Diffusion and Decomposition Approximations of Stochastic Models of Multiclass Processing Networks

by

Joseph Edward Nemec

B.A. in Mathematics, Manhattanville College,NY (USA), 1992
S.M. in Operations Research, M.I.T., MA (USA), 1997

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

Doctor of Philosophy in Operations Research

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 1999

Author ....
Sloan School of Management
December 1st, 1998

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

Accepted by ....
Thomas L. Magnanti, Institute Professor
Codirector, Operations Research Center

Diffusion and Decomposition Approximations of Stochastic Models of Multiclass Processing Networks
by
Joseph Edward Nemec

Submitted to the Sloan School of Management
on December 1st, 1998, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Operations Research

Abstract

This dissertation presents the analysis of two separate manufacturing systems, using two different approximation procedures, contained in two stand-alone parts.

The first part of the dissertations presents an analysis of a production-inventory system with finitely-capacitated production centers that produce multiple components to stock on a first come, first serve basis (FCFS). Components are stored in inventory until customer demand, where they are assembled into *products* or *kits*. Product orders that cannot be assembled from existing inventory are either backlogged, or simply lost, and we present an analysis of both cases. Components are then replenished in a one-for-one manner up to a pre-determined, fixed base stock level. We consider first the case where all components are processed at a single capacitated resource. The production center workload process is approximated as a one-dimensional Brownian motion, and explicit formulas for all relevant performance measures, such as product lead time and fill rate, are derived. Qualitative results are also presented, specifically the existence of a linear trade-off between promised lead time and initial base stock level reminiscent of the rule discovered by Glasserman and Wang [21]. The results are then extended to the case where components can be processed at one of more than one capacitated resources. The versatility of the linear trade rule is confirmed with extensive discrete-event simulation.

The second part presents an analysis of a synchronous transfer line (one in which all production times are identical) that produces multiple different part types on unreliable machines. The rates of failure and repair of the machines need not be identical. Inventory is stored between machines in finite buffers, or in work areas in the machines themselves, and different buffers may have different sizes. The machines operate according to a *static priority rule*, operating on the highest priority part whenever possible, and only operating on lower priority parts when unable to produce those with higher priorities due to either blockage or starvation. We present an analysis of this system by decomposing it into smaller sub-systems which are analytically tractable. An iterative algorithm is employed to solve the resulting decomposition equations. Estimates for performance measures, such as average buffer levels, throughput, and proportion of time due to blockage and starvation are presented, and compared to extensive discrete event simulation.

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

Acknowledgments

This dissertation represents the end of my formal education, and the beginning of my life in the private sector. As such, I find it an appropriate time to reflect on teachers,fellow students, and friends who have made the last twenty-four years of formal education bearable. Whether they know it (or not) and whether they are concerned with the details (or not), the work that has gone into this thesis was not done in isolation, and could not have been done without their support.

First and foremost, I wish to thank my family for the love and support that they have given me over the past twenty-eight years. To my brothers, John and Michael, my gratitude for the time we spent in California that week in March of 1996, and for all time given to me before and since; to Mom and Dad, for the unending patience they have shown me, especially when I have often behaved selfishly; to my grandparents Babi and Děda, for uncountably many good times in Canada, and Wilhelm and Gertrude Magnus, for giving me a love of mathematics and a sense of the family history; to Tony and Paula, for the many times in the other Cambridge; and to Bob and Lida, for their hospitality in Montréal and Kingsdown. And finally, to Aunt Marie and Uncle Ronnie, for always believing in me when many others didn't. They were always my strong supporters, encouraging me to turn my failures into learning experiences, and never to quit, and to them this dissertation is dedicated.

From the Operations Research Center, I wish primarily to thank Stan Gershwin for giving me the opportunity to work with him on both this dissertation, and my Master's degree. Always patient, he let slide my bad moods, and concentrated on the goal. I often feel that I have not deserved the patience he has shown me. Also, to Viên Nguyen, for the work contained in the first third of this dissertation, for the continuing financial support after she left for Booz·Allen & Hamilton, Inc. I would like to thank Larry Wein for signing off on Viên's part of the thesis, and Yashan Wang for agreeing to serve on my thesis committee. To Paulette and Laura, who never complained when I missed almost every deadline, and to Rob Freund, who helped me get out of a few jams, my eternal gratitude. Last, to many fellow students in the OR Center to who seemed not only to put up with, but sometimes even enjoy my bizarre brand of humor, in no particular order, my sincerest thanks for the laughs and friendship: Martin Haugh, David Markowitz, Jérémie Gallien, Amy Cohn, Marina Epelman, Andy Armacost, Leon Hsu, Rebecca D'Amato, Rodrigo Rubio, and any others with whom I've shared a laugh, a bad joke, or a discussion of Itô's Lemma, my thanks. Special thanks go to Mitch Burman and Jim Schor, for the Boeing work, the friendship, and for teaching me how to be a consultant.

From Manhattanville College, my thanks go to Edward Schwartz, Arnold Koltun, and George Kiernan for introducing me to the beauty of the rigors of mathematics, and to Marrion Perret for introducing me to the wonders of Shakespeare.

As for friends in general, from MIT there are always Yonald and Fletch, both of whom tried to kill me in the gym at one point or another, saved my workstation from my stupidities, and made life in this pressure-cooker bearable. Along with Will Moyne, James Kao, Thomas Lohman, and Professor Troxel, they taught me most of what I know about computers, and I can never thank them enough.

Outside of Academia, I have benefitted from the friendship of many. From home, although now dispersed, I could always count on Charles and Alice, Mike and Lucy, Katherine, and Archie, for diversions. From NYC, Jean-Emmanuel and Louise always provided a opportunity for a good dinner. From my travels in Europe, thanks all who put me up, showed me around, and helped me to forget (momentarily) what an enormous amount of work this has all been: in Sweden, Henrik and Eva, Martin and Ulrika; in England, Ian Holmes, Andrew Gauntlett, Steve and Atsuko Brown, Tim Fitzmaurice, Eryk Martin, and Phil Young; in Belgium, Filip Camerman; and in Germany, Elizabeth, Christiane,Bettina, Gerd Lüers, and the entire Lepsius family.

Finally, looking way back to High School, I wish to thank Jack Ayer for being tough on me, and never letting me get away with enough to let me think that I didn't need to keep my nose to the grindstone. I also wish to thank Steve Weber for keeping me out of Jack's clutches on many an occasion when he was howling for my blood. As for the friends I made there — Stein Soelberg, Steve Robinson, Derek Van Vliet, Miguel Santo Domingo, Bill Ryckman, Brian Mason and Alex Von Summer — thanks for keeping in touch, and for not telling too many people about the stupid things I did there.

Contents

List of Figures

List of Tables

Chapter 1

Stochastic Models of Manufacturing Systems

1.1 Introduction

Almost all economic activity in the modern world takes one of two forms: production of a product for sale, or performance of a service for a fee. This dissertation deals with first, presenting an analysis of two important kinds of manufacturing systems. We are interested primarily in developing intuition about the systems under investigation, as well as methods to predict their behavior, and estimate their performance.

Manufacturing activity in the United States— production of durable and non-durable goods — represents a large and critical part of the overall economy. It employs some 25 million people, producing almost 27% of the country's Gross Domestic Product. In some cases, such as semi-conductor manufacturing, the up-front capital investments required to begin large-scale production can be enormous; production facilities often cost well in excess of 1.5 billion dollars. Because of this, it is important to be able to understand and predict the overall behavior of the manufacturing system under consideration. It is important to have an answer to questions such as "Will I be able to meet my predicted demand rate" or "Given that I know the price that the market is willing to pay for the the product I intend to produce, will I be able to control my unit-production cost so as to make a decent profit?" If the demand rate and the production times of the products were known and fixed in advance, then the manufacturer could answer such questions with minimal effort. For example, if he knew that a customer would order ten units per day, and that each unit always took him exactly one hour to manufacture, he would have no difficulty in planning his production to meet customer demand. Unfortunately for the manufacturer, customer demands for his products do not usually come in a completely predictable manner. Compounding this, randomness inherent in the manufacturing process — due to such phenomenon as machine failures, produced parts not meeting certain quality standards, or production jobs taking longer than expected — make answering such questions at first glance a non-trivial endeavor. Some sort of rigorous scientific analysis that takes into account the variability inherent in the system is in order.

We are interested in a model of real-world phenomenon that takes in to account the possibility of randomness. *Queueing theory* is the generally-used term for the systematic study of non-deterministic dynamic systems where some sort of service is provided to customers who arrive and request it. Dating to the beginning of this century, the systems studied in the Queueing theoretic literature have evolved from simple systems where one type of customer arrives with a service request, waits in a queue until it is his turn to be served, and then leaves the system after having the service performed (see, for example, [41, 51]) to networks where customers with different service requirements pass through a sequence of service centers, often revisiting a service center multiple times (see [1], [37]). The models developed have led to powerful insights, as well as techniques for predicting the behavior of the systems modeled. The application of these models to evaluating manufacturing systems has been substantial (see [8]).

Unfortunately, even the simplest systems are amenable to exact probabilistic analysis only if one makes some assumptions about the nature of the randomness in the inter-arrival or service-length times of the customers. Because the necessary assumptions are often too restrictive for real-world purposes, the focus of more recent research has been to develop approximations that allow analysis of the systems in the face of the relaxation of simplifying assumptions. This dissertation is involved in developing approximations of Queueing models of two different kinds of manufacturing systems using two of the most commonly used approximation techniques, *diffusion approximations* and *decomposition*.

1.2 Diffusion Approximations

Diffusion approximations of Queueing systems, also called *heavy-traffic approximations*, are based on the theory of weak convergence of stochastic processes. In the absence of simplifying assumptions, such basic system processes as the number of parts in queue at a workcenter become complicated and analytically intractable, making it impossible to find an expression for the distribution function. A diffusion approximation seeks to replace the basic system processes with a process whose structure is well understood, and to provide conditions for when such a replacement is possible. The underlying idea is best seen by analogy to sums of random variables.

Given a sequence of independent, identically distributed (i.i.d) random variables, it is usually an extremely difficult problem to find the distribution of the sum of an arbitrary number of them. There are simple cases, of course — sums of i.i.d. exponential random variables have a gamma distribution; sums of i.i.d. normal random variables have a normal distribution — but these are the exception, not the rule. It is possible, however, to find an excellent approximation to the distribution of the sum under certain mild conditions that arise from the following fact. If we assume that the random variables have finite mean and variance, then as the number of random variables summed grows large, the distribution of the centered, normalized sum of the random variable converges to the standard normal distribution. In mathematical terms, given an i.i.d. sequence of random variables $\{X_n : n \geq 0\}$ with finite mean μ, and finite variance σ_2, then

$$\lim_{n\to\infty} Pr\left[a \leq \tfrac{X_1+X_2+\ldots+X_n-n\mu}{\sigma\sqrt{n}} \leq b\right] = \frac{1}{\sqrt{2\pi}}\int_a^b \exp^{-\frac{y^2}{2}}\,dy \tag{1.1}$$

This is the celebrated Central Limit Theorem.[1] It suggests that if we "unscale" the normalized sum of random variables in the proper manner, then we are able to find an approximation for the original sum. In fact, for large n, we have the following approximation:

$$Pr\left[a \leq X_1 + X_2 + \ldots + X_n \leq b\right] \approx \frac{1}{\sigma\sqrt{2\pi n}}\int_a^b \exp^{-\frac{(y-\mu)^2}{2n\sigma^2}}\,dy. \tag{1.2}$$

In essence, what we have done is replaced the distribution function of something that is hard to calculate (the sum of arbitrary i.i.d. random variables) with a distribution that we know a great deal about (a Gaussian distribution with a given mean and variance).

From the Central Limit Theorem, we see that it is possible to approximate the distribution of a complicated random variable with one that is known. A natural question is if this can be done for the system processes we are interested in estimating, such as the queue length processes of the network. As an example, consider the following situation: customers arrive randomly to a facility that makes d different kinds of components, and every customer order requests some amount of *each* component. Take the case where the system processes 2 different components, and arriving customers place orders for 2 type one components and 3 type two components one-

[1]In other words, the distribution of a centered, normalized sum of i.i.d. random variables (with finite mean and variance) converges to the standard normal distribution as the number of summed variables increases. This sort of convergence — where we are stating results about *distributions* of random variables — is known as *weak convergence*. Specifically, a sequence of random variables $\{X_n : n \geq 0\}$ is said to converge weakly to X if $Pr[X_n \leq z] \to Pr[X \leq z]$, for all continuity points of the limiting distribution function $Pr[X \leq \cdot]$. In this case we write $X_n \Longrightarrow X$. If we denote the standard normal distribution by $N(0,1)$, then the standard way of writing (1.1) is

$$\frac{X_1 + X_2 + \ldots + X_n - n\mu}{\sigma\sqrt{n}} \Longrightarrow N(0,1).$$

half the time, and orders for 1 type one component and 4 type two components the other one-half of the time. One question we would like to be able to answer is what is the probability that the number of type one component orders exceeds a certain threshold before some time T. If the inter-arrival times of the customer orders don't have some well-behaved distribution, then this question can be impossible to answer explicitly. In light of the Central Limit Theorem, one might ask if it is possible to estimate this probability under certain conditions. The result that it is possible to do so is known as *Donsker's Theorem* (see [12]).

