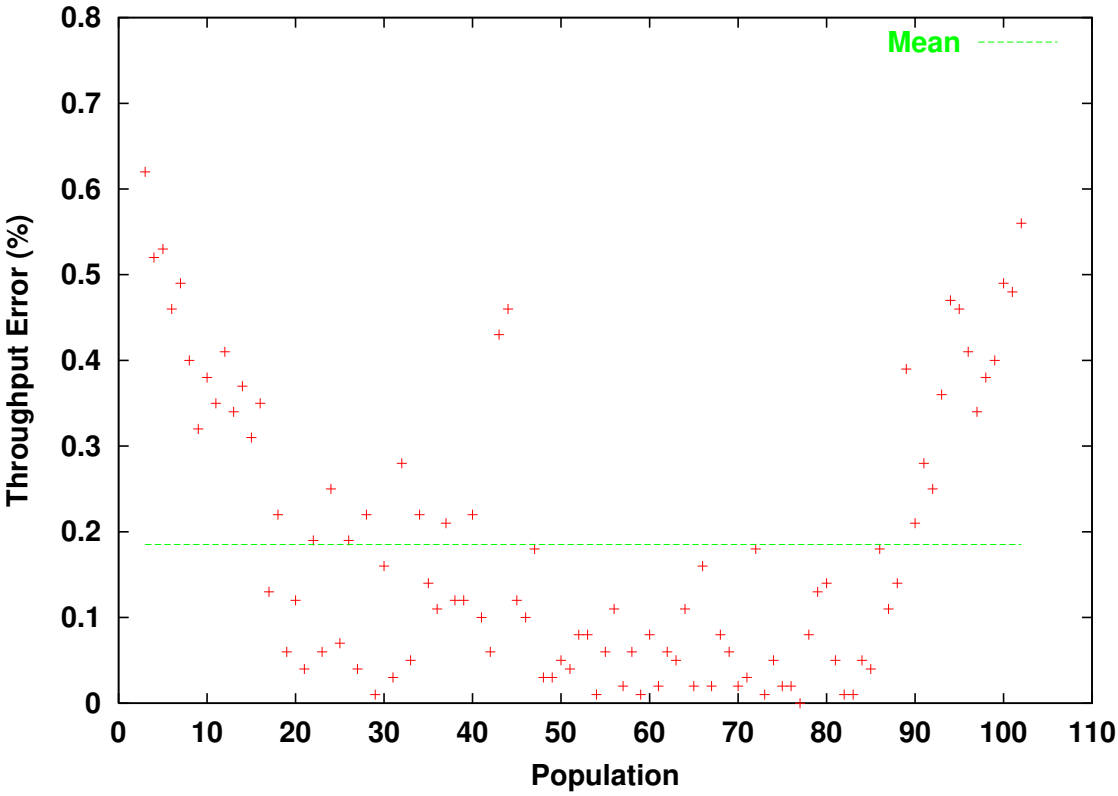

Figure 5-3: Throughput error versus population (Loop 3.2)

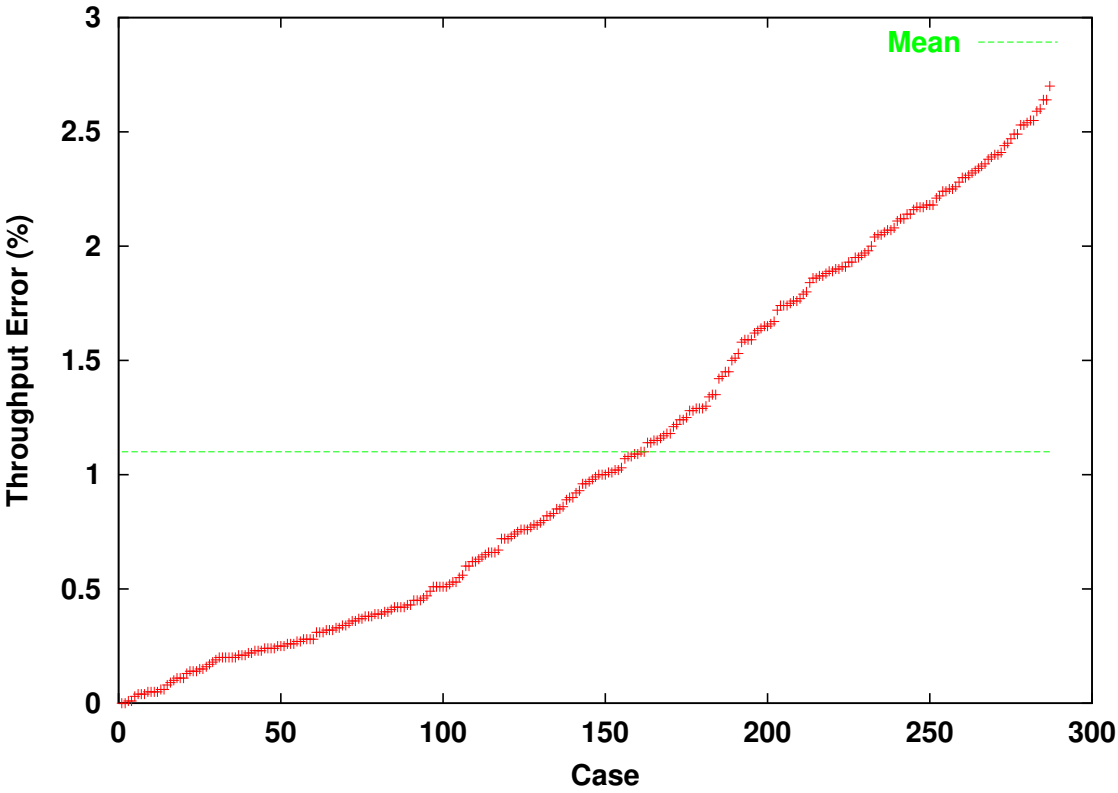

Figure 5-4: Throughput error (six-machine loops)

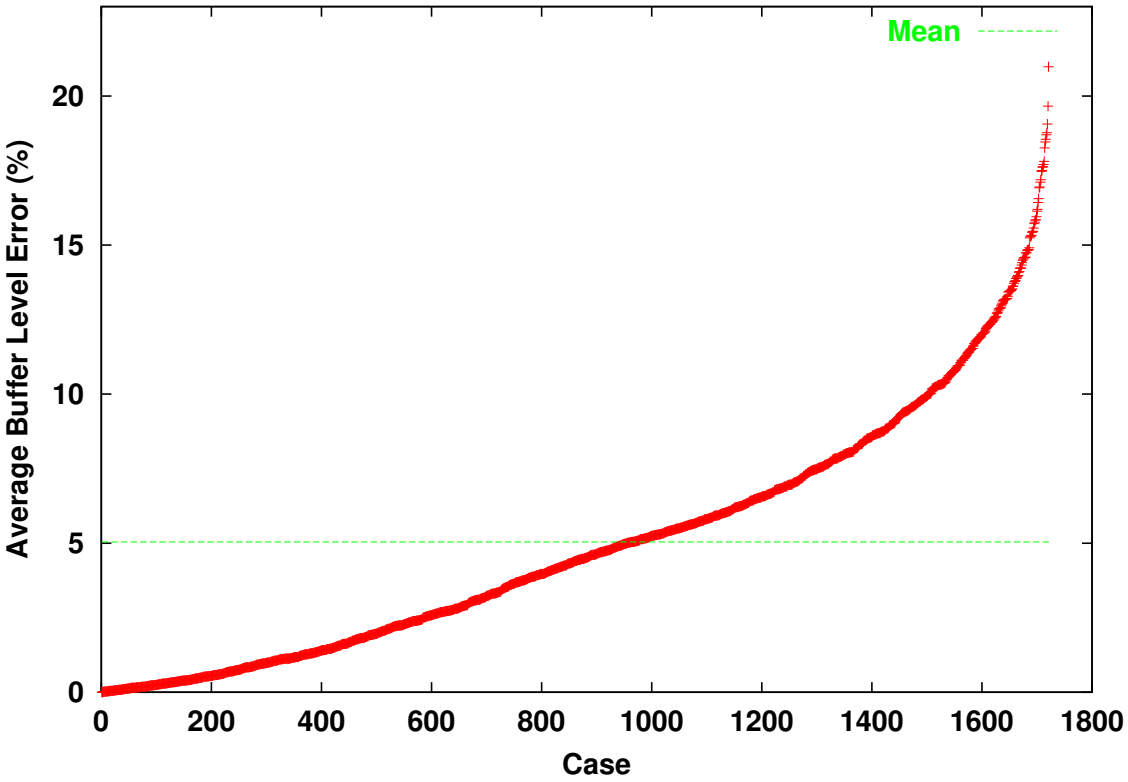

Figure 5-5: Average buffer level error (six-machine loops)

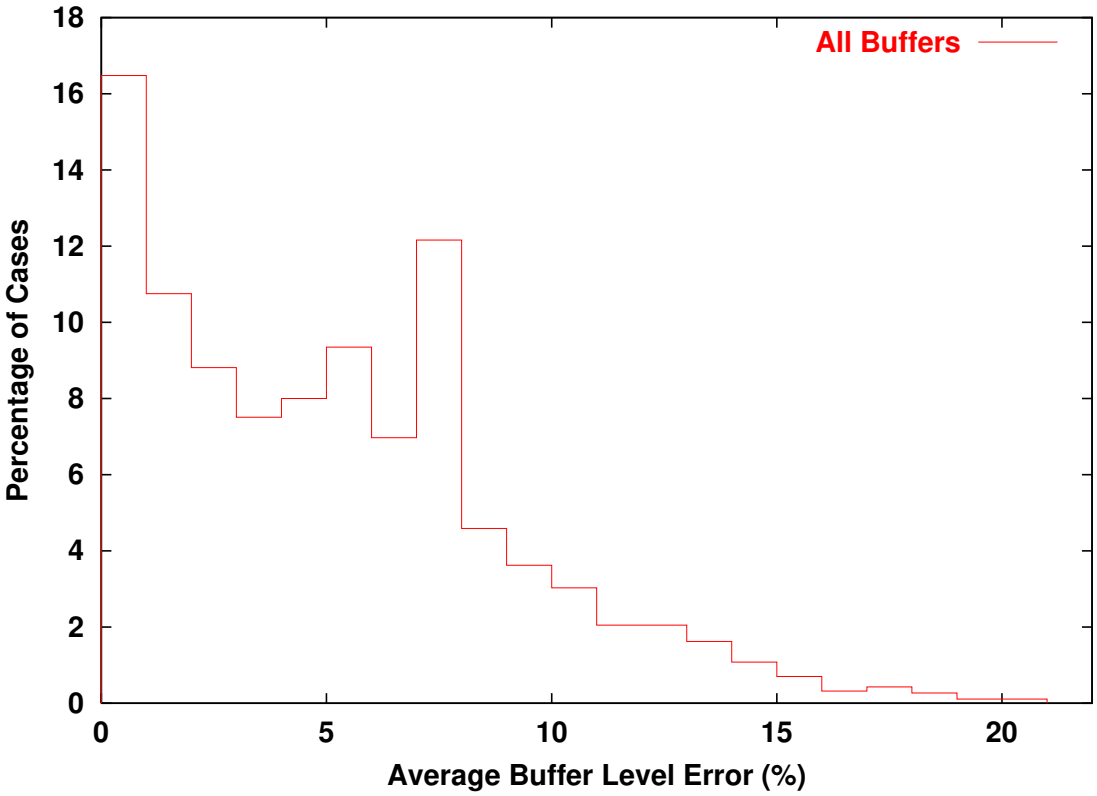

Figure 5-6: Average buffer level error (six-machine loops)

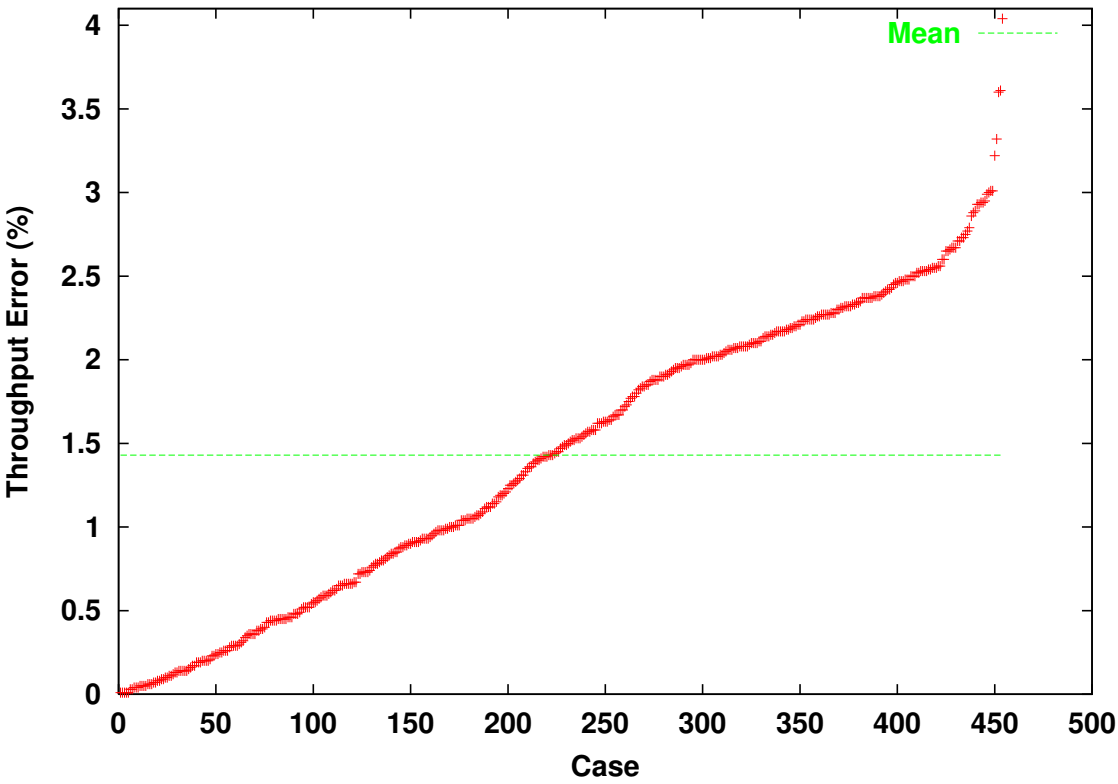

Figure 5-7: Throughput error (ten-machine loops)

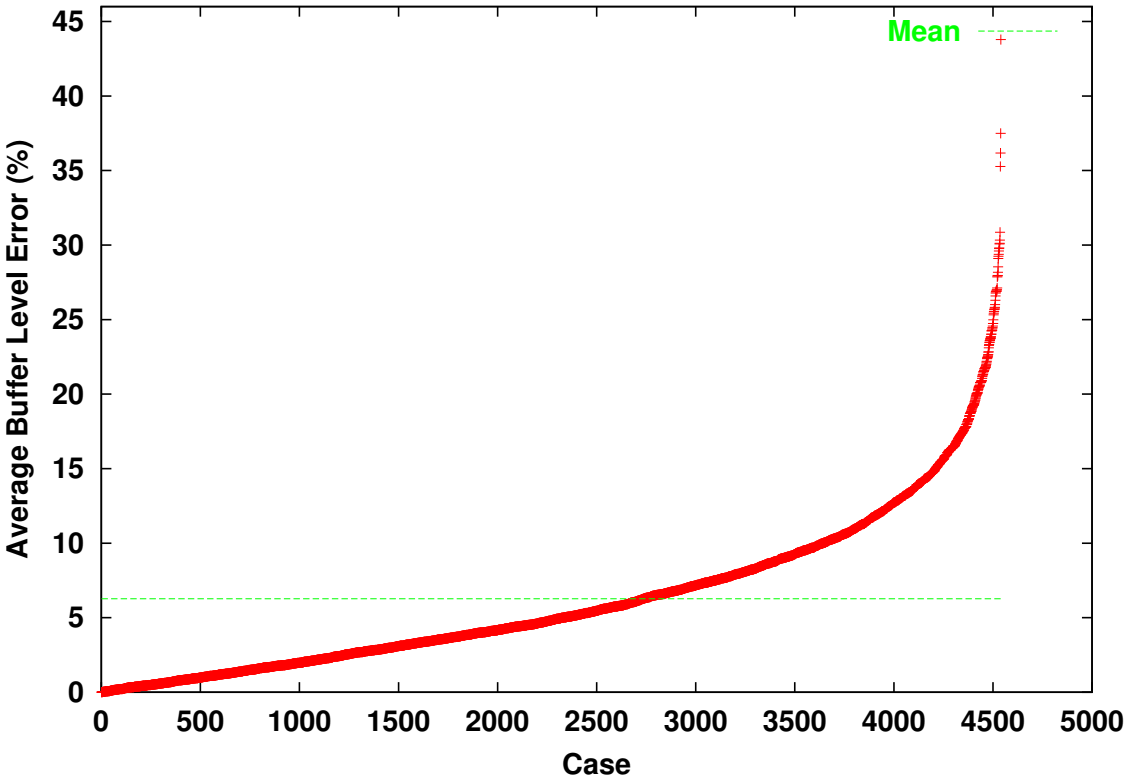

Figure 5-8: Average buffer level error (ten-machine loops)

5.2.5 The Batman Effect

The accuracy of the algorithm seems to deteriorate slightly when the population of the loop is nearly equal to the total capacity of one or more buffers. We call this phenomenon the *batman effect*. To explain, we consider a three-machine loop in which all the machines and buffers are identical. For this case, we set $r = 0.1$, $p = 0.01$, and $N = 10$. Figure 5-9 shows the throughput versus population curve for this loop.

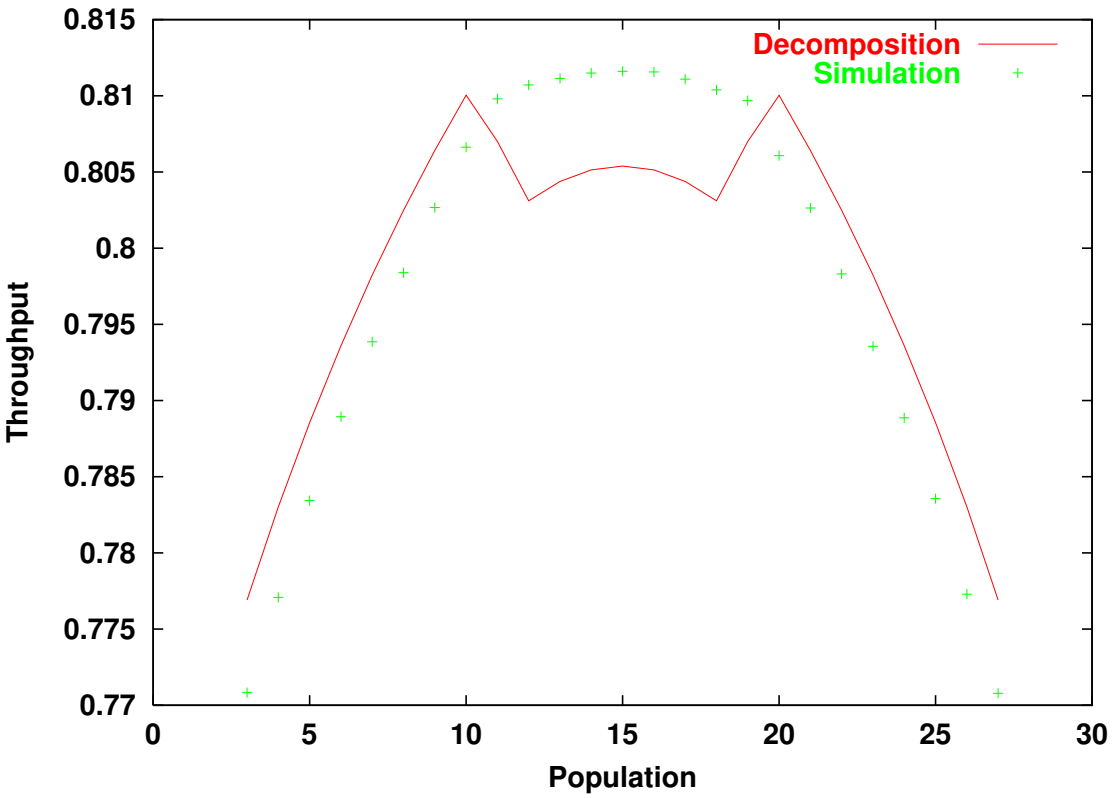

Figure 5-9: Illustration of the batman effect for a three-machine loop

The throughput peaks at $N^p = 10$ and $N^p = 20$. We do not completely understand the cause of the batman effect. However, we hypothesize that it results from the small buffers created by the transformation in those regions and the fact that we convert all buffers of size one into buffers of size two (see Section 2.5.1).

The throughput versus population curves of larger symmetrical loops exhibit a similar behavior. Figure 5-10 gives the curve of a six-machine loop with $r = 0.1$, $p = 0.01$, and $N = 10$. Non-symmetrical loops also show signs of discontinuity, but the relationship is more complex.

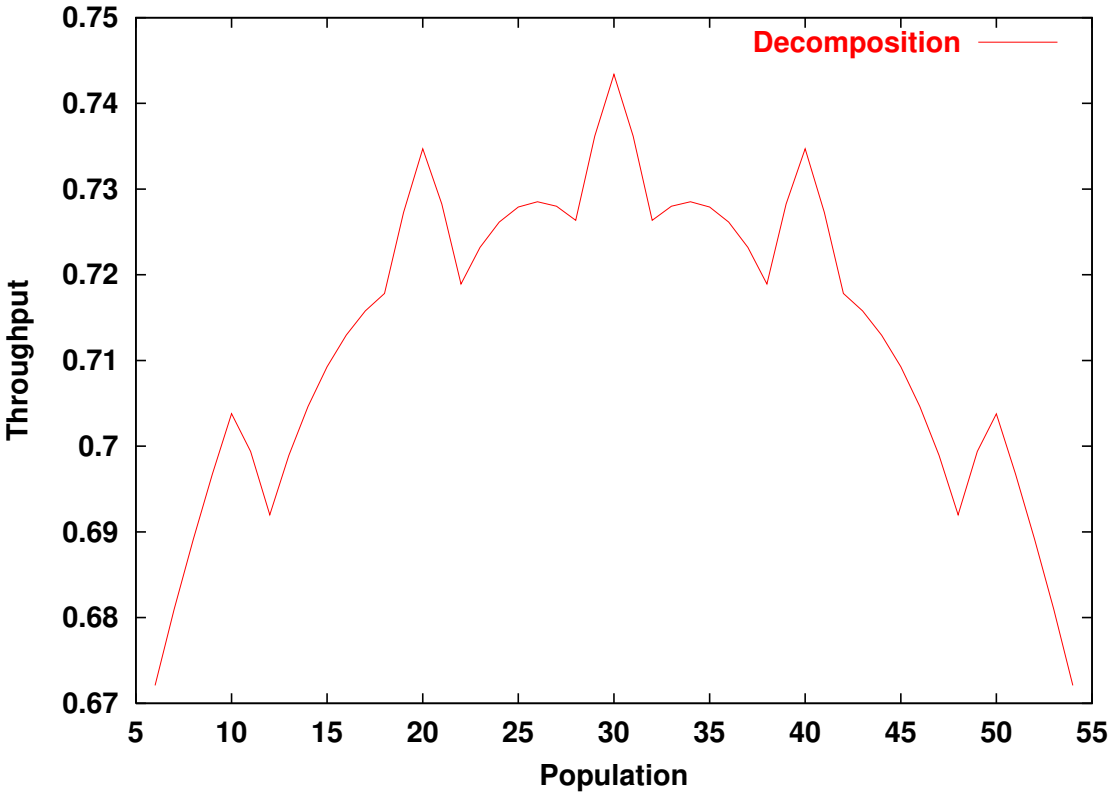

Figure 5-10: Illustration of the batman effect for a six-machine loop

5.3 Convergence Reliability

In nearly all cases studied, the decomposition algorithm converged. The criterion used for convergence was that difference in the value of all $p_u(ij)$s and $p_d(ij)$s between successive iterations be less than the specified tolerance of 10^{-6}. The few cases where the algorithm did not converge were ten-machine loops that exceeded the maximum number of iterations before satisfying the convergence criterion. Even though the algorithm did not converge to the tolerance, the errors in throughput and buffer levels are still very small.

As in Maggio's approach, our decomposition algorithm does not exactly satisfy conservation of flow[2] even though Tolio's equations imply that it should hold. However, the differences between the throughputs of the building blocks are generally very small.

[2]See item (f) in Section 4.2

5.4 Speed

The speed of the algorithm was tested by finding the computation time for five randomly generated loops of size three, six, and ten (See Section 5.1). Cases were run on a 650 MHz Pentium III PC with 128 MB of RAM. The mean run times were 0.61, 8.6, and 53.33 seconds for three-, six-, and ten-machine loops, respectively. Figure 5-11 shows the results. From the graph, we see that computation time increases rapidly as the number of machines in the loop increases.