Let $\mathbf{R}_+^d = \{(x_1, x_2, \ldots, x_d) : x_i \geq 0, i = 1, \ldots, d\}$. We can model the physical process as a $\mathbf{R}_+^d$-valued random walk in the following manner. Let $\{X_n : n \geq 0\}$ be an i.i.d. sequence of random variables taking values in $\mathbf{R}_+^d$, where $X_{n,i}$ is the size of the n^{th} component i order, and set $S_n = X_1 + X_2 + \ldots + X_n$. We call this the *cumulative order process*, where the j^{th} component of S_n, written $S_{n,j}$, is the total number of component j orders to be placed in the first n customer demands. Let θ be the d-dimensional vector of expected order sizes so that $\theta = \mathbf{E}[X_n]$, and Σ be the $d \times d$ covariance matrix so that $\Sigma_{ij} = \mathbf{E}[(X_{n,i} - \theta_i)(X_{n,j} - \theta_j)]$. We then define the *scaled cumulative order process*, written $\tilde{S}_n(t)$, as

$$\tilde{S}_n(t) = \frac{S_{\lfloor nt \rfloor}}{\sqrt{n}}, \tag{1.3}$$

where $\lfloor \cdot \rfloor$ rounds the value in the brackets down to the nearest integer.

Equation (1.3) has the following interpretation. If t is fixed, then (1.3) is just the sum of $\lfloor nt \rfloor$ i.i.d. vector-valued random variables, averaged by $\sqrt{n}$. As n gets large, we are summing more and more customer orders, but reducing the number of each order by a factor of $1/\sqrt{n}$. For example, let n be 100, and consider the time interval $[0, 10]$. Then (1.3) can be thought of as the sum of the first 1000 customer orders, but where we are reducing the size of each order by a factor of ten. In a sense, we are no longer counting orders from individual customers, but rather counting average orders of groups of ten customers. The intuition is that as individual customer orders lose their identity, the strong (and complicated) underlying inter-arrival behavior of the individual orders is smoothed out, while retaining the mean and covariance structure of the original process.

Let $N(0, I)$ denote the $\mathbf{R}^d$-valued multivariate normal random variable with mean vector 0 and the identity matrix as the covariance matrix, and let $\|\cdot\|$ denote the Euclidean norm on $\mathbf{R}^d$. Then Donsker's theorem (also known as the *functional central limit theorem*) states that if $\mathbf{E}[\|X_n\|] < \infty$, then as $n \to \infty$

$$\frac{S_{\lfloor n \cdot \rfloor}(t) - \theta n \cdot}{\sqrt{n}} \Longrightarrow BM(0, \Sigma), \tag{1.4}$$

where $BM(0, \Sigma)$ denotes d-dimensional Brownian motion with drift vector 0 and covariance matrix Σ, and "$\Longrightarrow$" refers to point wise convergence in the space $\mathbf{D}^d[0, \infty)$ of right-continuous functions with left limits and at most a countable number of discontinuities under the Skorohod $\mathbf{J}_1$ topology (see [3], [52]). The distribution function of $BM(0, \Sigma)$ is known to be just the d-dimensional multivariate normal distribution with drift vector 0 and covariance matrix Σ, from which all desired quantities can be calculated.

1.2.1 Literature Review

The idea presented in the preceding paragraphs — the analysis of complicated processes by scaling them into something more manageable — forms the basis for all diffusion approximations. Early work was done by Kingman [38, 39], who was the first to propose the analysis of a single-server queue with general inter-arrival and service times — the G/G/1 queue — with extensions made by Borovkov [5, 6]. Iglehart and Whitt [34, 35] extended the work by proving limit theorems for the G/G/s queue with multiple customer types, and provided extensions to the case where interarrival and service times were not assumed to be i.i.d., and allowed for batching. Whitt [58] made the further extension of allowing for random server selection. Harrison [23, 24] proved limit theorems for single-server queues in series, and queues where assembly operations were permitted. Harrison again considered single server tandem queues in [31]. Good surveys of this early work are found in Kingman [40],Iglehart [36], and Lemoine [43].

While early work concentrated on single-server queues, and simple networks, later work focused on more general networks. Reiman [50] provided limit theorems for queue length and sojourn time processes for open, single server queueing networks will a single customer class. Harrison and Williams [28] reformulated Reiman's work, and proved the existence and uniqueness of the limiting diffusion process. Closed networks with homogeneous customer populations were investigated by Harrison *et. al.* [30], and fork-join networks with homogeneous customer populations were investigated by Nguyen [44]. Extension of the homogeneous case to one with multiple different customer types has been difficult. Peterson [48] was able to extend the results to the case where customers always flow from lower numbered stations to higher numbers ones. Harrison and Williams [29] addressed the same sort of network, and gave an interesting interpretation of why some networks have joint density functions that break up into the product of the marginals (so-called *product-form solutions* in terms of the notion of quasireversibility introduced by Kelly [37] and expanded upon by Walrand [57]. Nguyen [45] was able to further extend the results to feedforward fork-join networks with multiple customer types. Unfortunately, the extension of from multiclass feedforward networks to ones where feedback is allowed has been quite difficult. Only recently have general results been available. Williams [59] has been able to prove limit theorems for for multiclass first-in-first-out (FIFO) networks where all customers at a given workcenter have the same processing distribution, and multiclass head-of-the-line proportional processor sharing (HLPPS) service disciplines. Excellent overall references for more recent developments in the theory may be found in Glynn [22], Flores [14], and Williams [60].

1.3 Decomposition Approximations

Automated production lines are a common feature in many factories, particularly those dedicated to high-volume manufacturing, such as automobile or camera production. Since it was first introduced by Henry Ford to speed the production of the Model T, the *tandem queueing network* or *flow line* has become one means of organizing high-volume manufacturing facilities. Before Henry Ford, automobile production was characterized by an immobile chassis and mobile labor. That is, an automobile in production would remain on a block, and laborers assigned to certain tasks, such as engine insertion, door assembly, and so on, would move to the half-assembled automobile, along with all the necessary equipment to do their job. Ford's innovation was to have the workers, and their equipment (which was often heavy and unwieldy) remain in place, and have the chassis move along a line from worker to worker. Many components and sub-components are still manufactured according to the flow line topology. In our case, we will be studying flow lines that are assumed to be fully automated. That is, both the flow of the material and the operations are automatic. To begin our analysis, we give a brief introduction to the current study of flow lines.

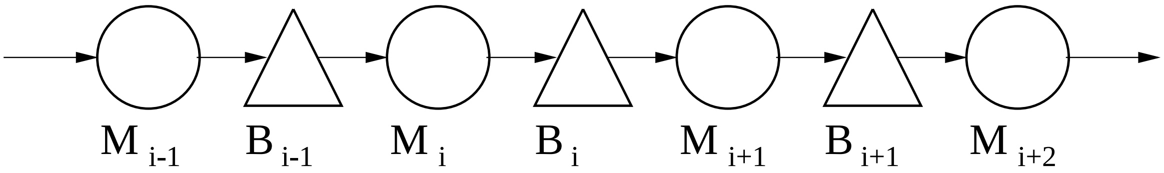

Figure 1-1: A basic abstraction of a flow line

1.3.1 Single-class Transfer Lines

Our basic model of a flow line, an example of which is depicted in Figure 1-1 has three main features. It has areas where work is done, often called *work centers*, *production centers*, or *machines*, depicted in the figure by the circles, and denoted by the letter M. Between the machines, there is space for storage of inventory, depicted in the figure by the triangles, and denoted by the letter B. We make the assumption that this space is limited — that is, the buffers are of finite size. The flow of the material through the resources is linear in that it begins at the first machine, is placed in that machine's downstream buffer upon completion of processing. It is loaded from there into the second machine, and so on. In particular, we assume that there is no re-entry in the flow. Thus, in the figure parts enter the system at machine M_1, then on to B_1, M_2, B_2, and so on. We assume that the production rates of each machine are deterministic and identical, and without loss of generality we assume that this rate is unity. We assume that the machines have *operation dependent failures*; that is, they can only fail while working on a part. We further assume that the machines have geometrically distributed up and down times: in any given time step, each machine that is operating has a fixed probability of failing, and each machine that is down has a fixed probability of being repaired. We do not impose the assumption that these probabilities are the same for any two machines.

We denote the size of buffer B_i by N_i, and the level of buffer B_i at time t by $n_i(t)$, and we set $0 \leq n_i(t) \leq N_i$. We do not require that the buffer sizes are equal. Each machine M_i has probability p_i of failing if it is working on a part, and probability of r_i of being repaired. It is important to observe that we assume that we can *only fail while operating on a part.* In particular, if a machine M_i is idle because it is either starved for parts (i.e. $n_{i-1} = 0$) or blocked (i.e. $n_i = N_i$), then it cannot fail. If a failure does occur, the part the machine failed on is placed back in the buffer, where it sits until the machine is repaired.

Given this setup, typical questions that are asked are "Which machine is the bottleneck?", "What is the production rate?", and "How much inventory is the system holding, on average, and where is it?" It turns out that exact analysis to answer such questions is not available, except in certain very small systems. We introduce such a system now.

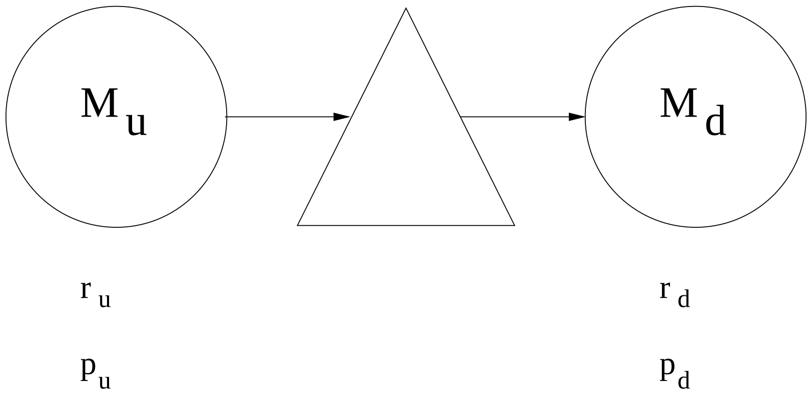

Figure 1-2: A two-machine transfer line.

We consider the case of a model of a transfer line with two unreliable machines, separated by a single, finite buffer with space for up to N parts. This system is depicted in Figure 1-2. The first machine is called the *upstream machines*, and is denoted by M_u. The second machine is called the *downstream machine*, and is denoted by M_d. The machines in the two-machine line share the same properties as the machines in the long line. That is, they have operation-dependent failures, with geometrically distributed up and down times. As before, we make no requirement that the probabilities of failure or repair are identical. Therefore, we denote the probability that M_u fails while producing a part by p_u, and the probability that M_d fails while producing a part by p_d. (We will see that our analysis of a multiclass transfer line requires that this assumption be dropped.) When machine M_u is down, we denote the probability that it is repaired in any given time step by r_u. Similarly, when machine M_d is down, it is repaired with probability r_d.[2] If we let α_u be one if M_u is operational, and zero if it is down, α_d be one if M_d is operational, and zero if it is down, and n, $0 \leq n \leq N$ be the level of the buffer, then the state of the two-machine transfer line can be completely characterized as $s = (n, \alpha_u, \alpha_d)$. If we denote the probability that M_u makes a part in a given time step by E_u, and the probability that M_d makes a part in a given time step by E_d. Since we provide no mechanism for the destruction or discarding of material, we must have that $E_u = E_d = E$. We call E the *production rate* of the transfer line. That is, E is the probability that the line produces a part in any given time step.

Under the assumptions above, the two-machine transfer line can be modeled as a discrete-time, discrete-state Markov process, the solution of which is tractable. The balance equations, solution technique, and probability mass function for the state s of the two-machine transfer line is given in Gershwin [16]. Two states have specific meanings. The first, $s = (0, 0, 1)$, represents the proportion of time that the downstream machines is starved. We denote this p_s, mnemonic for "probability of starvation". The second, $s = (N, 1, 0)$, represents the proportion of time that the upstream machine is blocked. We denote this p_b, mnemonic for "probability of blockage".

Unfortunately, transfer lines longer than two machines are not, in general, analytically tractable. Gershwin and Schick [20] managed to find solutions for three-stage lines with small buffer sizes, but the technique is unwieldy, and does not generalize to longer lines or larger buffers. Since most factories have transfer lines with many machines, other methods must be found. One method which has proved particularly robust for transfer lines with finite buffers and unreliable machines is *decomposition.*

1.3.2 Decomposition of a Long Single-class Transfer Line

Since two-machine transfer lines are analytically tractable, and transfer lines with more than two machines are not, the question poses itself: how can we use the robust solutions for two-machine lines to analyze longer lines? The answer is to decompose the long lines into shorter ones, analyze the pieces, and patch the shorter lines back

[2] The initial source of this notation is French, with p standing for *panne* and r standing for *repair*, the French for *failure* and *repair*, respectively.

together in a way that the performance measures of the longer line are approximated by the performance measures of the shorter sub-lines. This decomposition method is depicted graphically in Figure 1-3.