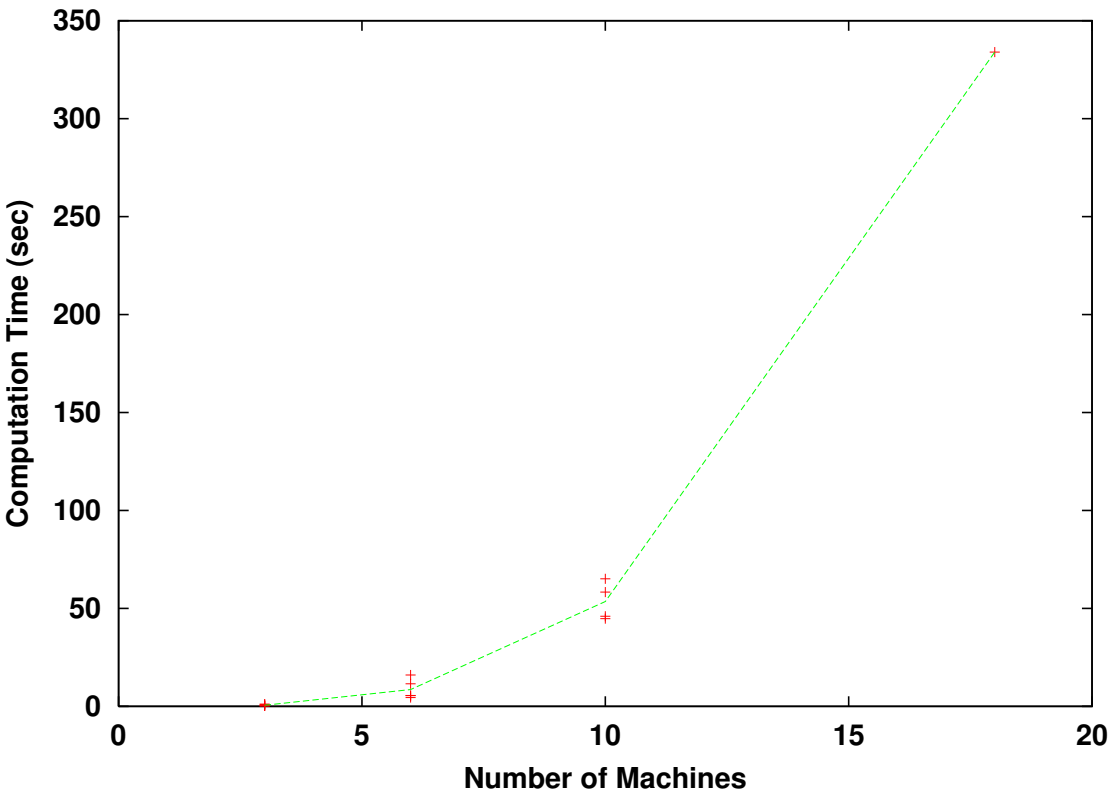

Figure 5-11: Computation time versus loop size

In addition, one case of an 18-machine loop was tested. This was the largest loop evaluated. Each of the machines had $r = 0.1$ and $p = 0.01$. All of the buffers were size 10. The case was run with 100 parts in the system and required 334 seconds of computation time.

Chapter 6

Observations on Loop Behavior

In this section we discuss some of the loop phenomenon we observed while developing and testing the method.

6.1 Flatness

The most interesting observation we have made using the algorithm has to do with the relationship between the loop parameters, population, and throughput. Specifically, we observed a characteristic we call *flatness*, which refers to the shape of the throughput versus population curves of closed-loop systems.

6.1.1 Transfer Line Flatness

This special type of flatness occurs in loops where the capacity of the largest buffer is greater than the sum of the capacities of the other buffers. For all population levels N^p such that $N^{total} - N^{max} < N^p < N^{max}$, the throughput is constant.

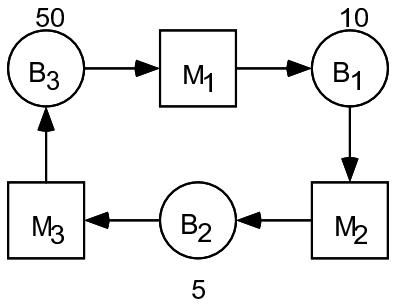

Figure 6-1: Example of loop with transfer line flatness

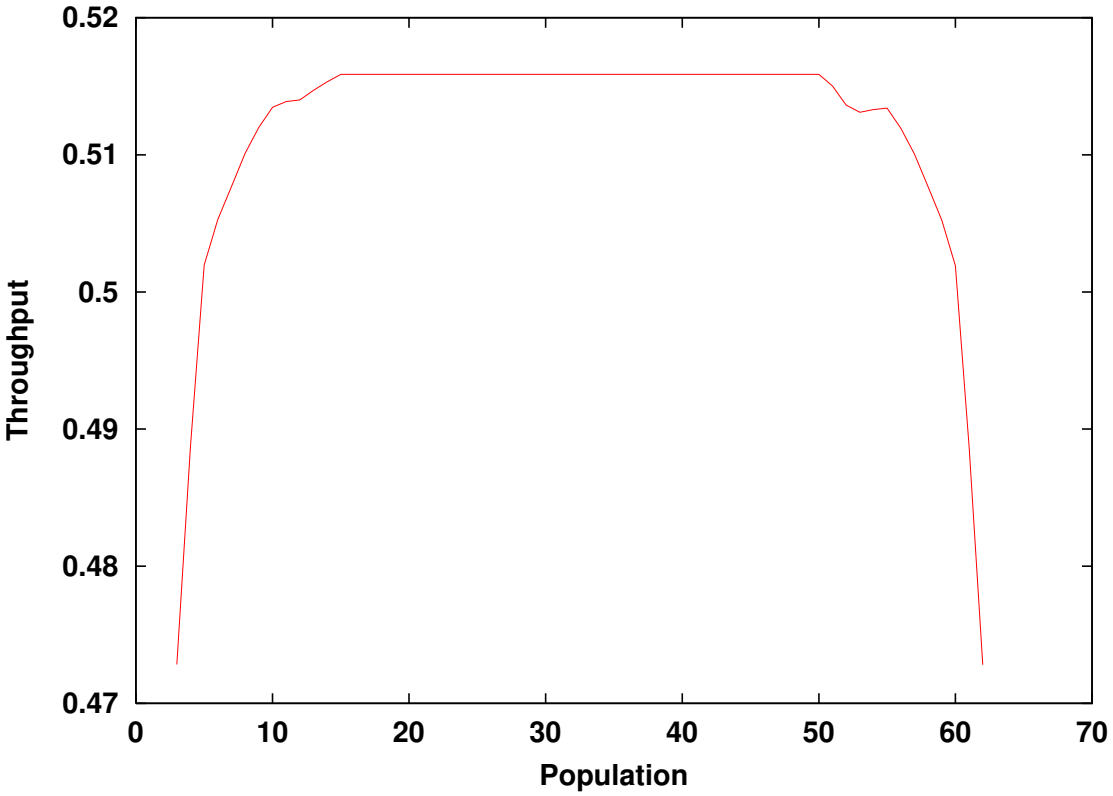

Figure 6-2: Analytical throughput as a function of population

To illustrate the concept of transfer line flatness, we consider a three-machine loop with buffers of size 10, 5, and 50 (see Figure 6-1). When there are 16 parts in the system, it is possible for buffers B_1 and B_2 to be both full and empty. However, B_3 can never become full or empty. This means that machine M_1 can never be starved and M_3 can never be blocked. If we ignore B_3, the system has the same production rate and average buffer levels as a transfer line consisting of M_1, B_1, M_2, B_2, and M_3. This behavior remains the same for populations up to 50 because in each of these cases M_1 is never starved and M_3 is never blocked. In this population range, the average throughput and average buffer levels of B_1 and B_2 remain the same (see Figures 6-2 and 6-3). The only difference is the average buffer level of B_3. Note that the throughput versus population and average buffer level versus population curves do not appear totally smooth. This is due to the approximation involved in the analytical method (see section 5.2.5).

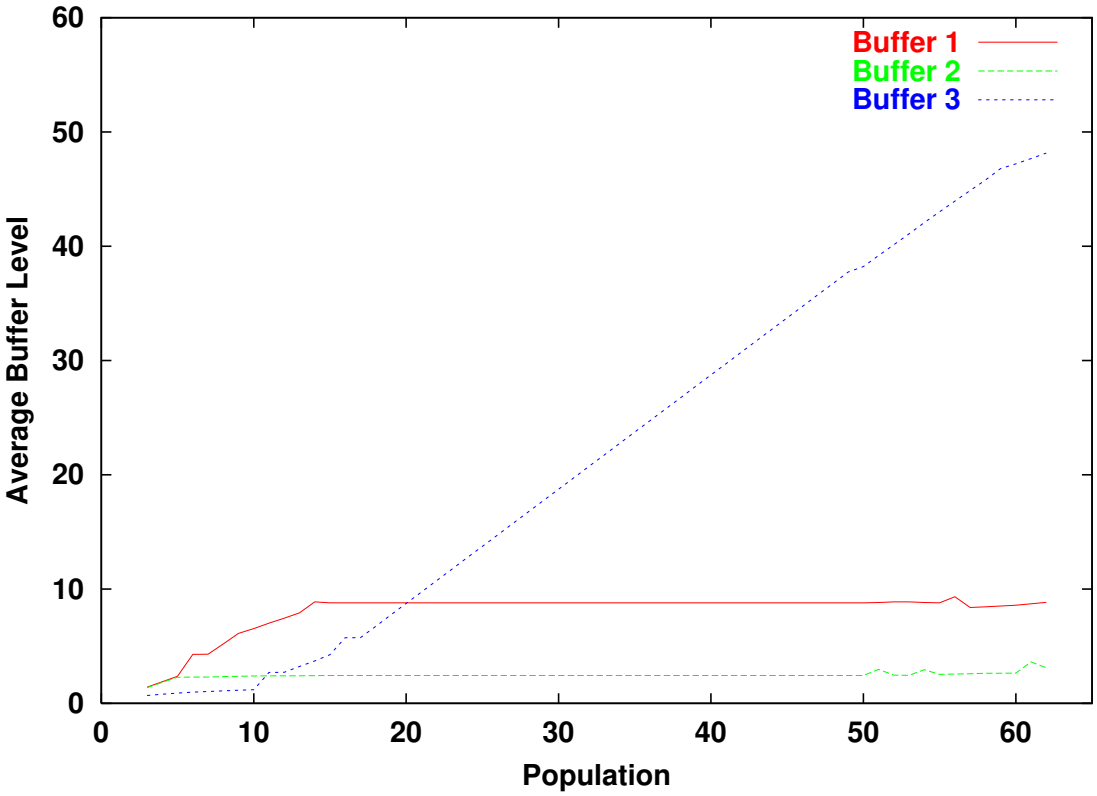

Figure 6-3: Analytical average buffer level as a function of population

6.1.2 Near Flatness and Non-Flatness

We also observed a type of flatness we call *near flatness*. It occurs in loops that do not meet the requirements for transfer line flatness, but have population ranges where the throughput is nearly constant.

In the cases we studied, *symmetrical* loops, in which the machines are identical and the buffer capacities are the same, did not seem to exhibit near flatness. Loops which were very asymmetrical did exhibit near flatness. The degree of flatness seemed to increase with the degree of asymmetry in the loop.

Specifically, the standard deviation in the buffer capacities $\sigma_{buffers}$ seems to give a good indication of how flat the throughput versus population curve will be for a given loop. We calculate $\sigma_{buffers}$ for a K-machine loop as follows:

$$\sigma_{buffers} = \sqrt{\frac{K \sum_i N_i - (\sum_i N_i)^2}{K^2}} \tag{6.1}$$

Figures 6-4, 6-5, and 6-6 illustrate how the degree of flatness increases as $\sigma_{buffers}$ increases from 8.73 to 20.96[1].

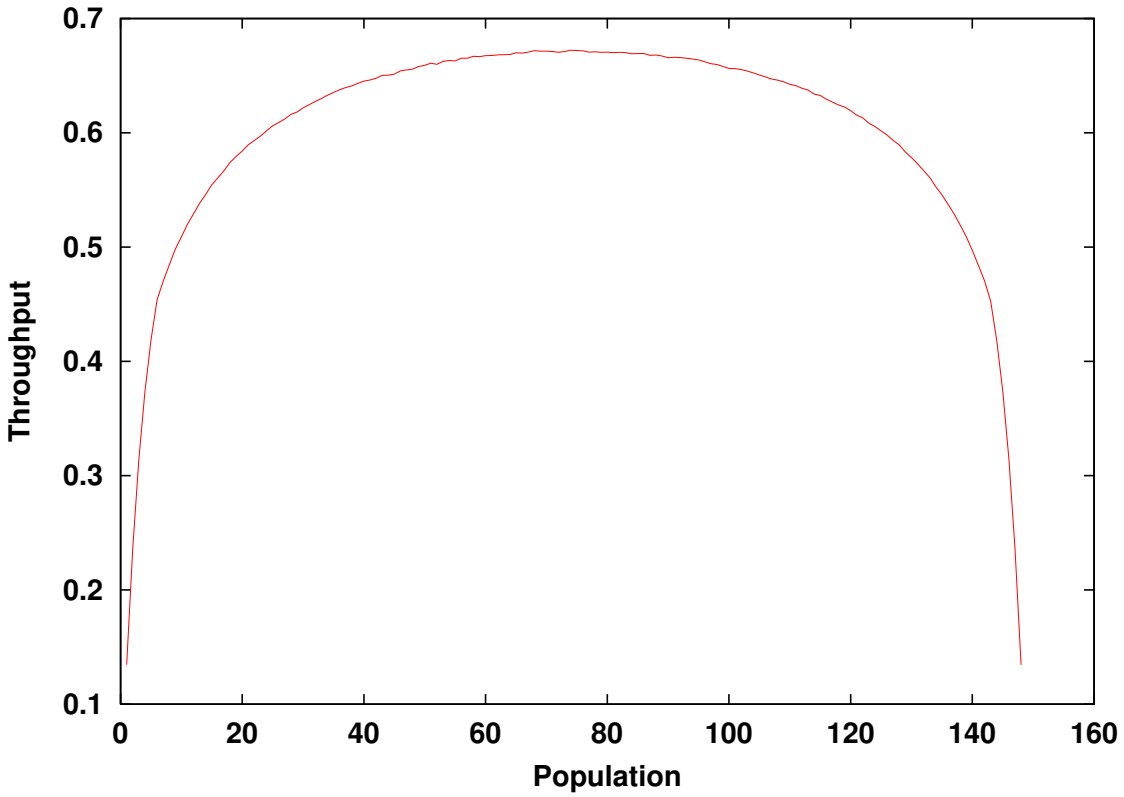

Figure 6-4: Loop 6.4 ($\sigma_{buffers} = 7.97$)

6.2 Changes in Loop Parameters

In this section, we explore how changes in machine parameters and buffer sizes effect average throughput and buffer levels. The basic loop used for these tests is a symmetrical three-machine loop with $r = 0.1$, $p = 0.01$ and $N = 10$. For all of the tests, the loop population is held constant at $N^p = 15$.

[1]The parameters for these loops can be found in Appendix B

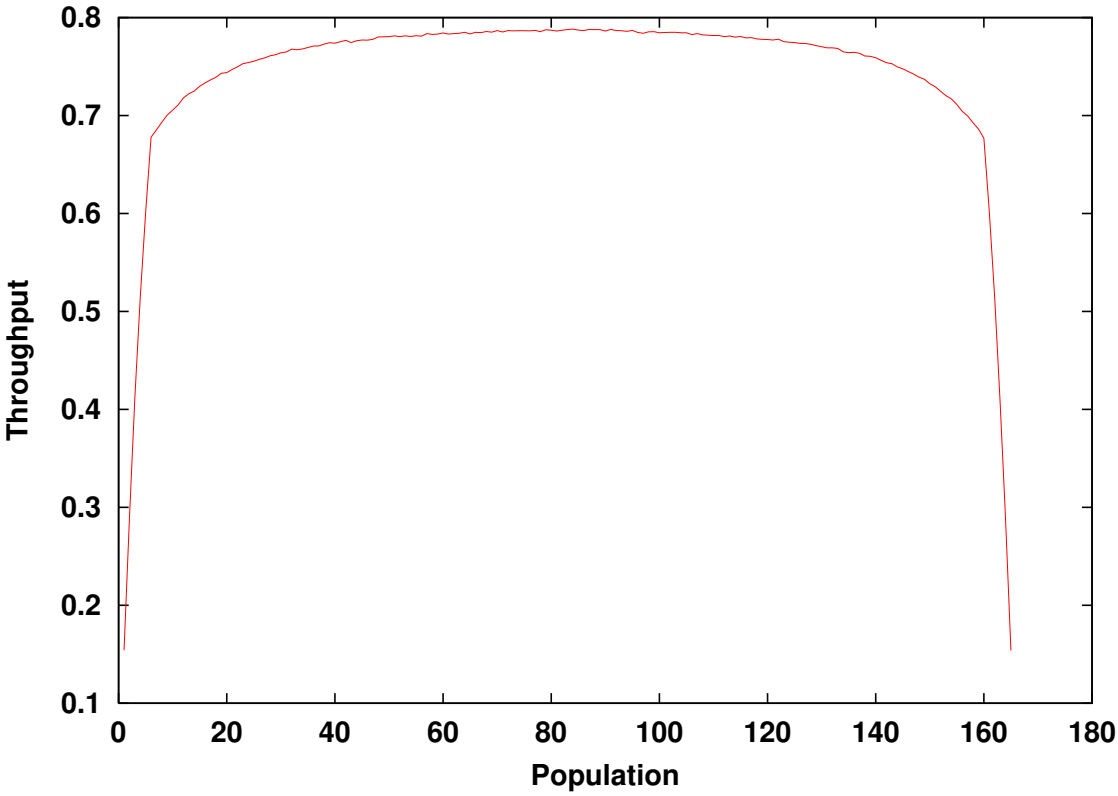

Figure 6-5: Loop 6.1 ($\sigma_{buffers} = 11.80$)

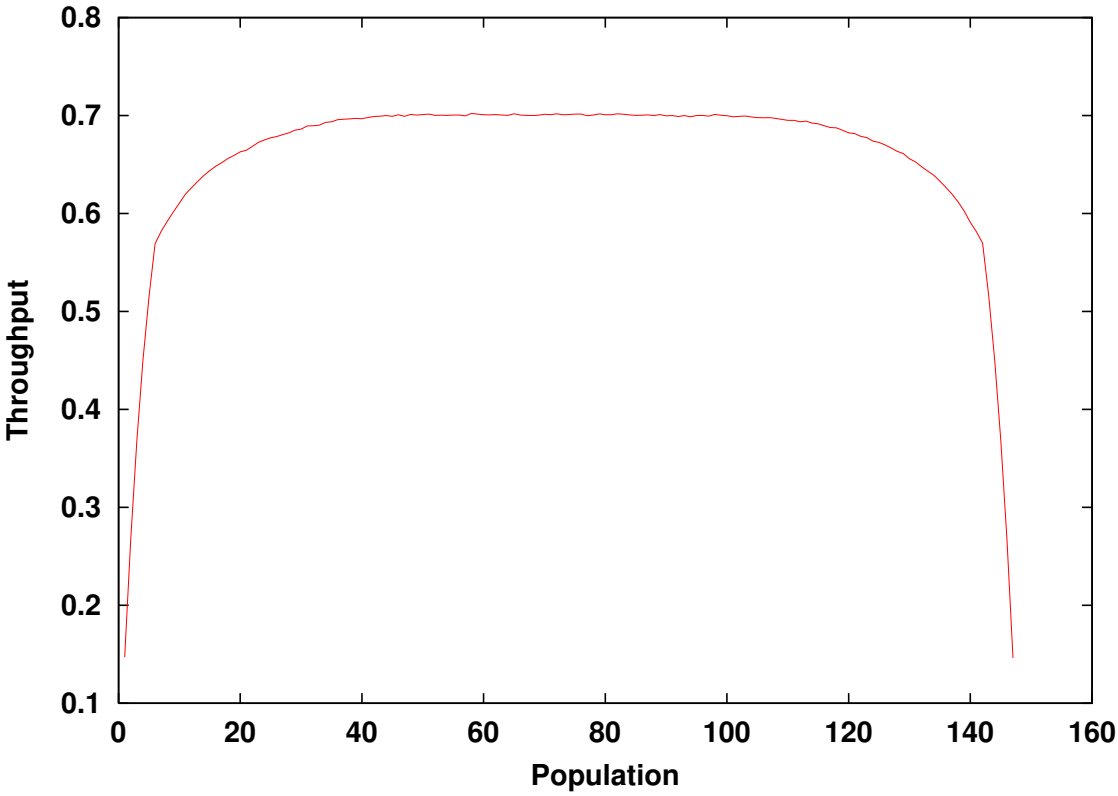

Figure 6-6: Loop 6.2 ($\sigma_{buffers} = 19.12$)

6.2.1 Machine Parameters

For this series of tests, we focus on changes in the repair probability of one or more machines in the loop.

First, we examine the effect of varying r_1 between 0.0 and 1.0. Figures 6-7 and 6-8 give average throughput E and average buffer levels $\overline{n}(i)$ as a function of r_1.

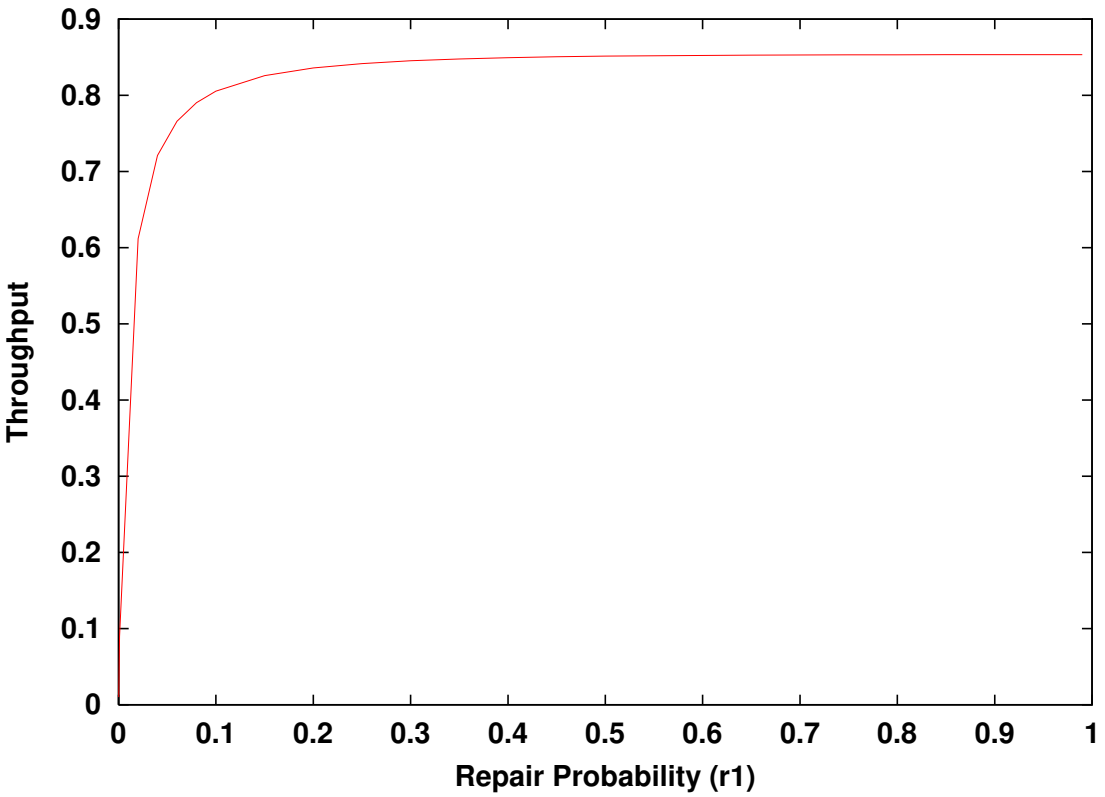

Figure 6-7: Average Throughput as a Function of r_1

We see that as r_1 approaches 0.0, throughput goes to 0.0. In addition, we see that $\overline{n}(1)$ approaches 0.0, $\overline{n}(2)$ approaches 5.0, and $\overline{n}(3)$ approaches 10.0. This result is consistent with intuition. When machine M_1 fails, parts begin to build up buffer B_3 until it reaches its capacity of ten. This causes M_3 to become blocked and parts begin to build up in B_2 until all of the five remaining parts are in B_2. Since $r_1 = 0.0$, the system can never leave this state and throughput is zero.

As r_1 increases, M_1 spends less time down. M_2 is starved less frequently and M_3 is blocked less frequently. This translates into an increase in throughput and $\overline{n}(1)$ and a decrease in $\overline{n}(3)$.