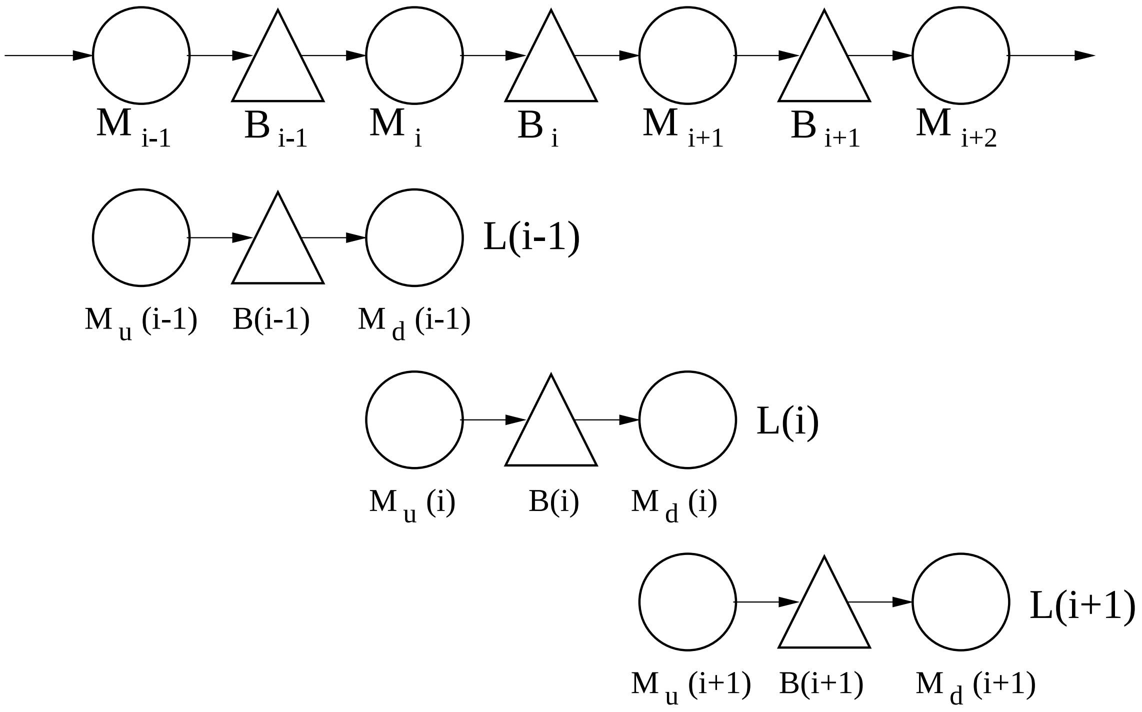

Figure 1-3: Decomposition of a long transfer line.

The idea is the following. We suppose that there is an observer in each buffer in the line. This observer watches the flow of parts entering and leaving the buffer that he is in. Inflow may stop for two reasons: the machine directly upstream of the buffer may have failed, or some machine further upstream may have failed, starving the line downstream of it. Similarly, outflow may stop for two reasons: the machine directly downstream of the buffer may have failed, or a machine further downstream may have failed, blocking the line upstream. We assume that the observer in the buffer is unable to determine the reason that inflow or outflow has ceased, and we adjust the parameters of a two-machine line so that judging by the behavior of the inflow and outflow, the observer in the buffer is unable to tell whether or not he is in a buffer in the real line. The remainder of this section is a review of the single part type decomposition, and it is taken from Gershwin [16].

In order to calculate the parameters of the two-machine lines that act as the decomposition of the long line, let us observe that for a long line with K machines, we will have $K-1$ buffers, and hence $K-1$ two-machine sub-lines. We recall that each two-machine line has five different parameters associated with it: the buffer size

N, the repair and failure probabilities of the upstream machine, and the repair and failure probabilities of the downstream machine. For every two-machine sub-line, we shall take as the buffer size the size of the buffer in the real line that the two-machine line is representing. This reduces the number of parameters we need to calculate for each two-machine sub-line to four. Thus, for a decomposition of a long line with K machines, we will have $4(K-1)$ unknowns. It remains for us to come up with $4(K-1)$ equations in these unknowns in order to guarantee a unique solution.

In order to derive the necessary equations for the decomposition, let us begin by observing a necessary property of the line: since no mechanism allows for the destruction of material, the production rates at each machine in the line must be identical in steady state. That is, if E_i is the production rate of M_i, then

$$E_1 = E_2 = \ldots = E_k. \tag{1.5}$$

We call this property *conservation of flow*. Since we require this property to hold for the real machines, we would also expect it to hold for the two-machine sub-lines. If we let $E(i)$, $i = 1, \ldots, K-1$ denote the production rate of the i^{th} two-machine sub-line, we observe that

$$E(i) = E(1), \quad i = 2, \ldots, K-1. \tag{1.6}$$

Recalling from Section 1.3.1 that the production rate of a two-machine line is determined entirely by the five parameters of the line, we observe that conservation of flow gives us $K-2$ equations.

The next property we can observe about the machines in the real line is known as the *flow rate-idle time* relationship. Define e_i to be the *isolated production rate* of machine M_i. Then e_i is the production rate of M_i if the machine were never impeded by blockages or starvations. The isolated production rate is given by

$$e_i = \frac{r_i}{r_i + p_i}. \tag{1.7}$$

Since M_i can be blocked or starved, however, its actual production rate is given by

$$E_i = e_i Pr\left[\{n_{i-1} > 0\} \cap \{n_i < N_i\}\right]. \tag{1.8}$$

In order to calculate this quantity, we must make a crucial assumption:

Assumption 1 *The probability that a machine M_i is simultaneously blocked and starved is negligible. That is,*

$$Pr\left[\{n_{i-1} = 0\} \cap \{n_i = N_i\}\right] \approx 0. \tag{1.9}$$

The justification for this assumption is the observation that the event that a machine is simultaneously blocked and starved can only be reached from states in which $n_{i-1} = 1$ and $n_i = N_i - 1$ by a transition in which the machine M_{i-1} is not producing (either down or starved), the machine M_{i+1} is not producing (either down or blocked) and machine M_i is up and producing. An immediate consequence of Assumption 1 is that we can rewrite the flow rate-idle time relationship given in Equation (1.8) as the following approximate flow rate-idle time relationship:

$$E_i \approx e_i(1 - Pr\,[n_{i-1} = 0] - Pr\,[n_i = N_i]). \tag{1.10}$$

We now observe that the event $n_{i-1} = 0$ is identical to the event $p_s(i-1)$ — the probability that the buffer in two-machine sub-line $L(i-1)$ is empty. Likewise, the event $n_i = N_i$ is equivalent to the event $p_b(i)$, the probability that two-machine sub-line $L(i)$ is blocked. Therefore, another expression for the production rate of a given two-machine sub-line is:

$$E(i) = e_i(1 - p_s(i-1) - p_b(i)), \quad i = 2, \ldots, K-1 \tag{1.11}$$

This gives another $K - 2$ equations.

The third set of equations that are required for the decomposition are the *resumption of flow equations*. These are expressions for the parameters $r_u(i)$ and $r_d(i)$, the repair probabilities for each two-machine sub-line, where $i = 1, \ldots, K-1$. Expressions for these terms are derived by observing that the repair parameters are conditional expectations that a machine makes a part in time step $t+1$, given that the machine did not make a part in time step t. We will not derive these expressions here, but rather refer the reader to Gershwin [16] for details. The resumption of flow equations give a further $2(K-2)$ equations.

Finally, the last set of equations are the boundary conditions. Specifically, for the upstream machine of two-machine sub-line $L(1)$, we set $r_u(1) = r_1$ and $p_u(1) = p_1$. Likewise, for the downstream machine of two-machine sub-line $L(K-1)$, we set $r_d(K-1) = r_K$ and $p_d(K-1) = p_K$. The boundary conditions give a total of four more equations. So we have a total of $4(K-1)$ equations in $4(K-1)$ unknowns.

It remains for us to solve for the two-machine sub-line parameters. Unfortunately, this is not easily done directly, as the equations derived for the decomposition are non-linear. Therefore, an iterative numerical algorithm is used to calculate the parameters. This algorithm was formulated by Dallery, David, and Xie [62]. It is very efficient, converging in most realistic cases.

1.3.3 Literature Review

The decomposition of the discrete-time processing line described above was extended to fork/join, feed-forward networks by Gershwin [15]. Under the same assumptions, a decomposition method for lines with rework loops and imperfect yield was developed by Helber [33]. In the case of random processing times,Gershwin and Berman [17] extended the two-machine solution to the case where the processing times of the machines are exponential with different rates for each machine. Building on this two-machine model, Gershwin and Choong [18] formulated a decomposition for a long transfer line similar in structure to the single-class line discussed in Chapter 1, but one where the processing times at the machines were exponentially distributed, and could have different processing rates. Gershwin and Schick [19] extended the two-machine solution to the case where the processing times are deterministic, but different at each machine using a fluid model. This two-machine solution was used as the basis for a decomposition of an asynchronous, deterministic processing time processing line by Burman [7]. Other cases are discussed in the survey paper by Dallery and Gershwin [11].

1.4 Thesis Outline

This dissertation is organized as follows. Chapter 2 presents an analysis of a production/inventory system with a single capacitated resource that produces multiple different components to stock on a first-come, first-serve basis. Components are stored in finished goods inventories with fixed initial base stock levels, and are assembled into products upon customer demand. Products that cannot be filled are either backlogged, or simply lost. Both cases are considered. Heavy-traffic limit theorems are presented for both cases, the limiting distributions identified, and formulae for all relevant performance measures are derived. A trade off rule reminiscent of that discovered by Glasserman and Wang [21] is discussed. Numerical results for the analysis done are then presented in Chapter 3. Chapter 4 extends the results of the backorder case to one in which more than one capacitated resource is allowed. A heavy traffic limit theorem is discussed, the limiting diffusion process is classified, and an extension of the linear tradeoff rule is derived. Numerical experiments are done to test the validity of the rule. Chapter 5 introduces a Markov model of a processing line with two different part types, flexible, unreliable machines, and finite intermediate buffers. The decomposition of the long line into smaller, tractable two-machine lines is discussed. Chapter 6 presents the analysis of the Markov chain for the two-machine lines. The decomposition derivation is derived in Chapter 7. An algorithm to solve the decomposition is presented in Chapter 8, as are numerical results concerning the accuracy of the decomposition, and the qualitative behavior of the system.

Chapter 2

A Production-Inventory System with a Single Capacitated Production Resource

2.1 Introduction

In this part of the dissertation we present an analysis of a production-inventory system in which components are produced to stock but final products, which may require multiple components, are assembled only upon customer demand. A single capacitated production resource produces all components, and final products are assembled by an uncapacitated resource. Demands that cannot be filled from inventory are either backlogged, or simply lost. The main result is a heavy-traffic based method for estimating inventory levels, fill rates, and lead times. The heavy-traffic regime can be interpreted as an asymptotic regime where utilization of the production resource is close to 100% and base-stock levels are high relative to utilization. The analysis leads to a linear trade-off rule for inventory and lead times, reminiscent of the rule recently discovered by Glasserman and Wang [21].

2.1.1 The Model

In our model, we consider a system that produces c different components. Components are stocked in finished goods (FG) inventories, represented by triangles. All inventories have finite non-negative fixed initial base-stock levels. We consider two different cases. In the first, we allow each of the component inventories to be infinitely backlogged. We refer to this as the *full backorder case*, or often just the *backorder case*. In the second, we only allow orders for components to be filled if there are enough components held in inventory. We refer to this as the *lost sales case*.

We envision product orders arriving from K different *product streams*. These streams can be interpreted as different kinds of individual customers; large corporate purchasing offices, small independent businesses, or individual consumers, for example. Each stream brings with it an arbitrary number of different kinds of product orders and the probability that the order placed is of a certain sort. As an example, suppose we have a system with two different kinds of customers placing orders: a single large corporation on the one hand, and many small businesses on the other. We model this situation as a system with two independent product arrival streams, one modeling the purchasing habits of a large corporation, and another to capture the behavior of individual small businesses. We might expect the purchases from the corporate customers to come infrequently, and be relatively constant in their order mix. Then the arrival rate for orders from the first stream would be low, but the variability in the component-level orders would also be low, and we would expect a single product order to contain large component orders. On the other hand, orders from small business customers would tend to be smaller, but different businesses would have different component mixes to serve individual needs. We would then expect the product orders from the second stream to have a high arrival rate and a large amount of variability in the component orders, but the component batch orders would be small. We assume that orders from different product order streams (i.i.d.), inter-arrival times, with mutually independent general distributions across product streams. Each product brings with it an order for one or more of each of the components produced by the system. If all components requested are available at the time of the product order, the order is assembled instantaneously by the assembly machine. In this case we say that the product order is *filled immediately*. When backlogging is allowed, if at least one of the components requested is unavailable, the product order is backlogged. In the lost sales case, if any one of the requested component orders is unavailable at the time of product order, the product order is *rejected*, and the customer leaves empty-handed. Product orders that are not rejected are said to be *accepted*.