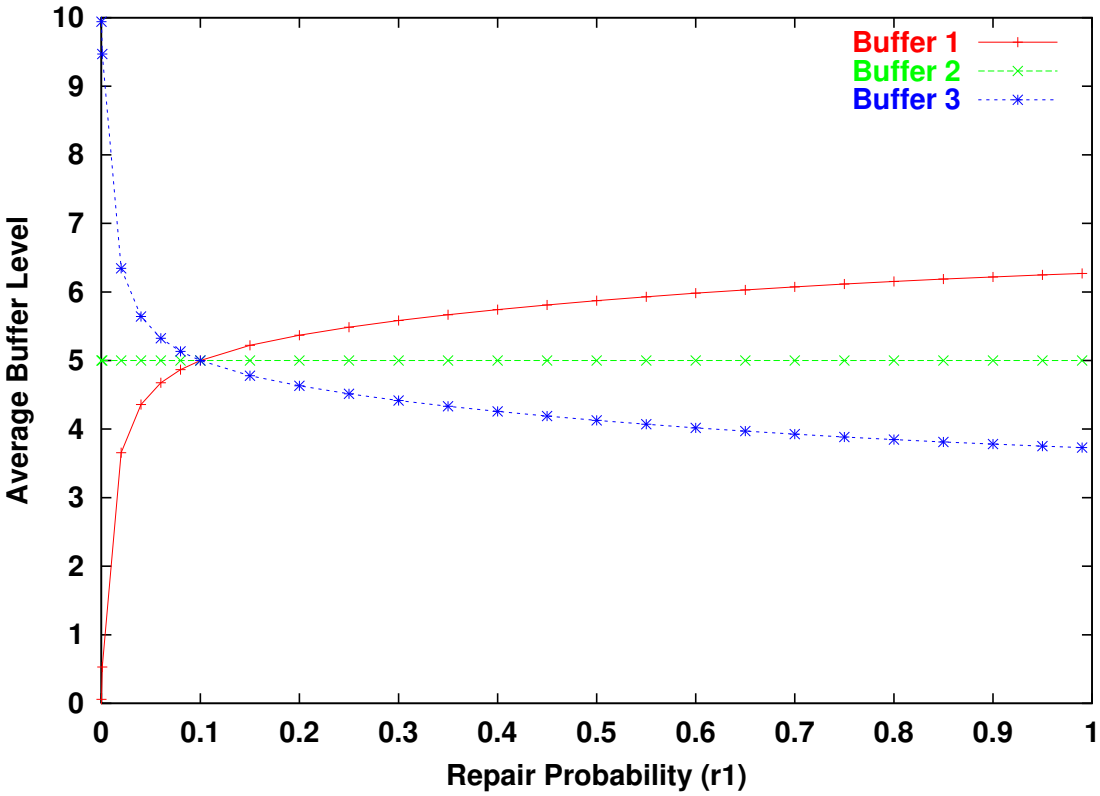

Figure 6-8: Average Buffer Level as a Function of r_1

When $r_1 = 0.1$, the loop is symmetrical and all of the average buffer levels are equal to 5.0 since the 15 parts are distributed evenly between the three buffers.

As r_1 approaches 1.0, the probability that M_2 will be starved approaches 0.0, as does the probability that M_3 will be blocked. By performing what is essentially the inverse of our loop transformation, we can view the system as a two-machine loop where M_1^{new} represents M_2, B_1^{new} represents B_2, M_2^{new} represents M_3, and B_2^{new} represents B_3, M_1, B_1. Since $N_1^{new} = 10$, $N_2^{new} = 20$, and $N^p = 15$, the system acts essentially like a two-machine transfer line made up of the original M_2, B_2, and M_3. When we compare the throughput of the loop with that of the line, we find that they are nearly identical. As r_1 approaches 1.0, the average throughput of the loop approaches 0.8535. The average throughput of the corresponding two-machine transfer line is 0.8561.

Next, we study the effect of varying all of the rs together between 0.0 and 1.0. Since the loop is symmetrical in all cases here, the average buffer levels remain unchanged (i.e. $\overline{n}(i) = 5.0$). However, it is interesting to look at average throughput

as a function of both r and the isolated efficiency $e = \frac{r}{r+p}$ of the machines. Figures 6-9 and 6-10 illustrate the relationship. We observe that throughput is equal to 0.0 when $r = 0$ and approaches 1.0 asymptotically as r increases to 1.0. In addition, we see that throughput increases hyper-linearly as a function of isolated efficiency.

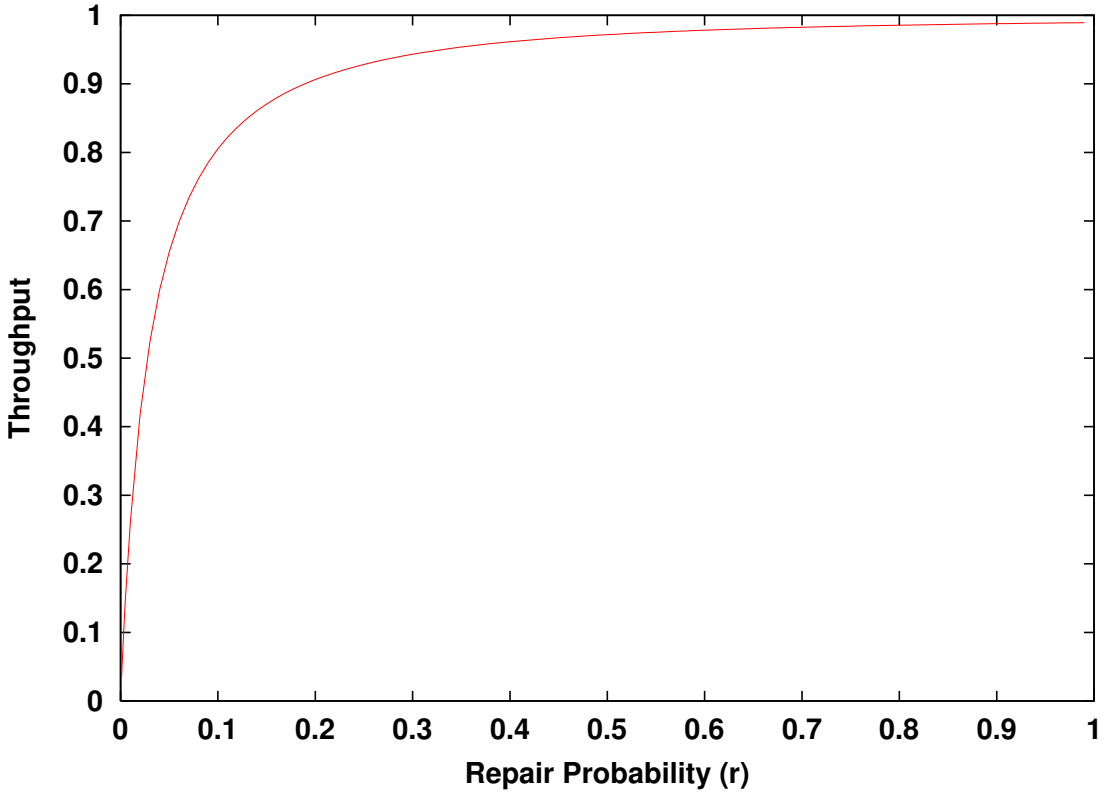

Figure 6-9: Average Throughput as a Function of r

6.2.2 Buffer Size

Here, we consider the effect that changing the buffer sizes has on average throughput. To do this, we use our standard symmetrical three-machine loop but set $N^p = 28$. Figure 6-11 shows how throughput changes as we vary the buffer size N between 10 and 35. The discontinuities in the curve are a result of the batman effect discussed in Section 5.2.5.

When $N = 10$, the probability of blocking P^{bl} is very high, causing throughput to be relatively low. At this point, the probability of starvation P^{st} is zero because the number of holes is less than N. As N increases to 14, P^{bl} decreases but P^{st} remains

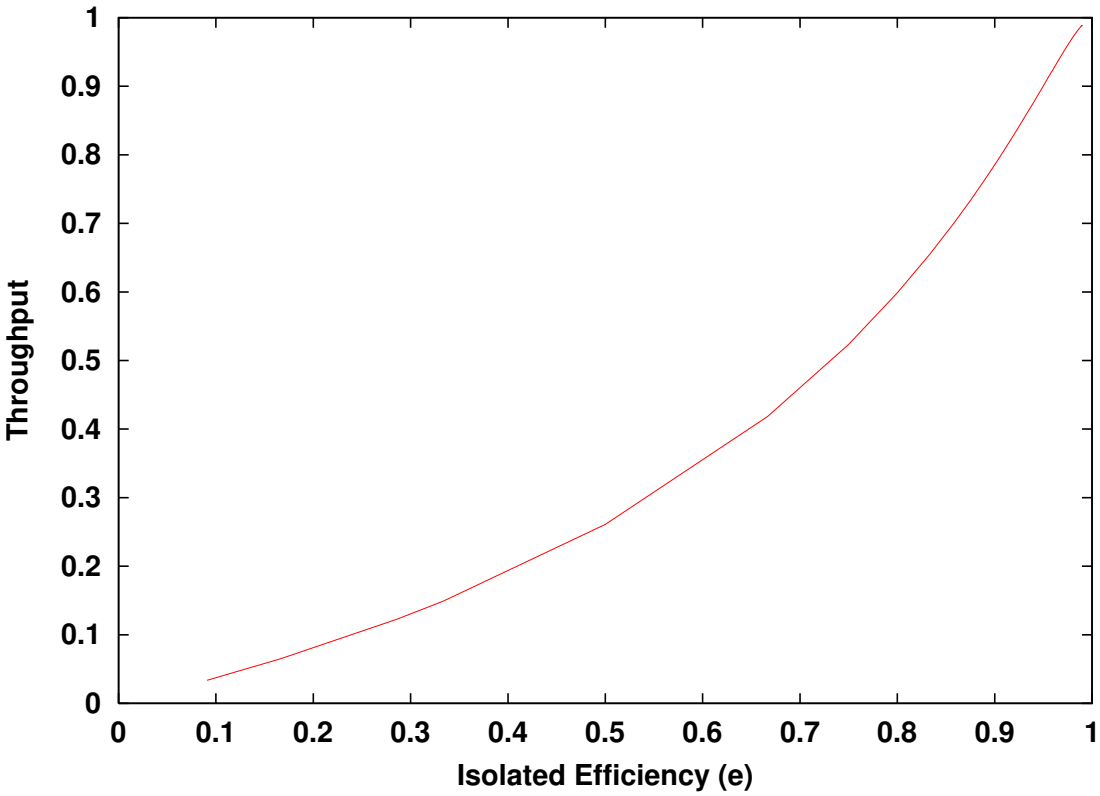

Figure 6-10: Average Throughput as a Function of e

zero, resulting in an increase in throughput. For $N > 13$, P^{st} is no longer zero. In the range $13 < N < 28$, the decrease in P^{bl} is greater than the increase in P^{st} so there is a net increase in throughput. However, for $N \geq 28$, $P^{bl} = 0.0$, P^{st} is constant, and throughput is constant. All of the parts can fit in any of the buffers so no machine can ever become blocked. Furthermore, increasing the buffer size beyond N^p does not increase the probability that one of the buffers can become empty.

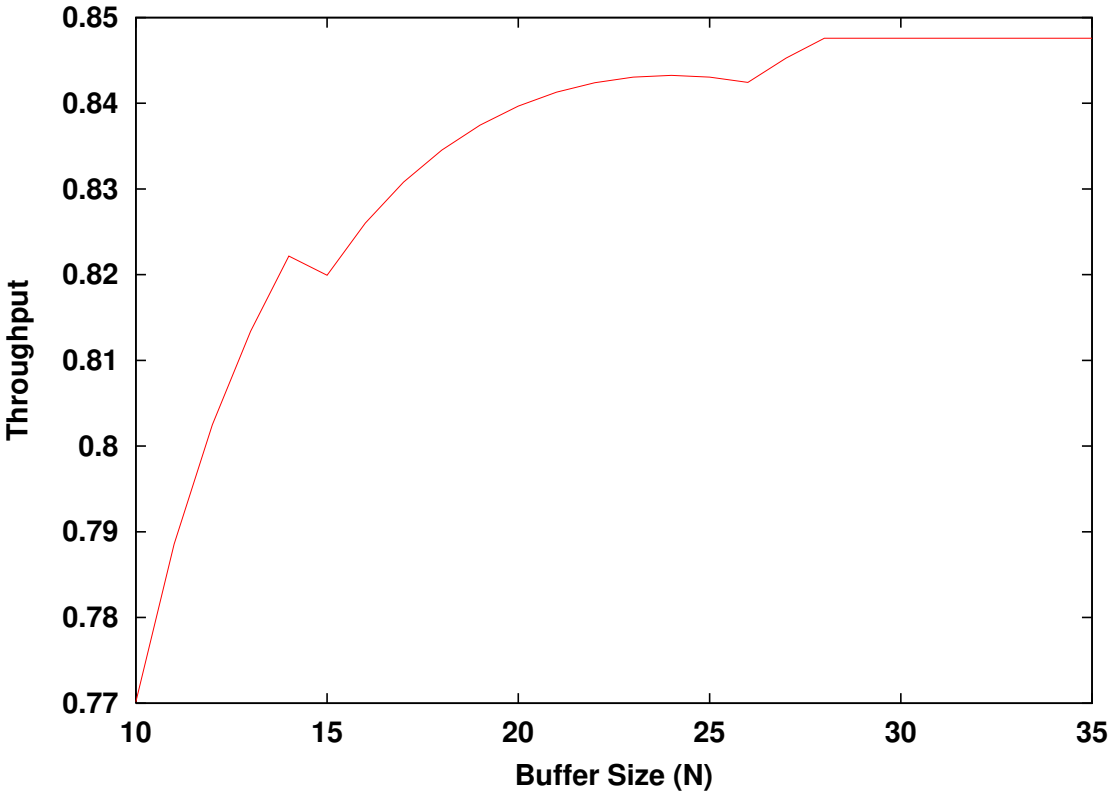

Figure 6-11: Analytical Throughput as a Function of Buffer Size

Chapter 7

Applying the Method

Here we describe an actual case study. For purposes of confidentiality, the name of the company, the details of the production process, and quantities such as operation time and production rates, are not discussed in this thesis.

7.1 Background

A Japanese electronics company produces a network connection device which is used to improve the quality of signals transmitted over long distances. The company is expecting an increase in demand. They are considering different ways of increasing the capacity of the factory and are interested in the effect that each of these options would have on their required in-process inventory level.

7.1.1 The Network Device

The components of the network connection device are a photo detector, amplifier, laser diode, and a protective case. Its purpose is to receive a signal from a fiber-optic cable, filter out noise, amplify the signal, and then transmit the signal to a fiber-optic output.

7.1.2 Production Process

Production of the network device consists of three separate phases: 1) assembly, 2) packing, and 3) aging and final testing. In the assembly phase, the photo detector, amplifier, and laser diode are bonded to a motherboard and placed in the protective case. The case is sealed in the packing phase and injected with nitrogen gas. The device is then baked and aged in order to cure the bonding compounds. Finally, the device is subjected to a series of electrical and climate tests. The production system is made up of 53 sequential stations.

7.2 Problem Statement

7.2.1 Objective

The company must increase the capacity of the factory in order to meet demand. To do so, they are considering two options. The first is to increase the number of overtime hours for the slower stations to the maximum level. The second option is to purchase one additional machine for the slowest station.

Currently, the factory is operated according to a CONWIP (constant work in process) control policy. In this type of policy, the total amount of inventory in the system is fixed, but there is no limit on the amount of inventory in any given buffer. The goal of the study was to provide the company with information about the WIP levels required in order to meet demand for each of the two options.

7.2.2 Limiting the Scope

We needed to consider two main issues in modeling the system: the size of the system and the characteristics of the individual machines. Due to limitations in the current computer implementation of our method, we are only able to deal with systems of about 10 machines.

The deterministic processing time model is intended to deal with single-part flow and machines with identical deterministic processing times. Unfortunately, production system for the network device is not quite so simple. Several of the operations are performed on various sized lots or batches rather than on a single part. Some of the processes are performed by multiple machines operating in parallel. In addition, operations require different amounts of time. Aging, for example, requires much longer than a simple visual inspection. To model the entire system, we would need to be able to consider the following factors:

(a) stations with different processing rates

(b) stations with multiple machines operating in parallel

(c) stations with longer work days

(d) stations that process lots/batches

(e) scrapping

Because of these factors, we decided to limit our study to the assembly phase of the production process. The assembly phase consists of 20 stations, each with only one machine. Additionally, there are no batch operations and no scrap at any of the stations.

7.3 Transforming the Data

In order to use the current version of the loop transformation and decomposition method, it was necessary to make several approximations. This section outlines the procedures we used to convert the data into a usable form.

7.3.1 Original Data

From the original data set, we had the the following information for each of the work stations in the assembly phase:

- number of machines at the station
- processing time T
- lot size
- scrap rate
- mean time to failure (MTTF)
- mean time to repair (MTTR)
- number of working hours per day

At this point, it is natural to model the system using the continuous processing time model described in [Ger94] in which machines have a failure rate p, a repair rate r, and a processing rate μ. After converting processing time, MTTF, and MTTR into common time units, we calculated the initial parameters p_0, r_0, and μ_0 for the stations as follows:

$$p_0 = \frac{1}{MTTF} \tag{7.1}$$

$$r_0 = \frac{1}{MTTR} \tag{7.2}$$

$$\mu_0 = \frac{1}{T} \tag{7.3}$$

7.3.2 Parallel Machines

When considering the option of buying an additional machine, we needed a way to model machines operating in parallel. Here, we assume that the parameters of the new machine are identical to those of the original machine. It seems reasonable that a station with two machines operating in parallel should produce twice as much as a

single machine. Furthermore, failures of the two-machine station occur less frequently than failures of an individual machine. In general terms, we calculated the parameters for a station with Y machines operating in parallel using the following equations:

$$p = \frac{p_0}{Y} \tag{7.4}$$

$$r = r_0 \tag{7.5}$$

$$\mu = Y\mu_0 \tag{7.6}$$

7.3.3 Overtime

For the stations that use overtime, we needed to adjust all three parameters. If a normal station works H_0 hours per day and a station with overtime works H_1 hours per day, we defined the *overtime factor* H as:

$$H = \frac{H_1}{H_0} \tag{7.7}$$

We then calculated the parameters of the station with overtime as follows:

$$p = Hp_0 \tag{7.8}$$

$$r = Hr_0 \tag{7.9}$$

$$\mu = H\mu_0 \tag{7.10}$$

7.4 Reducing the System

Due to limitations in our computer implementation of the method, it was impossible to model the system of all 20 stations with buffers between each station. Therefore, we attempted to model the system as one with fewer machines and buffers having approximately the same performance characteristics. To do this, we placed buffers before bottleneck machines and modeled the behavior of all machines between consecutive buffers using a single machine.

7.4.1 Identifying the Bottlenecks

To locate the bottlenecks in the assembly phase, we compared the isolated production rates of the machines. The isolated production rate ρ is calculated as:

$$\rho = \mu \frac{r}{r+p} \tag{7.11}$$

If all buffers are infinite, the production rate of a line is equal to that of the machine with the smallest ρ. Therefore, machines with relatively small values of ρ are bottlenecks in the system. We identified three machines with ρ very close to the desired production rate: M_7, M_{13}, and M_{18}. These were the same machines that the company had already identified as bottlenecks.

7.4.2 Zero-buffer Approximation

In our model, we placed buffers in front of each of the three bottleneck machines M_7, M_{13}, and M_{18}. To model the collective performance of machines between consecutive buffers, we used a variation of the zero buffer equations presented in [Ger94].

In a line with no buffers, no machine can operate at a rate greater than the processing rate of the slowest machine. Since we assume operation dependent failures, the effective failure rates of all machines must be scaled by the processing rate of the slowest machine. Therefore, we performed the following conversion for each machine M_i:

$$p_i' = p_i(\frac{\mu_{min}}{\mu_i}) \tag{7.12}$$

Using the values of p_i', we calculated the production rate of the zero buffer line $\rho_{\text{zero buffer}}$ as

$$\rho_{\text{zero buffer}} = \mu_{min} \frac{1}{1 + \sum_i \frac{p_i'}{r_i}} \tag{7.13}$$

This gave us the processing rate ρ of our new machine. The next step was to find values for r and p. We calculated r for the new machine as a weighted average of the r_i's in the zero buffer line using the following formula:

$$r = \sum_i (r_i \frac{p_i}{\sum_j p_j}) \tag{7.14}$$

Solving (7.11) for p gave us

$$p = r(\frac{\mu - \rho}{\rho}) \tag{7.15}$$

Using this approach, we reduced the assembly phase with a CONWIP control policy to a closed loop system made up of four machines and four buffers (see Figure 7-1).

M_1' models the behavior of the real machines M_1 through M_6. M_2' corresponds to M_7 through M_{12}, M_3' represents M_{13} through M_{17}, and M_4' models M_{18} to M_{20}.

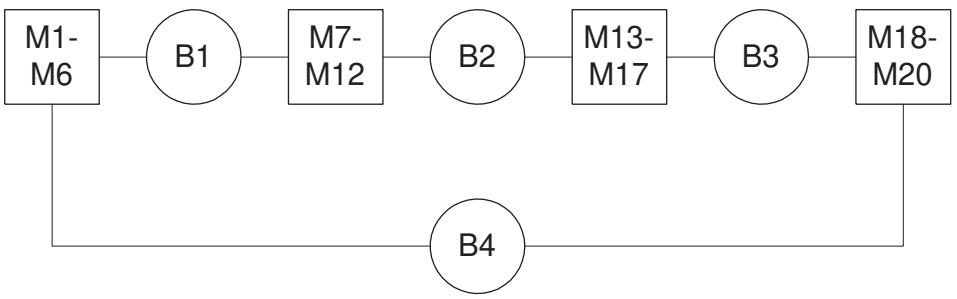

Figure 7-1: Illustration of reduced system

Buffers B_1' through B_3' are the actual parts buffers located directly upstream of the bottleneck machines. We can think of buffer B_4' as a finished goods buffer. If we did not include B_4', the failure of machine M_1' could prevent finished parts from exiting the system. Since this does not make sense in a real manufacturing system, we included B_4' in the model.

Tables I and II show the parameters of the reduced system for the max overtime and additional machinery options. The minimum isolated production rates are indicated in bold.

Table I: Parameters of reduced loop (max overtime)

Machine	r	p	μ	ρ
1-6	0.100793651	0.002234259	0.810810811	0.793227599
7-11	0.074530612	0.004023891	0.220971867	**0.209652763**
13-17	0.070373984	0.003939683	0.429850746	0.407062535
18-20	0.061402062	0.003397915	0.352653061	0.334161001

Table II: Parameters of reduced loop (additional machine)

Machine	r	p	μ	ρ
1-6	0.100793651	0.002234259	0.810810811	0.793227599
7-11	0.082868217	0.002668667	0.398976982	0.386529289
13-17	0.069109195	0.003828684	0.388059701	0.367689519
18-20	0.059074074	0.003238425	0.318367347	**0.301821571**

7.4.3 Processing Times

Because the method had not yet been implemented for the continuous processing time model, we needed to model the system using machines with equal processing times. To do this, we used a methodology developed by [Ger87b]. Gershwin shows that the performance of a machine with parameters p, r, and μ can be accurately approximated by two machines, M_1 and M_2, with parameters p_1, r_1, p_2, r_2 and zero buffer (see Figure 7-2).

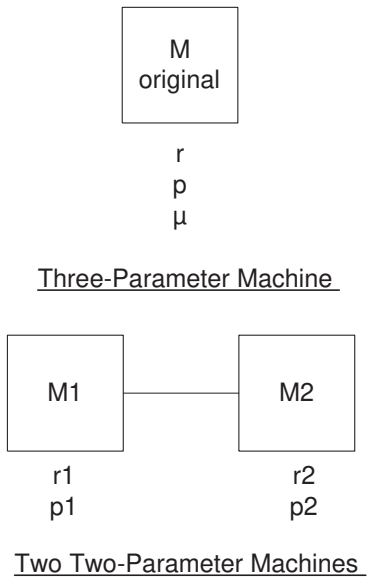

Figure 7-2: Representation of a three-parameter machine as two two-parameter machines

The first machine is used to represent the processing rate and the second machine models the failure behavior. The parameters are calculated so as to satisfy the following equations:

$$\frac{r_1}{r_1 + p_1} = \mu \tag{7.16}$$

$$r_2 = r \tag{7.17}$$

$$p_2 = \frac{p}{\mu} \tag{7.18}$$

$$r_1, p_1 \gg r_2, p_2 \tag{7.19}$$

[Ger87b] gives the following procedure for converting a line consisting of machines with different processing rates to one with identical processing rates:

(a) Divide all r's, p's, and μ's by the value of the largest μ. This scales the time unit so that the largest μ is equal to 1.

(b) Replace each machine with μ less than 1 by two machines whose parameters satisfy (7.16), (7.17), (7.18), (7.19).

Since our loop model cannot deal with buffers of size zero or 1, we inserted a buffer of size 2 between M_1 and M_2 in each case. While this adds some slight approximation, we felt that it was reasonable because the real buffers in the line are larger.

Performing this transformation on the reduced loop shown in Figure 7-1 gave us our final model of the system. It consisted of seven machines and seven buffers (see Figure 7-3).