In the case where a product order is either filled immediately or backlogged, component orders are sent to the workstation, where they are processed in a first-come first-serve (FCFS) manner. The workstation is assumed to be *multiclass* in that it is capable of producing all the components in the system, and is *completely flexible* in that there is no setup time or cost incurred when the workstation switches production between different components. The processing times for the component orders are assumed to be i.i.d. random variables with general distributions, and are mutually independent across different components. When the workstation completes work on a component, it is placed instantaneously into its FG inventory, where it is either stocked, or fills a backlogged product order.

While there is a large body of literature for determination of performance measures in multi-component systems with no correlation across component demands, investigation of multi-component systems where demands are correlated over components has been a recent endeavor. Hausman, Lee, and A. Zhang [32] consider

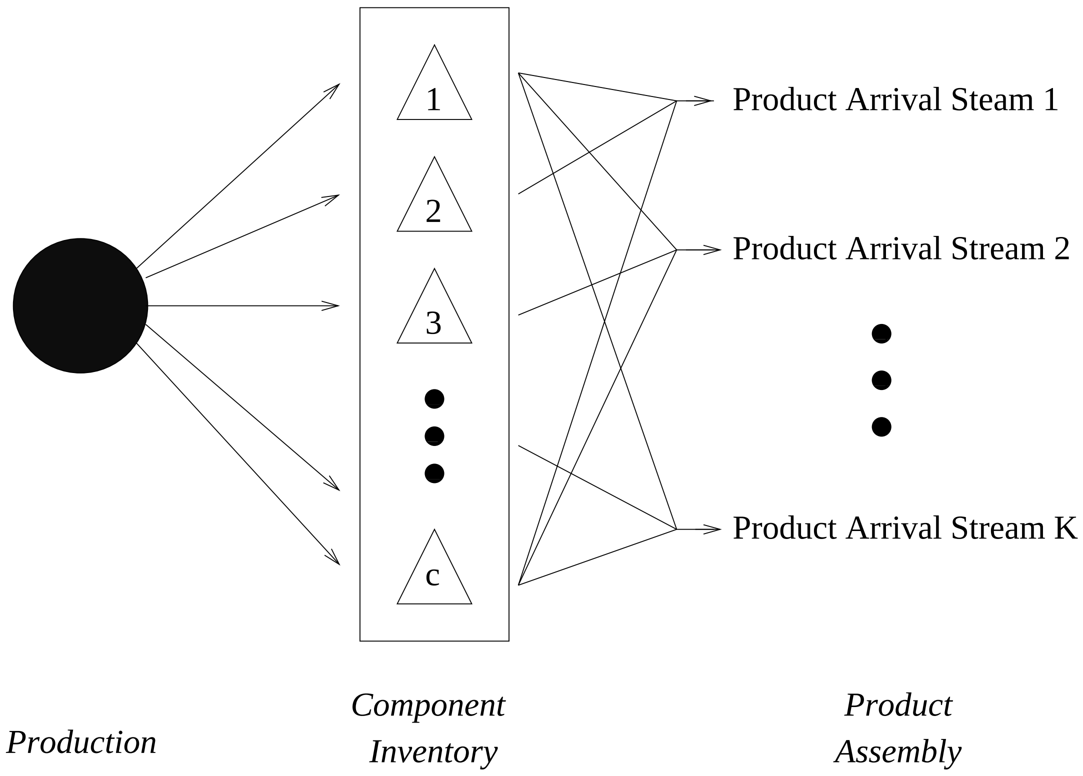

Figure 2-1: A production-inventory system with a single capacitated resource.

a multi-component inventory system with dependent component demands where a single, generic product requires a random subset of all the components. Each component is managed independently through a periodic-review order-up-to inventory policy. The order-up-to level for each component is the same over all periods, and replenishment lead times are deterministic integer non-negative multiples of the period length. They obtain order-fulfillment probabilities within a pre-specified fixed time limit under a first-come, first serve (FCFS) order fulfillment policy. A. Zhang [63] extends this framework to multiple products. Song [53] considers a similar system with fixed initial base-stock levels and Poisson product demands that request up to one unit of each component, but may request more than one component. Product orders that are not immediately filled can be backlogged, and are filled in a FCFS manner. There is a constant, positive replenishment lead time for each component. The lead times for different components need not be the same. Under these assumptions, Song obtains an efficient method of computing an exact order acceptance rate expression. Song also investigates the relationship between component acceptance rates and order acceptance rates. Song, Xu, and Liu [54] expand upon this by considering a similar model with an identical demand process, but with stochastic component replenishment lead times. Each component is produced at a separate facility in a FCFS manner, and the processing times are i.i.d. exponential random variables where the rates for each component need not be the same. Song, *et al.* provide methods for calculating both component and order acceptance rates. R. Zhang [64] considers a similar system where the the exponential assumption of the processing time at the facilities is relaxed to allow of i.i.d. processing times with positive mean, and considers expected total time delay of a customer. Glasserman and Wang [21] consider a production-inventory model similar to that of Song, but allow for general inter-demand and processing distributions. They define the fill rate of a product as the proportion of product orders that are completed within a pre-set time limit. They use asymptotic analysis to investigate the relationship between lead time and inventory, finding an essentially linear trade off between the two at high fill rates.

There are three issues that make analysis of the system we consider difficult. First, the buffer levels in the finished goods inventories are highly correlated. Second, the lack of any Poisson assumption on the inter-arrival times of the product orders means that the demand process seen by the individual finished goods inventories is not a renewal process. Third, the machine is multiclass, with general service distributions for each class of component it processes. In light of these difficulties, traditional Queueing methods do not provide satisfactory analysis of the system. Therefore, we have taken the *heavy-traffic* approach to approximate the system discussed in Chapter 1.

Our primary quantity of interest is the *workstation workload process*, denoted $W(t)$, that represents the amount of work at the workstation at time t. We show that if we scale the workstation workload process in the "proper" fashion, then under certain conditions, the resulting scaled process, denoted $\tilde{W}^n$, can be well approximated by a reflected Brownian motion on an interval, where the mean and variance of the Brownian motion are determined entirely from the first and second moments of the inter-demand and processing times.

There is now a wide body of heavy-traffic literature that deals with multiclass stochastic systems, as described in Chapter 1, and we refer the reader there for general details. However, two papers of note that have used heavy-traffic to investigate the behavior of systems similar to ours are both due to Nguyen. Nguyen [46] considers a two-station mixed Queueing network that behaves as both an open and a closed network in that there are customers that arrive exogenously to one station that are processed out (the "open" customers), and customers that circulate perpetually between the two stations (the "closed" customers). She generalizes this to multiple open and closed customers in [47], considering a very similar system to the one we investigate, but does not allow for correlation across the buffers. The notation used is similar to that of Nguyen [47]. Models of systems of the sort we study were also constructed by Rao and Suri [49]. While our work concentrates on performance analysis, research focusing on scheduling policies for production/inventory systems of the sort we address have also been undertaken recently. Two of note that we mention are

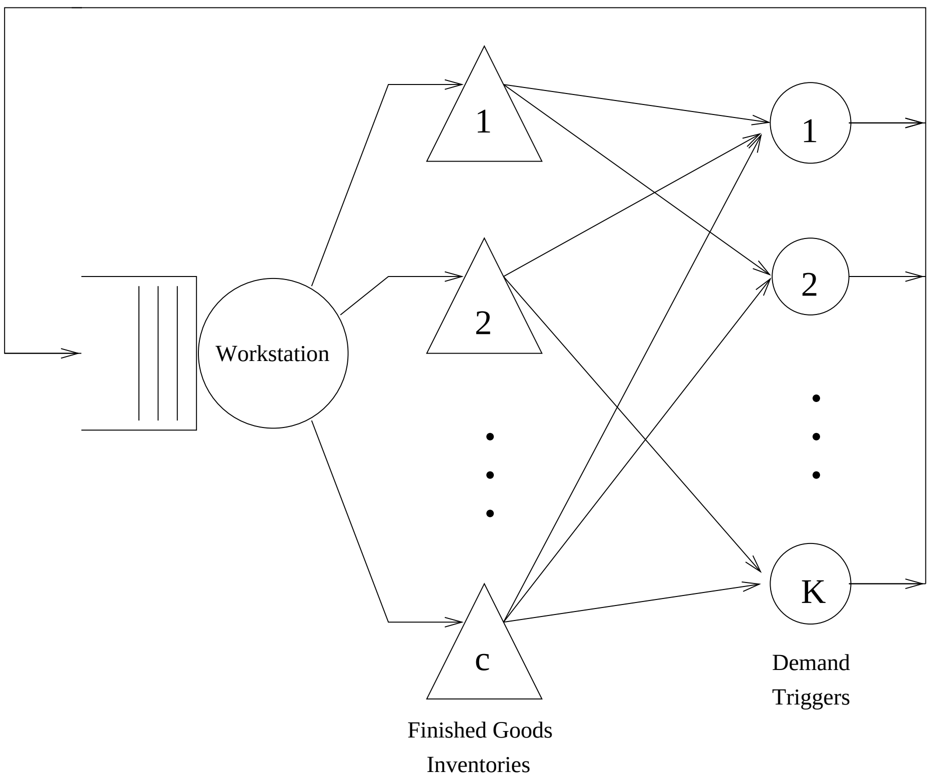

Figure 2-2: The modeling method.

those by Duenyas and Keblis [13], which used more traditional Queueing-theoretic techniques, and Tetzlaff and Kushner [55]. Tetzlaff and Kushner study essentially the same system that we do, concentrating on optimal control in a limiting regime. Their work concentrates on proving a very general heavy-traffic limit theorem, and showing that their optimal control formulation holds for very general systems. In many respects, their work can be seen as the other side of the same coin.

2.2 Primitive Network Data

We propose to study the both the backorder and lost sales cases of the production-inventory system depicted in Figure 2-1 by considering $K+1$-machine system depicted in Figure 2-2. Machine 0 is taken to be the workstation, and Machines $1, \ldots, K$ are "demand triggers" that capture the behavior of the various product order streams. The busy periods of the demand triggers correspond to periods when all of the demand trigger's FG inventories are capable of filling component orders. For the system where backlogging is allowed, the demand triggers are always busy.

For the lost sales case, the busy periods of demand triggers are exactly those times when *all* component orders of the corresponding product order can be filled. Because of this, we assume that each demand stream brings with it only *one* product order.

In the full backorder case, each product arrival stream has $\ell = 1, \ldots, L(k)$ possible product orders, where the probability that a product order from stream k is type ℓ is denoted p_{kl}. We let $q^k_{i\ell}$ denote the size of the component i order of product ℓ from stream k. We assume that $q^k_{i\ell}$ can be any positive integer. In the full backorder case, we define q_{ik} to be the expected component i batch order size for a product order from stream k. That is,

$$q_{ik} = \sum_{\ell=1}^{L(k)} p_{k\ell} q^k_{i\ell}. \tag{2.1}$$

In the lost sales case, component i orders from stream k are denoted by q_{ik}, where q_{ik} is zero if product k does not require component i, and a positive integer representing the number of components needed if it does. Notice that in the lost sales case, q_{ik} corresponds to the *exact* component i batch order from stream k, while in the full backorder case it corresponds to the *expected* component i batch order. We denote the long-run arrival rate of product orders of class k by α_k. The aggregate demand rate for component type i is then

$$\lambda_i \equiv \sum_{k=1}^{K} q_{ik} \alpha_k. \tag{2.2}$$

The mean processing time for component type i at the workstation is m_i. Let $\{u_k(\ell)\}$, $k = 1, \ldots, m$, $\ell = 0, 1, 2, \ldots$ be a sequence of i.i.d. random variables defined on a common probability space $(\Omega, \mathcal{F}, P)$ with $u_k(0) = 0$, $E[u_k(\ell)] = 1$, and $Var[u_k(\ell)] = c^2_{a_k} < \infty$. Similarly, let $\{v_i(\ell)\}$, $i = 1, \ldots, m$, $\ell = 1, 2, 3, \ldots$ be a sequence of i.i.d. random variables defined on the same probability space $(\Omega, \mathcal{F}, P)$ with $E[v_i(\ell)] = 1$ and $Var[v_i(\ell)] = c^2_i < \infty$. We then define $u_k(\ell)/\alpha_k$ to be the time between the ℓth and $(\ell+1)$st demand for a product from stream k. Therefore, $u_k(1)/\alpha_k$ is the time until the first demand for product k, $u_k(2)/\alpha_k$ is the inter-arrival time between the first and second product k demands, and so on. Likewise, we define $m_i v_i(\ell)$ to be the service time of the ℓth type i component at the workstation. Throughout, we make the following stability assumption

$$m_i < \lambda_i^{-1}, \tag{2.3}$$

that is, that the expected processing time of component i at the workstation is less than the expected inter-arrival times for component i. We define the traffic intensity of the jth machine

$$\rho_j = \begin{cases} \sum_{i=1}^{c} \lambda_i m_i & \text{if } j = 0 \\ 1 & \text{otherwise} \end{cases} \tag{2.4}$$

We denote the initial base-stock level of FG inventory i by s_i (mnemonic for *stock*).