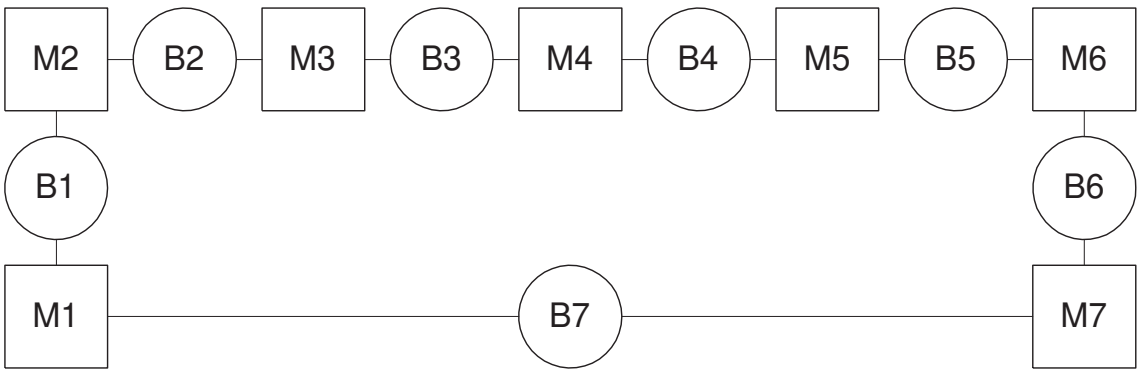

Figure 7-3: Illustration of final model

7.5 Solution Methodology

To study the effect of CONWIP size on production rate, we set all three of the real buffers equal the population size, which we varied from 7 to 74[1]. Figures 7-4 and 7-5 show the results for each of the two options.

In theory, the production rate of the loop should converge to the production rate of the slowest machine as the population is increased to infinity. The results from our model agree with this property. For the overtime option, the smallest ρ is 0.21045.

[1]This was the largest population that the computer program could handle.

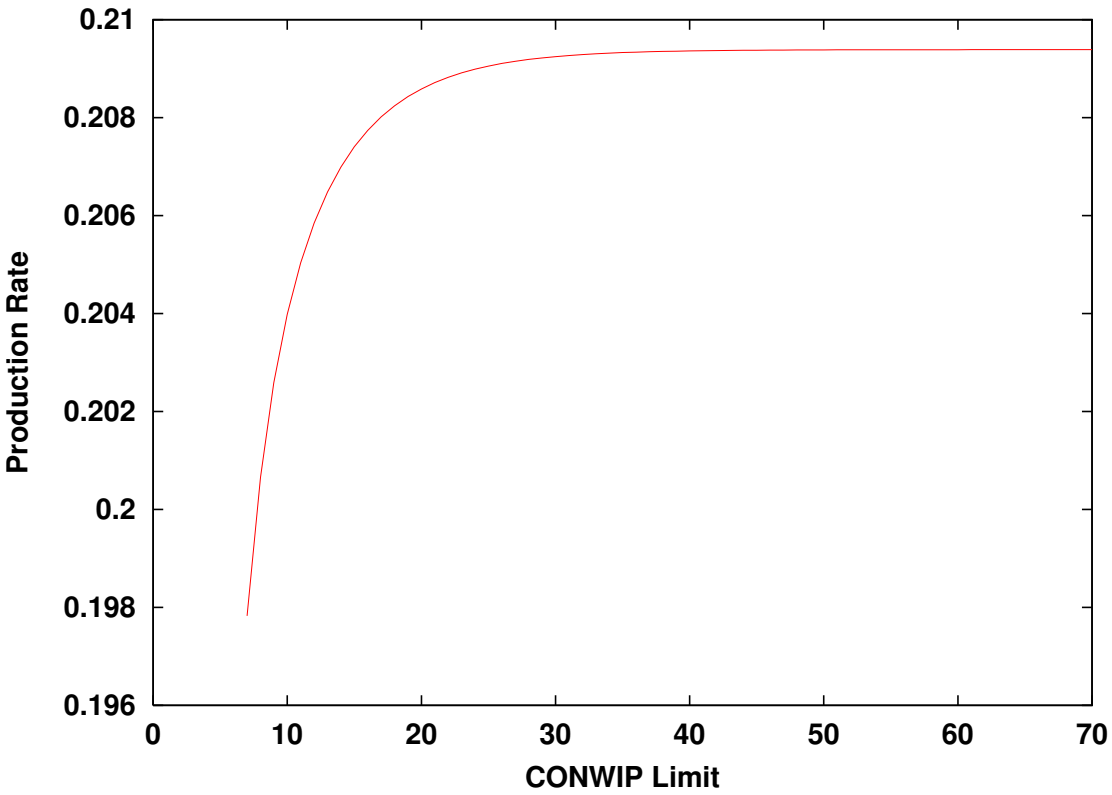

Figure 7-4: Production Rate as a Function of CONWIP Limit (Max Overtime)

According to the model, the maximum production rate is 0.20939. This is an error of 0.50%. The result for the additional machine option is similar. In this case, the smallest ρ is 0.30321 and the model gives the limit on production rate as 0.29912. Here, the error is 1.35% but it should be noted that production rate had not yet converged when the population was 74.

7.6 Conclusions

7.6.1 Case Conclusions

According to our model, each of the options can achieve the required production rate of 0.2 parts per time unit. If the company decides to use overtime hours to meet demand, they will need to maintain a WIP of 8. However, if they purchase an additional machine, demand can be met with essentially zero WIP.

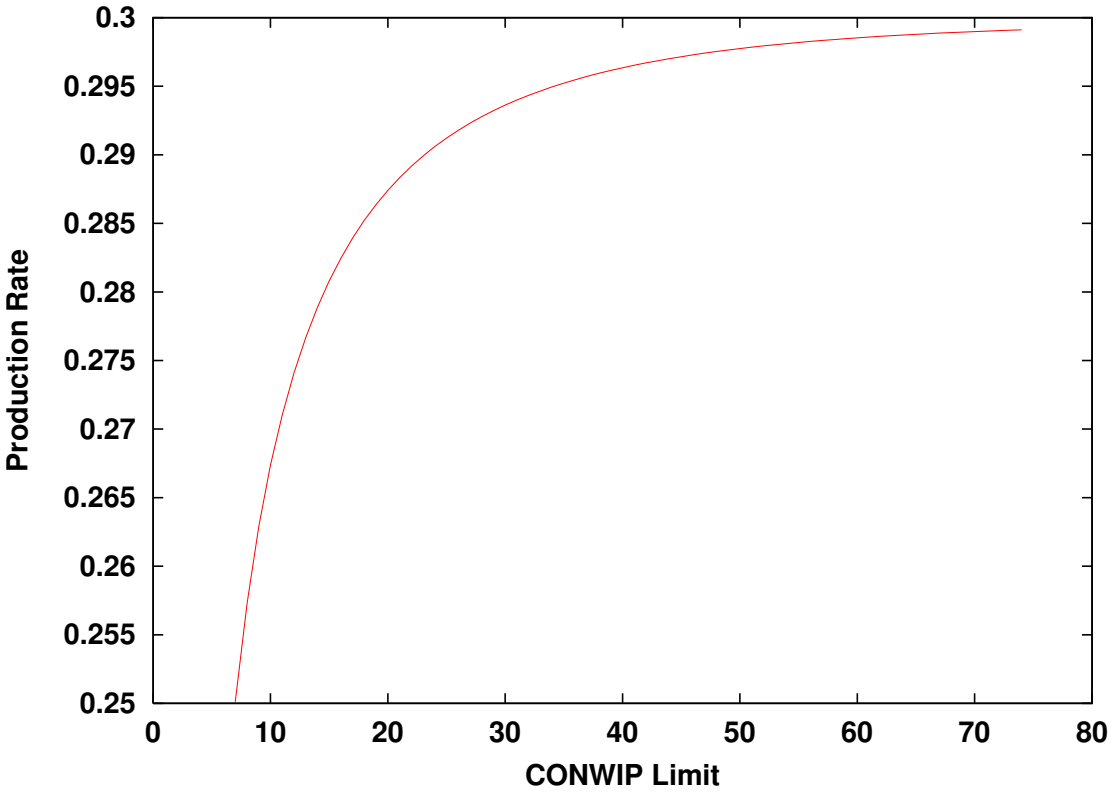

Figure 7-5: Production Rate as a Function of CONWIP Limit (Buy Additional Machine)

7.6.2 Needed Research

Additional research is needed to enable the method to better handle real world systems. The first step is to implement the loop transformation and decomposition for the the continuous material model. This would eliminate the need for the transformation described in Section 7.4.3. The models also need to be extended to deal with a wider variety of real-world issues such as batch operations and scrap rates. In conducting this case study, we considered possible methods for dealing with these problems. However, there were difficulties with implementation.

Lots and Batches In attempting to model machines that process lots, we looked at the behavior of parts exiting the machine. An observer positioned just downstream of the machine sees parts exiting the machine in bursts. The machine seems to be idle for a relatively long period of time and then rapidly processes the number of parts in

one lot. This behavior is similar to that of a machine with a very large processing rate and relatively small repair rate. In cases where operations are performed on a batch of W parts, we though about transforming the parameters by solving the following equations:

$$\frac{\mu}{p} = W \tag{7.20}$$

$$\mu\frac{r}{r+p} = W\mu\frac{r_0}{r_0+p_0} \tag{7.21}$$

$$\mu = \max_i \mu_i \tag{7.22}$$

Scrap Rates Scrap has the effect of reducing the amount of usable parts produced. One though we had on dealing with the scrap rate S was to reduced the processing rate using the following equation:

$$\mu = (1-S)\mu_u \tag{7.23}$$

Chapter 8

Conclusions and Future Work

The purpose of this research was to build on Maggio's work [Mag00], [MMGT00] to find a more practical general approach to evaluating closed-loop systems. Our transformation algorithm significantly reduces the complexity of large loops by eliminating multiple thresholds. The transformation and decomposition technique described in this paper provide extremely accurate approximations of average production rate.

There are several opportunities for future research:

(a) First is the issue of conservation of flow. Conservation of flow is not satisfied exactly in the examples we studied. Maggio [Mag00] observed the same phenomenon. Further refinement of the model and the algorithm might correct this.

(b) Another area where improvement is needed is in dealing with very small and very large populations. The method is not applicable when the number of parts (or holes) in the system is less than the number of machines or the size of the smallest buffer.

(c) The error in the average buffer level is high compared to the error in average production rate. This is typical with decomposition methods. The economic importance of average buffer level is motivation for improving the accuracy, particularly when the method is used to predict CONWIP performance.

In addition, there are several extensions to the method which would prove useful:

(a) The approach described here could be extended to multiple loop systems. This is of particular interest for evaluating the performance of systems operated under token-based control policies.

(b) The method could also be modified to deal with closed-loop systems in which multiple part types share a common set of resources. In this type of system, different part types compete for resources and therefore the production of one part interferes with the production of another.

(c) Another possibility is the combination of the first two items. The method can be extended to evaluate multiple loop with multiple part types.

Finally, we see from the case study that there is a need to extend our models to better deal with the complex issues found in real-world manufacturing systems. We need to find ways of dealing with phenomenon such as scrap, batch operations, and setups.

Appendix A

Three-Machine Loop Parameters

Table I: Loop 3.1

Machine	r	p	N
1	0.079034	0.015666	14
2	0.154824	0.008093	15
3	0.104084	0.028991	17

Table II: Loop 3.2

Machine	r	p	N
1	0.109964	0.013478	16
2	0.066043	0.006876	47
3	0.075848	0.003851	42

Table III: Loop 3.3

Machine	r	p	N
1	0.073385	0.000800	22
2	0.051737	0.014319	91
3	0.185704	0.042009	7

Table IV: Loop 3.4

Machine	r	p	N
1	0.108038	0.033304	10
2	0.118678	0.033130	34
3	0.134489	0.043302	37

Table V: Loop 3.5

Machine	r	p	N
1	0.115294	0.012335	13
2	0.154188	0.013918	27
3	0.128420	0.003913	26

Table VI: Loop 3.6

Machine	r	p	N
1	0.094680	0.014881	32
2	0.153075	0.015084	31
3	0.086905	0.007076	31

Table VII: Loop 3.7

Machine	r	p	N
1	0.145946	0.040302	6
2	0.154042	0.046828	8
3	0.187383	0.035154	6

Table VIII: Loop 3.8

Machine	r	p	N
1	0.145969	0.045026	24
2	0.190776	0.030020	10
3	0.180072	0.034301	15

Table IX: Loop 3.9

Machine	r	p	N
1	0.068014	0.019957	50
2	0.096967	0.004181	18
3	0.143706	0.008655	18

Table X: Loop 3.10

Machine	r	p	N
1	0.116440	0.006851	13
2	0.153762	0.012806	30
3	0.163042	0.023943	16

Table XI: Loop 3.11

Machine	r	p	N
1	0.132362	0.038206	30
2	0.177131	0.008438	27
3	0.099809	0.010842	20

Table XII: Loop 3.12

Machine	r	p	N
1	0.175235	0.003359	23
2	0.155171	0.002007	6
3	0.139068	0.015532	30

Table XIII: Loop 3.13

Machine	r	p	N
1	0.143831	0.002634	25
2	0.101839	0.007666	16
3	0.068283	0.010469	22

Table XIV: Loop 3.14

Machine	r	p	N
1	0.135596	0.017421	19
2	0.117353	0.014449	5
3	0.175223	0.058233	29

Table XV: Loop 3.15

Machine	r	p	N
1	0.192298	0.016537	26
2	0.056324	0.002992	25
3	0.131941	0.016659	10

Appendix B

Six-Machine Loop Parameters

Table I: Loop 6.1

Machine	r	p	N
1	0.144976	0.005013	23
2	0.122173	0.031507	26
3	0.070337	0.002061	53
4	0.071841	0.009081	18
5	0.173843	0.006039	19
6	0.135477	0.002155	27

Table II: Loop 6.2

Machine	r	p	N
1	0.051575	0.001270	38
2	0.154330	0.002909	3
3	0.185522	0.017783	10
4	0.165467	0.021359	9
5	0.113468	0.032113	31
6	0.084132	0.022852	57

Table III: Loop 6.3

Machine	r	p	N
1	0.099044	0.003600	18
2	0.168118	0.020465	25
3	0.092734	0.009901	11
4	0.153563	0.037536	29
5	0.099500	0.022009	10
6	0.113892	0.005789	21

Table IV: Loop 6.4

Machine	r	p	N
1	0.156161	0.038948	32
2	0.118474	0.038527	22
3	0.180505	0.008335	24
4	0.172510	0.046598	12
5	0.115746	0.035078	37
6	0.120613	0.028265	22

Table V: Loop 6.5

Machine	r	p	N
1	0.139099	0.009286	31
2	0.074966	0.002458	58
3	0.108032	0.032984	36
4	0.105859	0.027211	27
5	0.050272	0.008933	9
6	0.123161	0.015312	16

Table VI: Loop 6.6

Machine	r	p	N
1	0.054364	0.002379	86
2	0.151093	0.013050	7
3	0.167559	0.004043	7
4	0.187911	0.012212	20
5	0.081529	0.011213	33
6	0.197713	0.014628	14

Table VII: Loop 6.7

Machine	r	p	N
1	0.174954	0.014931	12
2	0.094903	0.014742	11
3	0.079639	0.006415	49
4	0.156236	0.013279	28
5	0.146070	0.042562	11
6	0.174957	0.031042	17

Table VIII: Loop 6.8

Machine	r	p	N
1	0.119150	0.019578	31
2	0.127030	0.031903	26
3	0.096766	0.028858	9
4	0.156114	0.048272	28
5	0.101547	0.007119	13
6	0.139220	0.038150	29

Table IX: Loop 6.9

Machine	r	p	N
1	0.154592	0.011436	8
2	0.177782	0.026871	17
3	0.051092	0.010684	74
4	0.051365	0.012852	60
5	0.113699	0.021514	36
6	0.148604	0.039191	8

Table X: Loop 6.10

Machine	r	p	N
1	0.052433	0.004137	54
2	0.054848	0.002557	61
3	0.189452	0.054197	18
4	0.174411	0.022313	24
5	0.141156	0.038437	21
6	0.122348	0.013056	7

Table XI: Loop 6.11

Machine	r	p	N
1	0.196819	0.014055	26
2	0.108697	0.011077	17
3	0.171942	0.004530	7
4	0.160102	0.024514	31
5	0.149351	0.048518	5
6	0.195518	0.047342	21

Table XII: Loop 6.12

Machine	r	p	N
1	0.124244	0.036404	11
2	0.101522	0.023950	28
3	0.076480	0.022765	63
4	0.156819	0.025186	4
5	0.098818	0.004277	28
6	0.141932	0.039537	13

Table XIII: Loop 6.13

Machine	r	p	N
1	0.090278	0.018959	21
2	0.107836	0.020524	45
3	0.118208	0.003318	4
4	0.113052	0.022836	21
5	0.085216	0.021297	57
6	0.149578	0.006272	33

Table XIV: Loop 6.14

Machine	r	p	N
1	0.066974	0.012704	46
2	0.125536	0.018545	34
3	0.198950	0.028777	14
4	0.142355	0.008065	4
5	0.056312	0.014029	44
6	0.178580	0.044827	24

Table XV: Loop 6.15

Machine	r	p	N
1	0.167547	0.027823	14
2	0.152081	0.012831	29
3	0.125324	0.021906	40
4	0.181046	0.007981	8
5	0.149910	0.024550	25
6	0.160957	0.052798	31

Appendix C

Ten-machine Loop Parameters

Table I: Loop 10.1

Machine	r	p	N
1	0.183191	0.00323241	19
2	0.0596206	0.0153859	13
3	0.190935	0.0393347	20
4	0.108374	0.0286632	10
5	0.181383	0.0168536	11
6	0.157299	0.0438734	17
7	0.119705	0.0153203	6
8	0.189016	0.024867	4
9	0.187734	0.0445411	6
10	0.150502	0.0474253	17

Table II: Loop 10.2

Machine	r	p	N
1	0.0934505	0.0286787	7
2	0.0923861	0.0145521	23
3	0.164453	0.0229854	26
4	0.0915718	0.00915303	7
5	0.0810424	0.0159579	43
6	0.150806	0.0125936	13
7	0.0524406	0.00865559	6
8	0.191048	0.0221118	5
9	0.141525	0.0448559	26
10	0.189569	0.0132722	9

Table III: Loop 10.3

Machine	r	p	N
1	0.096151	0.0165369	47
2	0.172758	0.0418478	19
3	0.124073	0.00953056	4
4	0.190301	0.0106644	8
5	0.154966	0.0333771	24
6	0.171514	0.00588372	7
7	0.108986	0.0358047	12
8	0.0713838	0.00673011	23
9	0.183247	0.036767	12
10	0.158687	0.0286408	12

Table IV: Loop 10.4

Machine	r	p	N
1	0.0668503	0.00716098	37
2	0.0829729	0.0162864	7
3	0.186004	0.00457479	24
4	0.135212	0.023754	15
5	0.170287	0.0166307	24
6	0.145281	0.0214392	23
7	0.140052	0.0113248	10
8	0.187683	0.0122668	22
9	0.170644	0.0307481	7
10	0.0885645	0.0118796	12

Table V: Loop 10.5

Machine	r	p	N
1	0.147844	0.017156	15
2	0.198878	0.015339	20
3	0.112477	0.00525383	37
4	0.0675318	0.016859	10
5	0.0584904	0.0115953	14
6	0.168995	0.0403546	8
7	0.185909	0.0478418	19
8	0.174411	0.0150609	25
9	0.13578	0.00370909	20
10	0.18496	0.0101551	15

Table VI: Loop 10.6

Machine	r	p	N
1	0.0509647	0.00763798	41
2	0.187989	0.0401954	23
3	0.102561	0.0191265	30
4	0.188803	0.0557934	5
5	0.147749	0.0278022	8
6	0.0811585	0.00332041	19
7	0.0857089	0.0253284	14
8	0.166954	0.0330174	20
9	0.0591614	0.0175425	8
10	0.19732	0.0526415	21

Table VII: Loop 10.7

Machine	r	p	N
1	0.136021	0.0132521	13
2	0.0677516	0.00158414	38
3	0.166637	0.0185526	27
4	0.136585	0.0134482	10
5	0.0849185	0.0043656	29
6	0.174155	0.0243614	4
7	0.109489	0.029863	11
8	0.128166	0.010146	40
9	0.0918017	0.0168728	9
10	0.0913906	0.027571	14

Table VIII: Loop 10.8

Machine	r	p	N
1	0.135556	0.040307	34
2	0.180555	0.05108	4
3	0.0745758	0.00559296	22
4	0.12501	0.027444	18
5	0.134653	0.0256811	29
6	0.147213	0.0449349	6
7	0.0845431	0.0258679	53
8	0.198731	0.0559985	6
9	0.168475	0.0512436	7
10	0.136649	0.0308736	18

Table IX: Loop 10.9

Machine	r	p	N
1	0.170459	0.0290001	5
2	0.0917568	0.00309716	13
3	0.0943664	0.00914266	5
4	0.130203	0.0133686	18
5	0.0938124	0.0187011	42
6	0.114833	0.0356584	6
7	0.0740705	0.017065	16
8	0.149879	0.0181412	22
9	0.0965006	0.00204711	41
10	0.143681	0.03267	31

Table X: Loop 10.10

Machine	r	p	N
1	0.0712139	0.0106448	15
2	0.0509503	0.00834283	46
3	0.121815	0.0124318	11
4	0.0886558	0.00904048	11
5	0.0912276	0.00608365	31
6	0.186618	0.00912488	3
7	0.084455	0.00851612	9
8	0.0970067	0.0223825	48
9	0.141548	0.0248338	9
10	0.0647839	0.00454106	16

Table XI: Loop 10.11

Machine	r	p	N
1	0.0772395	0.00901581	46
2	0.0759825	0.0101245	28
3	0.17106	0.0265557	14
4	0.195079	0.0349879	3
5	0.186056	0.00390329	8
6	0.191817	0.00998447	13
7	0.172097	0.00778906	11
8	0.110406	0.00271777	12
9	0.0815184	0.0133652	53
10	0.130644	0.028638	12

Table XII: Loop 10.12

Machine	r	p	N
1	0.143847	0.0190965	26
2	0.19883	0.00856757	17
3	0.0770671	0.00236775	28
4	0.123923	0.00638938	11
5	0.129643	0.0264424	30
6	0.0787548	0.0137422	58
7	0.151352	0.0288481	6
8	0.132754	0.0249265	17
9	0.0969942	0.0219435	7
10	0.165298	0.0291156	7

Table XIII: Loop 10.13

Machine	r	p	N
1	0.118125	0.0377436	28
2	0.149115	0.00443894	22
3	0.097255	0.0032413	11
4	0.0671956	0.00161092	9
5	0.0979067	0.0197175	44
6	0.134621	0.00733583	25
7	0.107397	0.00231324	13
8	0.144363	0.016121	19
9	0.0655627	0.00336603	16
10	0.062403	0.01211	26

Table XIV: Loop 10.14

Machine	r	p	N
1	0.17529	0.0261186	13
2	0.0637467	0.00389113	24
3	0.0575583	0.00831246	12
4	0.192179	0.0320509	6
5	0.0529493	0.00578869	33
6	0.083012	0.0141553	39
7	0.199729	0.0509779	14
8	0.178454	0.0118492	16
9	0.100352	0.0239082	49
10	0.108679	0.0059908	8

Table XV: Loop 10.15

Machine	r	p	N
1	0.0666081	0.00293728	48
2	0.178503	0.0501667	23
3	0.178165	0.0112919	19
4	0.195553	0.00617274	4
5	0.103127	0.0193468	31
6	0.160034	0.0438026	17
7	0.178563	0.0243703	21
8	0.0726214	0.00506734	25
9	0.0567353	0.00409832	21
10	0.0668132	0.00762629	6

Bibliography

[Aky88] I.F. Akyildiz. On the exact and approximate throughput analysis of closed queueing networks with blocking. *IEEE Trans on Soft Eng*, vol. 14, 1988.

[BFD92] A. Bouhchouch, Y. Frein, and Y. Dallery. Analysis of a closed loop manufacturing system with finite buffers. *CARs and FOF. 8th International Conference on CAD/CAM, Robotics and Factories of the Future*, vol. 2, 1992.

[FCD94] Yannick Frein, Christian Commault, and Yves Dallery. Modeling and analysis of closed-loop production lines with unreliable machines and finite buffers. Technical Report LAG No. 92-123, revised Nov 1994, Laboratoire d'Automatique de Grenoble, URA 228 CNRS, ENSIEG, BP 46, 38402 Saint Martin d'Hères, 1994. To appear in IIE Transactions.

[Ger87a] Stanley B. Gershwin. An efficient decomposition method for the approximate evaluation of tandem queues with finite storage space and blocking. *Operations Research*, 35:291–305, 1987.

[Ger87b] Stanley B. Gershwin. Representation and analysis of transfer lines with machines that have different processing rates. *Annals of Operations Research*, vol. 9, 1987.

[Ger94] Stanley B. Gershwin. *Manufacturing Systems Engineering.* Prentice-Hall, Inc., 1994.

[Mag00] N. Maggio. Nicola's thesis. you probably won't be able to find it, 2000.

[MMGT00] N. Maggio, A. Matta, S.B. Gershwin, and T. Tolio. An analytical method for the performance evaluation of closed loop production lines with multiple unreliable machines and finite buffers. will be published, 2000.

[OP90] R. Onvural and H.G. Perros. Throughput analysis of cyclic queueing networks with blocking. Technical report, CS Dept, North Carolina State University Raleigh, 1990.

[TG96] T. Tolio and Stanley B. Gershwin. Deterministic two-machine lines with multiple failure modes. Technical report, Politicnico di Milano - Dipartimento di Meccanica, 1996.

[TM98] T. Tolio and A. Matta. A method for performance evaluation of automated flow lines. *Annals of the CIRP*, 47, 1998.