The initial base-stock level may be zero, corresponding to a pure make-to-order case, or a finite positive integer. We assume that if an allowable backlog level is infinite if, and only if, all components can be infinitely backlogged.

Last, we are able to classify our demand structure. Define $D_k(t)$ to be the total number of arrivals of product orders from stream k, $(1 \leq k \leq K)$, in t units of busy time at the k-th virtual station. In order for our results to hold, we require only that the demand process $D = (D_1, D_2, \ldots, D_K)$ satisfy the functional central limit theorem discussed in Chapter 1. That is, we assume the existence of a positive constant vector $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_K)$ and a $K \times K$ covariance matrix Ω such that

$$r^{-1/2}\{D(r\cdot) - r\alpha\cdot\} \Longrightarrow D^*(\cdot),$$

as $r \to \infty$, where D^* is a $K-$dimensional Brownian motion with zero drift and covariance matrix Ω, and "$\Longrightarrow$" denotes convergence in distribution where the state space is $\mathbf{D}[0, \infty]$, the space of right-continuous functions with left limits endowed with the Skorohod J_1 topology (see Billingsley, [4]). In Section 2.2, we required the sequence $\{u_k(\ell)\}$ to be i.i.d. with unit mean and finite variance, ensuring the the FCLT assumption does hold for our model. The parameter α_k is the long-run demand rate of product k, and the matrix Ω is the asymptotic covariance matrix of the demand process. In particular, we allow Ω to have non-zero elements off the diagonal so that our model can accommodate correlated demand processes.

2.3 Two Limit Results

This section states two limit results for the following processes:

- $W(t)$: hours of work remaining at production resource at time t;
- $I_0(t)$: cumulative hours, up to time t, of intervals during which the production resource is idle;
- $Z_i(t)$: number of units in component i inventory at time t;
- $I_k(t)$: cumulative hours, up to time t, of intervals during which there is insufficient inventory to satisfy type k orders.

The workload level $W(t)$ is the sum of the production times of all those components at the production resource at time t. Note that while W and I are nonnegative processes, $Z_i(t)$ may be negative if component k is in backorder at time t. In particular, in the lost sales case, Z_i takes values between 0 and s_i, while in the backlog case, Z_i may range from negative infinity to the upper limit of s_k. For each product k, the process I_k tallies the cumulative amount of time during which there is not enough inventory to satisfy orders; that is, intervals of time during which $Z_i(t) < q_{ik}$ for some component i used in product k. Henceforth, we will write Z and I to mean the vector processes

$(Z_1, \ldots, Z_c)$ and $(I_0, I_1, \ldots, I_K)$.

The theorems characterize the behavior of our system in a certain "limiting regime", and these theorems will serve as the basis of our analysis. Loosely speaking, the limiting regime, or "asymptotic regime", is one in which we require that the utilization at the workstation be nearly 100%. That is, we require that

$$\rho \approx 1. \tag{2.5}$$

Furthermore, we require that the base-stock levels be high relative to utilization according to

$$s_i \sim O\left(\frac{1}{1-\rho}\right). \tag{2.6}$$

We expect our estimates to perform well when the system under consideration satisfies the heavy-traffic conditions given in (2.5) and (2.6). For example, we expect good estimates for a system with a nominal utilization of 99% and base-stock levels on the order of 100, or a system with nominal utilization of 95% and base-stock levels on the order of 20. In simulation studies, we found that the estimates are robust even in regimes with lower nominal utilization levels and lower base-stock levels. We now make precise what we mean by the "regime of analysis".

Our limit results are essentially "central limit theorems" that characterize the behavior of the system when the primitive processes are scaled by a parameter and the scaling parameter is taken to infinity. To be specific, we define the following scaling notation, as applied to a generic process $X(t)$:

$$X^n(t) \equiv \frac{1}{n} X(n^2 t). \tag{2.7}$$

Observe that X^n is scaled in a manner consistent with central limit theorems.

The parameter n can be better explained by considering separately the two scenarios of lost sales and backorder. We begin with the backorder system.

In this scenario, we take the scaling parameter to be

$$n \equiv \left\lfloor \frac{1}{1-\rho} \right\rfloor, \tag{2.8}$$

where $\lfloor x \rfloor$ denotes the inter part of x. With this convention, passing n to infinity is equivalent to letting nominal utilization approach 100%. Let us call the system associated with a given value of n the n^{th} system, and use the superscript "(n)" to mark the parameters and processes associated with the n^{th} system. For example, the demand rates and mean production times of type i components in the n^{th} system are λ_i^n and m_i^n, respectively. Product k demand rates in the n^{th} system are similarly denoted α_k^n.

To avoid simplistic, or degenerate, limiting behavior, we make two additional compensating assumptions. First, we want all base-stock levels to be of the same order. Second, we assume that as utilization of the resource increases to 100%, the component base-stock levels, in turn, would increase (implying higher service levels). Specifically, we require

$$\frac{s_i^n}{n} \to \beta_i \quad \text{as} \quad n \to \infty, i = 1, \ldots, c \tag{2.9}$$

where s_i^n is the base-stock level of type i components in the n^{th} system and β_i is a finite and positive number.

Let

$$C(k) = \{i : q_{ik} > 0\}.$$

Then $C(k)$ represents the set of components used in product k. Set

$$\sigma^2 = \sum_{i=1}^{c} \lambda_i m_i^2 c_{0i}^2 + \sum_{j=1}^{K} \sum_{k=1}^{K} \left(\sum_{i \in C(j)} m_i q_{ij} \right) \left(\sum_{\ell \in C(k)} m_\ell q_{\ell k} \right) \Omega_{jk}. \tag{2.10}$$

Finally, define θ as

$$\theta n(\rho - 1). \tag{2.11}$$

Under the scaling convention given in (2.7) and assumptions (2.8)-(2.9), the following theorem holds.

Theorem 1 *As $n \to \infty$, $(\tilde{W}^n, \tilde{N}^n, \tilde{I}_0^n, \tilde{Z}^n) \Longrightarrow (W^*, N^*, I_0^*, Z^*)$ where*

$$W^*(t) = \xi_0^*(t) + I_0^*(t) \tag{2.12}$$
$$\xi^*(t) \textit{ is a Brownian motion with drift } \theta \textit{ and variance } \sigma^2 \tag{2.13}$$
$$I_0^*(t) \textit{ is continuous and non-decreasing with } I_k^*(0) = 0 \tag{2.14}$$
$$I_0^*(t) \textit{ increases only at times } t \textit{ when } W^*(t) = 0 \tag{2.15}$$
$$N_i^*(t) = \lambda_i W^*(t) \tag{2.16}$$
$$Z_i^*(t) = \beta_i - \lambda_i W^*(t) \tag{2.17}$$

This will be proved in an appendix. For our purposes here, it suffices to note that statements (2.12)-(2.15) identify the limiting workload process W^* as a reflected Brownian motion with drift θ and variance σ^2 whose state space is the nonnegative half-line $[0, \infty)$, or $RBM(1, \sigma^2, [0, \infty))$.

We now turn our attention to the lost sales case. We define the index of the n^{th} system to be

$$n = s_1 + s_2 + \ldots s_c. \tag{2.18}$$

Then n can be thought of as the total capacity of the system to accept product orders. When we take $n \to \infty$ in the theorem, we are, in effect, considering successively larger systems.

Again, to avoid simplistic, or degenerate, limiting behavior, we need to make two compensating assumptions. The first, as in the backlog scenario, is the assumption that all base-stock levels are of the same order:

$$\frac{s_i^n}{n} \to \beta_i \quad \text{as} n \to \infty, \ i = 1, \ldots, c \tag{2.19}$$

where s_i^n is the base-stock level of type i components in the n^{th} system and β_i is a finite and positive number. For the second, we assume the existence of parameters λ_i, m_i, and α_k such that $\lambda_i^n \to \lambda_i$ and $m_i^n \to m_i$ for $i = 1, \ldots, J$ and $\alpha_k^n \to \alpha_k$ for $k = 1, \ldots, K$. It follows that $\rho^n \to \rho$, as $n \to \infty$. We further assume that as base-stock levels increase (which implies higher fill rates), the utilization of the workstation also increases (in turn implying a degradation of service). Specifically, we make the following assumption:

$$n(\rho^n - 1) \to \theta \quad \text{as } n \to \infty \tag{2.20}$$

where θ is finite and negative.

Let σ^2 be defined as in (2.10). Then under the heavy-traffic conditions (2.19) and (2.20), we have the following theorem.

Theorem 2 *As $n \to \infty$, $(\tilde{W}^n, \tilde{Z}^n, \tilde{I}^n, \tilde{N}^n) \Longrightarrow (W^*, Z^*, I^*, N^*)$*

$$W^*(t) = \xi^*(t) + I_0^*(t) - Y^*(t), \tag{2.21}$$

$$Y^*(t) = \sum_{k=1}^{K} \left(\sum_{i \in C(k)} q_{ik} m_i \right) \alpha_k I_k^*(t), \tag{2.22}$$

$$\xi^*(t) \sim BM(\theta, \sigma^2), \tag{2.23}$$

$$I_k^*(t) \textit{ is continuous and non-decreasing with } I_k^*(0) = 0, \tag{2.24}$$

$$I_0^* \textit{ increases on at times } t \textit{ when } W^*(t) = 0, \tag{2.25}$$

$$I_k^*(t) \textit{ increases only at times } t \textit{ when } Z_i^*(t) = 0\textit{, for some } i \in C(k), \tag{2.26}$$

$$Z_i^*(t) = \beta_i - \lambda_i W^*(t), \tag{2.27}$$

$$N_i^*(t) = \lambda_i W^*(t). \tag{2.28}$$

The proof in the lost sales case is an extension of Nguyen [47], and will be provided in an appendix of the dissertation.

The theorem states that under the heavy-traffic regime of large inventories and a heavily utilized workstation, the scaled workload process at the workstation can be written as the sum of a Brownian motion and two non-decreasing processes. In the full backorder case, these collapse into only one non-decreasing process. The distribution of such a process is well known, and will be exploited to calculate performance measures. Specifically, in the lost sales case W^* has a truncated exponential distribution on a closed interval, while in the full backorder case it has an exponential distribution with mean $\sigma^2/2|\theta|$. Both the mean and the variance of the Brownian motion are calculated purely in terms of the primitive data of the system. In fact, it is only in the variance term that the correlation between different components makes an appearance. It is also evident from the variance term that mead processing times have a greater effect on the variability in the workload than arrival rates. Furthermore, a functional analog of Little's law holds for the number of component orders at the workstation in Equation (2.28), which Equation (2.27) relates to the level of finished goods inventories.

2.4 The Approximation

The heavy-traffic limit theorem gives us a theoretical justification for approximating the workstation workload process of our system with a reflected Brownian motion. In the lost sales case, it is a truncated exponential distribution, and in the backorder case it is an exponential distribution. In this section, we characterize the workstation workload process for both the lost sales and backorder case, and use this to develop approximations for the system's performance measures.

2.4.1 The Workstation Workload Process Under Lost Sales

We turn our attention first to the properties of the workstation workload process W^*. In particular, we wish to know its distribution, and range of values. As stated in the previous section, it is a well know result that the workload process is a reflected Brownian motion on a compact interval in the partial- backorder case, and a reflected Brownian motion on a half-open infinite interval in the full backorder case. We address the partial-backorder case first.

From Harrison [25], a reflected Brownian motion Z with drift $\mu \neq 0$ and variance σ^2 defined on $[0,b]$ has a long-run occupancy distribution

$$P(Z \in dz) = \frac{\theta e^{\theta z}}{e^{\theta b} - 1} dz, \tag{2.29}$$

where $\theta = 2\mu/\sigma^2$. When $\mu = 0$, the reflected Brownian motion is uniformly distributed on $[0,b]$. We need to determine μ, σ^2, and b in Equation (2.29). Consider Equation (2.21). By definition, $I_k^*(t)$ increases for all t where *at least one* of the input component FG buffers that product k draws from is empty. Thus, $Y^*(t)$ increases only at times t when $Z_i^*(t) = 0$ for some $i = 1, \ldots, N$. From Equation (2.27) this occurs only at times t when

$$\lambda_i W^*(t) = \beta_i. \tag{2.30}$$

for some $i = 1, \ldots, N$. Thus $Y^*(t)$ increases only at times t where

$$W^*(t) = \lambda_i^{-1}\beta_i. \tag{2.31}$$

It follows that $W^*(t) \sim RBM(\theta, \sigma^2)$ on the interval $[0,b^*]$, where

$$b^* = min\{\lambda_1^{-1}\beta_1, \lambda_2^{-1}\beta_2, \ldots, \lambda_c^{-1}\beta_c\}, \tag{2.32}$$

and σ^2 is given by Equation (2.10).

As $\tilde{W}^n \approx W^*$, $\tilde{W}^n$ behaves very much like an RBM on the interval $[0,b^*]$ with mean $n(\rho - 1)$ and variance σ^2 (given by (2.10)), for n large and $\rho \approx 1$. Thus, in order to estimate the performance measures of the original workload process $W(t)$, we must "unscale" the process $\tilde{W}^n$ in the following manner

$$W(t) = n\tilde{W}^n(t/n^2).$$

The Brownian component of $\tilde{W}^n$ is a (μ, σ^2) Brownian motion with mean $n(\rho - 1)$, and variance σ^2 is given by Equation (2.10). From the basic properties of Brownian motion, $X(t) \sim BM(\mu, \sigma^2)$, then $aX(t) \sim BM(a\mu, a^2\sigma^2)$ and $X(bt) \sim BM(b\mu, b\sigma^2)$, we see that the Brownian component of $n\tilde{W}^n(t/n^2)$ is a $BM(\mu/n, \sigma^2)$. Therefore,

$$W(t) \sim RBM(\rho - 1, \sigma^2), \tag{2.33}$$

on the interval $[0,\tilde{b}]$, where σ^2 is given by Equation (2.10) and

$$\tilde{b} = min\{\lambda_1^{-1}s_1, \lambda_2^{-1}s_2, \ldots, \lambda_c^{-1}s_c\}. \tag{2.34}$$

2.4.2 Components Most Likely to Stock Out

From (2.27) and the discussion following Theorem 2, it can be seen that the limiting inventory process Z_i^* is distributed as an RBM on the interval $[\beta_i - \lambda_i b^*, \beta_i]$ where $b^* = \min_{i=1,\ldots,c} \beta_i/\lambda_i$. Upon closer inspection, it becomes evident that for some components i, the limiting inventory process Z_i^* never reaches zero. Specifically, let

$$\mathcal{B}^* \equiv \left\{ i : \frac{\beta_i}{\lambda_i} = b^* \right\}.$$

For any component $i \notin \mathcal{B}^*$, the lower bound of the interval $\beta_i - \lambda_i b^*$ is greater than zero, so the inventory process Z_i^* is strictly positive!

Recall that $\beta_i \equiv \lim_{n\to\infty} s_i/n$. With the scaling reversed, Theorem 2 implies that the inventory process Z_i is an RBM on the interval $[s_i - \lambda_i\tilde{b}, s_i]$ where

$$\tilde{b} \equiv \min_{i=1,\ldots,c}\left\{\frac{s_i}{\lambda_i}\right\}.$$

Moreover, with

$$\mathcal{B} \equiv \left\{k : \frac{s_i}{\lambda_i} = \tilde{b}\right\},$$

Z_i is strictly positive for all components $i \notin \mathcal{B}$.

The components in $\mathcal{B}$ are those with the highest "run-out rate." That is, s_i/λ_i is the "run-out time," or the *expected* duration of time that a full inventory of type i components can satisfy demands without replenishment. So, *on average,* the components in $\mathcal{B}$ are the first components to deplete a full inventory. Theorem 2 suggests that in the limiting regime of utilization approaching 100%, the components with the highest run-out rate are the *only* ones to ever stock out.

Clearly this characterization is exact only in the heavy traffic limit and only for the *scaled* limiting processes. In the limit, whenever one of the components stocks out, the total rate at which work is generated for the production resource falls below the critical level; the production resource experiences temporary "excess" capacity; it therefore is able to keep all other inventories adequately replenished. (See Nguyen 1997 for a more detailed and rigorous discussion of this phenomenon.) The interpretation in non-limit systems and unscaled processes is more subtle. We interpret the result as saying that the fill rates of those components with the highest run-out rate are relatively low, whereas the fill rates for all other components are high and "essentially 100%." That is, Theorem 2 suggests that even under the presence of stochastic variability, the run-out rate, characterized simply by the average rate of demand and the base-stock level, determines those components most likely to stock out. Nguyen (1997) refers to the component with the highest run-out rate as the *bottleneck* component.

The role played by the bottleneck component is fundamental to the dynamics of the system. Basically, the bottleneck component acts as a control. It determines the range of the workstation workload process. Furthermore, product orders that draw on the bottleneck component will have lower acceptance rates than those that do not. Clearly, any product that requires the bottleneck component can have an acceptance rate no greater than the acceptance rate of the bottleneck component. Also, the bottleneck component gives an intuitive explanation for why the workstation workload process in the lost sales case is defined on a finite interval, when it theoretically could be arbitrarily large (such as in the case with exponential processing times). As more component orders are sent to the workstation, product orders begin to be rejected. This, in turn, gives the workstation time to catch up. Identification of the bottleneck component can also lead to policies for setting base-stock and backlog levels. For example, suppose a producer offers three products, and determines that the first should be filled 90%, while the other two should be filled 95% of the time. Suppose further that they do not all require the same components. He can then set initial base-stock levels so that the first product requires the bottleneck component, while the other two do not (providing, of course, that the remaining two products do not require the component the producer has designated to be the bottleneck). Since non-bottleneck components have an acceptance rate of essentially one, the producer can guarantee that he meets his targets.

2.4.3 Product Acceptance Rates Under Lost Sales

We next wish to calculate product acceptance rates. That is, we wish to calculate the probability that a particular customer arriving and requesting a certain kit of components will not be turned away empty handed. In the full backorder case the issue of product acceptance and rejection is irrelevant. This is because the assumption that all buffer levels can be infinitely backlogged means that no product order will ever be rejected. Therefore, the workstation order queue sees ever arrival of demand. Essentially, the queue behaves as an open $\Sigma GI/G/1$ queue, where the correlation across component orders is absorbed into the variance term of

the Brownian motion. We will therefore restrict ourselves to the case of lost sales.

Our calculation of product acceptance rate will depend entirely on the acceptance rates of the components in the product mix. In the following analysis, we will assume for the moment that the component acceptance rates are known.

Observe that a particular product order is allowed to arrive if and only if there is an component in *each* of the finished goods buffers that the product order draws from. Thus, a product order is prevented from arriving exactly when *at least* one of the finished goods buffers on which the order draws is empty. Therefore, the product acceptance rate is defined to be one minus the probability that at least one of the finished goods buffers is empty.

More formally, define the set $\mathcal{I}_k$ to be the set of all components that product order k draws from so that

$$\mathcal{I}_k = \{i | q_{ik} > 0, i = 1, \ldots, J\}. \tag{2.35}$$

Denote the acceptance rate of product order k by $\tilde{\pi_k}$. Then $\tilde{\pi_k}$ is defined to be

$$\tilde{\pi_k} = 1 - P(\bigcup_{i \in \mathcal{I}_k} \{Z_i(t) = 0\}), \tag{2.36}$$

where $\mathcal{I}_k$ is given by (2.35). Equation (2.36) states that the probability that an order for product k at time i is one minus the probability that at least one of the FG inventories that product k draws is empty.

We suppress the dependence on t with the understanding that we are considering a fixed point in time. From the *general inclusion-exclusion formula* (formula (2.9) on page 24 of [4]), and noting that $P(\{Z_i = 0\}$ is precisely the acceptance rate for the

i-th component, we have

$$\begin{aligned} P(\bigcup_i \{Z_i = 0\}) &= \sum_i (1-\tilde{\gamma}_i) - \sum_{i<j} P(\{Z_i = 0\} \bigcap \{Z_j = 0\}) \\ &\quad + \sum_{i<j<l} P(\{Z_i = 0\} \bigcap \{Z_j = 0\} \bigcap \{Q_l = 0\}) \\ &\quad + \ldots + (-1)^{n+1} P(\bigcap_i \{Z_i = 0\}) \end{aligned}$$

for all $i, j, l \in \mathcal{I}_k$.

Calculation of the higher terms is difficult in general, so we propose two different approximations for these terms.

Approximation 1: We make the assumption that the probability that two components will be stocked out simultaneously is zero. That is $P(\{Z_i = 0\} \bigcap \{Z_j = 0\}) = 0$ for all i, j where $i \neq j$. The reasoning behind this assumption is the observation that since the non-bottleneck components almost never stock out, these terms will be very small. Under this assumption, the higher order terms disappear, and the acceptance rate for product k is given as

$$\tilde{\pi}_k = 1 - \sum_{i \in \mathcal{I}_k} (1 - \tilde{\gamma}_i). \tag{2.37}$$

Approximation 2: We make the assumption that component buffer levels are independent of one another. That is, we assume that

$$P(\{Z_i = 0\} \bigcap \{Z_j = 0\}) = P(\{Z_i = 0\}) P(\{Z_j = 0\})$$

for all i, j where $i \neq j$. The reasoning behind this assumption is that the second order terms and above will be small since the component acceptance rates of non-bottleneck components will be very close to zero. Under this assumption, the procedure for calculating the acceptance rate of product k is

$$\begin{aligned} \tilde{\pi}_k &= 1 - P(\bigcup_{i \in \mathcal{I}_k} \{Z_i = 0\}) \\ &= P([\bigcup_{i \in \mathcal{I}_k} \{Z_i = 0\}]^c) \\ &= P(\bigcap_{i \in \mathcal{I}_k} \{Z_i = 0\}^c) \\ &= P(\bigcap_{i \in \mathcal{I}_k} \{Z_i > 0\}) \\ &= \prod_{i \in \mathcal{I}_k} P(\{Z_i > 0\}), \end{aligned} \tag{2.38}$$

where the last equality follows from our assumption of independence. Observing that the terms of the product in the last line are simply the acceptance rates of the respective components gives our second approximation of product acceptance rate as

$$\tilde{\pi}_k = \prod_{i \in \mathcal{I}_k} \tilde{\gamma}_i. \tag{2.39}$$

2.4.4 Component Acceptance Rates

We now describe the procedure for approximating the component acceptance rates used in the previous section. Define $\mathcal{P}_1$ to be the set of all products that draw from component 1. Formally, $\mathcal{P}_1 = \{k | q_{1k} > 0\}$. Under the realization that component 1 is the only component to ever stock out, we observe that the only idleness processes that will ever increase are the ones for products that draw on component 1. This is because the component acceptance rates for non-bottleneck components are essentially 100%. Therefore, any product order arriving that draws on these components, and not on the bottleneck component, should always be filled. Since the idleness processes capture the total amount of time that a given product order is unable to arrive due to an empty component finished goods inventory, they are strictly equal to zero in this case. Therefore, we can write (2.21) as

$$W^*(t) = \xi^*(t) + I_0^*(t) - \sum_{k \in \mathcal{P}_1} (\sum_{i=1}^{c} q_{ik} m_i) \alpha_k I_k^*(t). \tag{2.40}$$

The upper regulator $Y^*(t)$ from (2.22) records the cumulative total amount of time in $[0,t]$ that the workload process $W^*(t)$ is at the upper barrier b, given by (2.55). From Proposition 5 of §5.5 of Harrison ([25]), we know that for a process $Z(t) = \xi(t) + I(t) - Y(t)$ defined on the closed interval $[0,b]$, where $\xi(t)$ is a (μ, σ^2) Brownian motion, $I(t)$ and $Y(t)$ are continuous, non-decreasing and increase only for those t where $Z(t) = 0$, $Z(t) = b$, respectively, then

$$\frac{1}{t}\mathbf{E}[Y(t)] \sim \begin{cases} \frac{\sigma^2}{2b} & \text{if } \mu = 0 \\ \\ \frac{\mu}{1-e^{-\theta b}} & \text{otherwise} \end{cases} \tag{2.41}$$

where $\theta = 2\mu/\sigma^2$.

Therefore, in our case the approximate proportion of time that the workload process, given by (2.40), is at the upper barrier b, given by (2.55), is

$$\frac{1}{t}\mathbf{E}[(\sum_{k \in \mathcal{P}_1} (\sum_{i=1}^{c} q_{ik} m_i) \alpha_k I_k^*(t))] \sim \begin{cases} \frac{\sigma^2}{2b} & \text{if } \mu = 0 \\ \\ \frac{\mu}{1-e^{-\theta b}} & \text{otherwise} \end{cases}$$

Observe that $\mathbf{E}[(\sum_{k \in \mathcal{P}_1} (\sum_{i=1}^{c} q_{ik} m_i) \alpha_k I_k^*(t))] = \sum_{k \in \mathcal{P}_1} (\sum_{i=1}^{c} q_{ik} m_i) \alpha_k \mathbf{E}[I_k^*(t)]$, which follows from the linearity of expectation. The idleness processes I_k^* increase exactly for those t where the workstation workload process is at the upper barrier. But this occurs exactly when the bottleneck component is stocked out, as the acceptance rates of the non-bottleneck components are always unity in the limit. Therefore, each of the idleness processes will increase exactly when the bottleneck component is stocked out. In essence, they behave as one idleness process,which we denote I^*. The probability of rejection of a bottleneck component order is then precisely the proportion of time that this idleness process increases. If we define

$$C_1 = \sum_{k \in \mathcal{P}_1} (\alpha_k \sum_{i=1}^{c} q_{ik} m_i), \tag{2.42}$$

then by the reasoning above, $\sum_{k \in \mathcal{P}_1} (\sum_{i=1}^{c} q_{ik} m_i) \alpha_k \mathbf{E}[I^*(t)]$, and

$$\frac{1}{t}\mathbf{E}[I^*(t)] \sim \begin{cases} \frac{1}{C_1}\frac{\sigma^2}{2b} & \text{if } \mu = 0 \\ \\ \frac{1}{C_1}\frac{\mu}{1-e^{-\theta b}} & \text{otherwise} \end{cases} \tag{2.43}$$

The quantity given in (2.43) is the proportion of time that the combined idleness process is increasing, which is exactly the proportion of time that the bottleneck component is stocked out. We thus take our bottleneck component acceptance rate to be

$$\tilde{\gamma}_1 = \begin{cases} 1 - \frac{1}{C_1}\frac{\sigma^2}{2b} & \text{if } \mu = 0 \\ \\ 1 - \frac{1}{C_1}\frac{\mu}{1-e^{-\theta b}} & \text{otherwise} \end{cases} \tag{2.44}$$

The values of b, μ, and θ are as derived in the previous section. For the case where $\mu \neq 0$, we use $\mu = \rho - 1$ and (2.55) to get

$$\tilde{\gamma}_1 = 1 - \frac{1}{C_1}\frac{\rho - 1}{(1 - e^{-\theta\rho s_1/\tilde{\gamma}_1\lambda_1})} \tag{2.45}$$

where $\theta = 2(\rho - 1)/\sigma^2$. This cannot be further simplified, and $\tilde{\gamma}_1$ must be solved iteratively.

In the driftless case it is possible to simply the bottleneck component acceptance rate estimate considerably. The condition $\mu = \rho - 1$ is equivalent to the condition $\rho = 1$. From (2.43) and (2.55), the first case of (2.44) yields

$$\tilde{\gamma}_1 = 1 - \frac{1}{C_1}\frac{\sigma^2}{2}\frac{\tilde{\gamma}_1\lambda_1}{s_1}$$

Solving for $\tilde{\gamma_1}$ and simplifying gives

$$\tilde{\gamma_1} = 1 - \frac{\lambda_1\sigma^2/2C_1}{s_1 + \lambda_1\sigma^2/2C_1} = \frac{s_1}{s_1 + \lambda_1\sigma^2/2C_1}. \tag{2.46}$$

Summarizing, we use the following estimate for component acceptance rate of the approximation:

$$\tilde{\gamma_1} = \begin{cases} \frac{s_1}{s_1+\lambda_1\sigma^2/2C_1} & \text{if } \rho = 1 \\ \\ 1 - \frac{1}{C_1}\frac{\rho-1}{1-e^{-\theta\rho s_1/\tilde{\gamma_1}\lambda_1}} & \text{otherwise} \end{cases} \tag{2.47}$$

where $\theta = 2(\rho - 1)/\sigma^2$. The first can simply be computed, while the second must be solved iteratively.

To compute the non-bottleneck component acceptance rates, we use the following method due to Nguyen [47]. Recall that the components are well-ordered in terms of descending blockage. To estimate the component acceptance rate of the second component, we turn the arrival rate of the first component (the bottleneck component) down enough so that the second component is now the bottleneck. Some adjustments need to be made. Specifically, as the arrival rate of component 1 has been "turned down", the nominal utilization rate of the workstation will also be turned down. The value for ρ above must be recalculated. Likewise, as the value of θ depends on the value of ρ, that value, too, must be recalculated. When these have been recalculated to reflect the new, and reduced, arrival rate of requests for component 1, we then use the estimate provided above. More formally, define B_j to be the set of all product orders that require component j. Then

$$\mathcal{P}_j = \{k|q_{jk} > 0\}. \tag{2.48}$$

Let

$$C_j = \sum_{k\in\mathcal{P}_j}(\alpha_k \sum_{i=1}^{c} q_{ik}m_i) \tag{2.49}$$

As the arrival rate of component 1 has been "turned down", the nominal utilization rate of the workstation will also be turned down. Define this reduced nominal utilization rate to be $\rho^{(j)}$, where

$$\rho^{(j)} \equiv \sum_{i=1}^{j-1}\tilde{\gamma}_i\lambda_i m_{0i} + \sum_{i=j}^{N}\lambda_i m_{0i}. \tag{2.50}$$

The new value for θ is then

$$\theta^{(j)} \equiv \frac{2(\rho^{(j)} - 1)}{\sigma^2}. \tag{2.51}$$

The component acceptance rate is then calculated as

$$\tilde{\gamma}_j = 1 - \frac{1}{C_j} \frac{\rho^{(j)} - 1}{\left(1 - e^{-\theta^{(j)}\rho^{(j)} s_j / \tilde{\gamma}_j \lambda_j}\right)}. \tag{2.52}$$

2.4.5 Finished Goods Inventory Levels

We wish to use our approximation of the workload process and Equation (2.16) to approximate the number of orders for components at the workstation. Examination of the one-component, one-product case with exponential inter-demand and service times and infinite backlogging provides illumination on how to proceed. In this case, the workstation buffer process (the number of orders in the workstation queue, including order in service)is simply an M/M/1 queue, and the workstation workload process should be the sojourn time in the system. The workstation workload process has an exponential distribution with mean $\sigma^2/2(1-\rho)$. The variance term reduces to $2m$, and the mean of the workload process becomes $m\rho/(1-\rho)$. But this is the expected *delay in queue* for an M/M/1 system. The conclusion is that in the limit, W does not include the service time at the workstation. It is the delay in workstation queue, *not* the sojourn time at the queue. The result is that a strict application of Equation (2.16) would lead to an incorrect approximation of the number of orders at the workstation. Since the correct expression for mean sojourn time should be $m/(1-\rho)$, which can be obtained from $m\rho(1-\rho)$ by division by ρ. In light of this, we approximate WIP at the workstation by

$$N_i \approx \frac{\lambda_i}{\rho} W, \quad i = 1, \ldots, c \tag{2.53}$$

in the backorder case.

While we do not yet have a means for estimating component acceptance rates, let us for the moment assume that we do. The method for calculating these component acceptance rates will be addressed in a later section. Let $\tilde{\gamma}_i$ denote the acceptance rate for component i, $i = 1, \ldots, N$. The $\tilde{\gamma}_i$ is the probability that an component order of type i will not be rejected. Then, we approximate WIP at the workstation by

$$N_i \approx \frac{\tilde{\gamma}_i \lambda_i}{\rho} W, \quad i = 1, \ldots, c \tag{2.54}$$

in the lost sales case. Thus, the division by the traffic intensity is an acknowledgment of the fact that the workstation workload process does not include service time at the workstation.

For clarification, we let the components be ordered such that

$$\lambda_1^{-1}s_1 < \lambda_2^{-1}s_2 < \ldots < \lambda_c^{-1}s_c.$$

Thus, the indices of the components are increasing with descending blockage. That is, component 1 is always the bottleneck, component 2 the next bottleneck, and so on. Observe that Z_1, the component 1 process, lives on the interval $[0,s_1]$. It then follows from this, and Equation (2.54) that the workstation workload process is defined on the interval $[0,\tilde{b}]$, where $\tilde{b}$ is given by

$$\tilde{b} \equiv \frac{\rho s_1}{\tilde{\gamma}_1 \lambda_1}. \tag{2.55}$$

Finally, from Equations (2.10), (2.29), (2.33), and (2.55), it follows that we can approximate the long-run occupancy distribution of the workload at the workstation by

$$P(W(t) \in dz) = \frac{\theta e^{\theta z}}{e^{\theta \rho s_1/\tilde{\gamma}_1 \lambda_1} - 1} dz, \tag{2.56}$$

where

$$\theta = \frac{2(\rho - 1)}{\sigma^2}. \tag{2.57}$$

Integrating (2.56) over the set $[x, \rho s_1/\tilde{\gamma}_1 \lambda_1]$ yields

$$P(W(t) > x) = \frac{e^{\theta \rho s_1/\tilde{\gamma}_1 \lambda_1} - e^{\theta x}}{e^{\theta \rho s_1/\tilde{\gamma}_1 \lambda_1} - 1}. \tag{2.58}$$

Equation (2.58) is the complimentary distribution function for the amount of work at the workstation at time t in the lost sales case.

From §5.5 in Harrison [25], we find that the mean of the steady-state distribution for a reflected Brownian motion defined on the closed interval $[0,b]$ is given by $b/(1 - e^{-\theta b}) - 1/\theta$, where $\theta = 2\mu/\sigma^2$ for the case where $\mu \neq 0$. For the case where $\mu = 0$, the steady-state distribution of the RBM is uniform on the interval $[0,b]$. For the case where $\mu = 0$ it is a simple calculation to see that

$$\mathbf{E}[W(\infty)] \approx \frac{1}{2\lambda_1}(s_1 + \frac{\lambda_1 \sigma^2}{2C_1}). \tag{2.59}$$

Similarly, for the case where $\mu \neq 0$ it is a straightforward calculation to see that

$$\mathbf{E}[W(\infty)] \approx \frac{\rho s_1}{\tilde{\gamma}_1 \lambda_1 (1 - e^{-\rho \theta s_1/\tilde{\gamma}_1 \lambda_1})} - \frac{1}{\theta}. \tag{2.60}$$

Next, using (2.54), and $Z_i = s_i - N_i$, we can approximate mean FG inventory for component i by

$$
\begin{aligned}
\mathbf{E}[Z_i(\infty)] &= s_i - \frac{\tilde{\gamma}_i\lambda_i}{\rho}\mathbf{E}[W(\infty)] \\
&= \frac{\tilde{\gamma}_i\lambda_i}{\rho}\left[\frac{\rho s_1}{\tilde{\gamma}_1\lambda_1(1-e^{-\rho\theta s_1/\tilde{\gamma}_1\lambda_1})} - \frac{1}{\theta}\right]
\end{aligned}
\tag{2.61}
$$

2.4.6 Order Response Times Under Backlogging

When orders can be backlogged, order response time — the time interval from when an order is received and when it is filled — becomes an important component of customer service. We define *service level* to be the fraction of orders that are filled within a promised lead time. We are interested in the same question that Glasserman and Wang [21] posed in their paper: If we are to maintain a constant level of service, how much reduction in the promised lead time can be achieved with a unit increase in inventory?

Let $R_k(t)$ denote the response time of the next order for product k that arrives after time t. The following limit theorem holds under the scaling conventions and assumptions of Section2.3.

Theorem 3 *As $n \to \infty$, $R_k^n \Longrightarrow R_k^*$ where*

$$
R_k^*(t) = \max_{i\in\mathcal{C}(k)}\left(W^*(t) - \frac{\beta_i}{\lambda_i}\right)^+ = \left(W^*(t) - \min_{i\in\mathcal{C}(k)}\left\{\frac{\beta_i}{\lambda_i}\right\}\right)^+. \tag{2.62}
$$

To better understand the implications of (2.62), let us consider the relationship between response time and workload for the simplest case of a product that requires one unit of a single component type. In addition, let us, for now, assume that assembly requires zero time.

In this discussion we will drop all subscripts because they are not necessary to distinguish between component types. Observe that when the base-stock level is s each order is filled by the component that was put into production by its s^{th} previous order. Denote as the "marked" order the next order to arrive after time t, $\tau(t)$ the arrival time of the marked order, and $\phi(t)$ the arrival time of the s^{th} previous order. That is, if $D(\cdot)$ is the demand process then $D(\tau(t)) - D(\phi(t)) = s$. The component put into production by the s^{th} previous order will be completed at time $\phi(t)+W(\phi(t))$. Consequently the amount of time the "marked" order must wait for its component is precisely

$$
\begin{aligned}
R(t) &= (\phi(t) + W(\phi(t)) - \tau(t))^+ \\
&= (W(\phi(t)) - (\tau(t) - \phi(t)))^+ .
\end{aligned}
\tag{2.63}
$$

In our scaling convention $R^n(t) = n^{-1}R(n^2t)$, $W^n(t) = n^{-1}W(n^2t)$, and $\beta = \lim_{n\to\infty} s/n$, which implies $s \approx n\beta$ for large n. Moreover, Theorem 3 implies $R^n \approx R^*$ and $W^n \approx W^*$ for large n. By "undoing" the scaling in (2.62), we derive the following approximate relationship:

$$\begin{aligned}
(W(\phi(t)) - (\tau(t) - \phi(t)))^+ = R(t) = nR^n(n^{-2}t) &\approx nR^*(n^{-2}t) \\
&\approx \left(nW^*(n^{-2}t) - \left\{\frac{n\beta}{\lambda}\right\}\right)^+ \\
&\approx \left(nW^n(n^{-2}t) - \left\{\frac{s}{\lambda}\right\}\right)^+ \\
&= \left(W(t) - \left\{\frac{s}{\lambda}\right\}\right)^+.
\end{aligned}$$

The limit result of Theorem 3 provides theoretical justification for a rather simple characterization of the response time process. It suggests that in the heavy-traffic regime of high utilization and high base-stock levels, the following simplifications may be made: First, the current workload level is a good approximation of the workload level at the time of the s^{th} previous order. (Readers familiar with the literature of heavy-traffic analysis will recognize this result as the snapshot principle.) Second, the random length of time between the current arrival and that of the s^{th} previous order can be approximated simply by its mean.

2.4.7 The Lead time-Inventory Trade-off

For the general case of an order with multiple components,

$$R_k(t) \approx \left(nW^*(n^{-2}t) - \min_{i\in\mathcal{C}(k)} \left\{\frac{s_i}{\lambda_i}\right\}\right)^+.$$

The limiting process W^* is distributed as an RBM with drift $\theta = \lim_{n\to\infty} n(\rho - 1)$, variance σ^2 and state-space $[0, \infty)$. It is straightforward to verify that the "scaled" process $nW^*(n^{-2}\cdot)$ is distributed as an RBM with drift $\rho - 1$, variance σ^2 and state space $[0, \infty)$. Moreover, it is known that the steady-state distribution of this process is exponential with parameter

$$\tau \equiv \frac{2(1-\rho)}{\sigma^2}. \tag{2.64}$$

Let us write $R_k(\infty)$ to denote the steady state response time for type k orders and let T denote a random variable whose distribution is exponential with parameter

τ. The preceding discussion leads us to the following approximation:

$$\begin{aligned} Pr\{R_k(\infty) > x\} \approx Pr\left\{\left(T - \min_{i \in \mathcal{C}(k)} \left\{\frac{s_i}{\lambda_i}\right\}\right)^+ > x\right\} &= Pr\left(T - \min_{i \in \mathcal{C}(k)} \left\{\frac{s_i}{\lambda_i}\right\} > x\right) \\ &= Pr\left(T > x + \min_{i \in \mathcal{C}(k)} \left\{\frac{s_i}{\lambda_i}\right\}\right) \\ &= e^{-\tau\left(x + \min_{i \in \mathcal{C}(k)} s_i/\lambda_i\right)}. \end{aligned} \tag{2.65}$$

2.4.8 Single-Component Products

In the simplest case of a product that requires a single component type equation (2.65) reduces to

$$Pr\{R(\infty) > x\} \approx e^{-\tau\left(x + \frac{s}{\lambda}\right)}. \tag{2.66}$$

(We again omit subscripts in this discussion as they are not necessary.) As in Glasserman and Wang [21], we want to consider the set of pairs (x, s) such that the above expression equals δ for some $0 < \delta < 1$, resulting in a constant service level of $1 - \delta$. It is evident from (2.66) that there is a simple *linear* trade-off between promised lead time and inventory given by

$$\Delta s = -\lambda \Delta x. \tag{2.67}$$

In words, decreasing promised lead time by Δx while maintaining the same service level requires increasing the component base-stock level by $\lambda \Delta x$; conversely an increase in lead time by Δx implies an allowable decrease of $\lambda \Delta x$ in the base-stock level.

Such a linear trade-off rule was first shown by Glasserman and Wang [21], who analyzed the problem using large-deviation techniques. Their parameter for characterizing the trade-off rule is expressed in terms of the cumulant generating functions of order inter-arrival times and production times. Thus, the Glasserman-Wang parameter uses information from the entire distribution of inter-arrival and production times. Since our parameter is derived from a two-moment method of approximation, we expect it to be less sensitive to the probabilistic distributions of the system. In fact, our trade-off parameter is extremely simple. We find that in a two-moment analysis, the trade-off rule depends only on the first moment of the demand process.

Noting that distributional information may not be available always, Glasserman and Wang also give a two-moment approximation of their trade-off parameter, which they found to work "very well" in their simulation studies. It is straightforward to verify that in a system consisting of a single component and fixed batch size, the two-moment approximation of the Glasserman-Wang parameter agrees with ours; that is, the trade-off parameter is simply $-\lambda$. If we allow variable batch size, it can also be shown that the two-moment approximation of the Glasserman-Wang parameter is asymptotically $-\lambda$ as utilization approaches 100%.

2.4.9 Multiple-Component Products

When a product consists of more than one component type, equation (2.65) suggests that promised lead time can be reduced without affecting service only by increasing inventory of the *certain* components. That is, let $\mathcal{I}_k = \{\ell : \frac{s_\ell}{\lambda_\ell} = \min_{i \in \mathcal{C}(k)} \frac{s_i}{\lambda_i}\}$ be the set of "critical" components for product k. When promised lead time is changed, the base-stock level of critical components should be changed according to the rule

$$\Delta s_i = -\lambda_i \Delta x, \qquad i \in \mathcal{I}_k. \tag{2.68}$$

Moreover, (2.65) suggests that changing the base-stock level of other components will have negligible effects on the promised lead time. For a different setting where each component is produced by its own dedicated resource, Glasserman and Wang [21] showed a similar effect; that is, the base-stock level of only certain "critical" components need to be changed in order to affect the service level.

When we consider the most general case of product orders coming from multiple different arrival streams, the relationship given in (2.68) will still hold for *each* of product streams. That is, if we let x^k denote the promised lead time for product orders from the k^{th} stream, then when the promised lead time is changed, the base-stock should be changed according to the rule

$$\Delta s_i = -\lambda_i \Delta x^k, \qquad i \in \mathcal{I}_k, \qquad k = 1, \ldots, K. \tag{2.69}$$

2.4.10 Lead times

In the backlog case, the primary measure of interest is order response time. By unscaling equation (2.62) of Theorem 3, we obtain the "raw" approximate relationship for the response time of product i orders

$$R_k(t) \approx \left(W(t) - \min_{i \in \mathcal{C}(k)} \left\{ \frac{s_i}{\lambda_i} \right\} \right)^+ .$$

Let $R_k(\infty)$ and $W(\infty)$ be random variables corresponding to the steady-state distribution of the processes R_k and W, respectively. It is a straightforward consequence of the previous equation that

$$\begin{aligned} Pr(R_k(\infty) > x) &\approx Pr\left\{ \left(W(\infty) - \min_{i \in \mathcal{C}(k)} \left\{ \frac{s_i}{\lambda_i} \right\} \right)^+ > x \right\} \\ &= Pr\left(W(\infty) > x + \min_{i \in \mathcal{C}(k)} \left\{ \frac{s_i}{\lambda_i} \right\} \right). \end{aligned}$$

The random variable $W(\infty)$ is a surrogate for the sojourn time of a replenishment order — the time elapsed from when the production resource receives a replenishment order and when it completes the order. Harrison and Nguyen [26] noted that a better approximation can be obtained by regarding $W(\infty)$ as the steady-state *waiting time in the queue only.* That is, the sojourn time is better approximated by $W(\infty) + S$ where the random variable S is the sum of all service times associated with the order. To simplify our calculations, we propose that sojourn time be approximated by $W(\infty) + \mathbf{E}[S]$. Consequently, the steady-state distribution of response time is better approximated by

$$\begin{aligned} Pr(R_k(\infty) > x) &\approx Pr\left(W(\infty) + \sum_{k\in\mathcal{C}(i)} m_k > x + \min_{k\in\mathcal{C}(i)} \left\{\frac{s_k}{\lambda_k}\right\}\right) \\ &\approx e^{-\tau\left(x+\min_{k\in\mathcal{C}(i)} s_k/\lambda_k - \sum_{k\in\mathcal{C}(i)} m_k\right)}. \end{aligned} \tag{2.70}$$

The expected value of $R_k(\infty)$ is easily computed from (2.70). After some simple calculus, we have

$$\begin{aligned} \mathbf{E}[R_k(\infty)] &\approx \mathbf{E}[\left(W(\infty) + \textstyle\sum_{i\in\mathcal{C}(k)} m_i - \min_{i\in\mathcal{C}(k)} \left\{\frac{s_i}{\lambda_i}\right\}\right)^+] \\ &\approx \frac{1}{\tau} e^{-\tau\left(\min_{i\in\mathcal{C}(k)} s_i/\lambda_i - \sum_{i\in\mathcal{C}(k)} m_i\right)} \end{aligned} \tag{2.71}$$

Finally, our empirical experience suggest that a better approximation may be obtained by regarding R_k as the time spent *waiting for components only.* Let A_k denote the assembly time of type k orders. We propose that the total order response time be approximated by

$$R_k(\infty) + A_k$$

where the distribution of R_k is given by (2.70).

2.4.11 Assembly Times

The trade-off rules described above remain valid when assembly times are modeled explicitly as random delays. For this discussion it again is helpful to consider the basic scenario of a product that requires one unit of a single component type. Denote by A the assembly time of the product. With R defined as in (2.63), the total order response time, including assembly time, is $R(t) + A$. Under the scaling convention of (2.7), the scaled order response time is

$$\frac{1}{n}R(n^2t) + \frac{A}{n}.$$

Because there is no congestion associated with the assembly operation, assembly times remain of constant order throughout the sequence of systems (as $n \to \infty$) so the scaled assembly time simply converges to zero. Thus, Theorem 3 still holds when assembly times are modeled explicitly as random delays. Consequently the trade-off rules (2.67) and (2.68) discussed in this section still apply.

2.5 Conclusions

In this part of the dissertation, we presented a model for a make-to-stock assemble-to-order production-inventory system where a single capacitated resource produces components to stock, but they are assembled into orders only upon customer demand. By taking a heavy-traffic approach to the problem, we were able to classify a regime under which the main system process — the workstation workload process — could be approximated by a reflecting diffusion process with a simple distribution function. Using this, we were able to classify certain components as being more critical to the system performance than others — the bottleneck components — as well as identify a simple linear trade-off between promised delivery lead times and component inventory levels. Because we have closed-form distribution functions, we were able to find explicit formulas for all system performance measures of interest. In Chapter 3, we present the results of numerical studies that confirm the accuracy of our performance measure estimates, as well as the conjectured linear trade-off rule.

Chapter 3

Numerical Results for the Single-Workcenter System

3.1 Introduction

In this chapter of the dissertation, we provide numerical experiments to test the approximations derived in Chapter 2. We begin by numerically testing the validity of the linear lead time- inventory tradeoff rule given in (2.69) for the case where backlogging is allowed. Our results for systems

3.2 The Lead time-Inventory Trade-off

In Section 2.4.9 we made the following conjecture: if we increase the base-stock levels of certain critical components, then by decreasing the promised lead time for product delivery by the arrival rate of the component in question, we will incur no change in the probability that the product order will be delivered within the specified time. This trade off rule was captured in (2.69). We now numerically test this prediction.

In order to test the linear trade-off rule, we generated five different systems of varying sizes as characteristics. In general, we sought a balance in the parameters of the systems. That is, the arrival rates of component demand for all components are similar, as are the processing rates at the workstation. In all cases, we considered processing rates that were different from one another so that our results fall outside the class of systems for which exact analysis is possible (see [37, 57]). Furthermore, we set the parameters to ensure that the nominal utilization rate at the workstation that was sufficiently high so as to be within our limiting regime, but not so high as to be unrealistic.

Keeping these mild constraints on the systems in mind, we made few other restrictions on the generation process. Essentially, we sought a mix of arrival and processing times that combined both low- and high-variability distributions. After generating the parameters of the system, we set the base-stock levels and promised lead times for the product streams, and adjusted the promised lead times so that the streams had high, but not identical fill rates. We chose a mix of base-stock levels that would ensure that the chosen bottleneck component be the constraint on the system for a large range of promised lead times. We then simulated for the fill rates. Typically, we looked at a mix of product fill rates that ranged between 90% and 95%. We then used the linear trade-off rule given in (2.69) to generate multiple different systems by decreasing the promised lead times for all product streams by a unit amount, and adjusting the base-stock levels of the critical components accordingly. We did this reduction until the promised lead time of the product stream with the lowest value was zero. In most cases, the demand rate for the critical components were not integer, so the base-stock levels generated by the trade-off were rounded to the nearest integer. The product fill rates for each system were then calculated using discrete event simulation. If our theory is correct, we expect these points to plot to a straight line.

All the simulation experiments were run on the Operation Research Center's 400 Mhz dual-processor Pentium II workstations under UNIX using a simulator custom-written in C. Each experiment consisted of 30 different, independent runs of 1.3 million time units, with the first 300,000 units discarded to get rid of the initial transient. We then calculated approximate 95% confidence intervals in the standard manner (see [42]) for the product fill rates.

As with all the simulation experiments in this dissertation, the length of the simulation runs was chosen so that we could be sure that the transient period did not affect the results and that the period of data collection was long enough to give confidence intervals that gave at least three significant digits of accuracy to the right of the decimal place. To do so, we followed a simple procedure of picking a simulation length and transient period at random, and the doubling it until the results from one iteration to the next were essentially identical.

3.2.1 System I

In the first system, we wanted to confirm our results in an uncomplicated case. We chose both the inter-arrival and processing times to be exponential, and set the parameters so that the component-level arrival rate of the bottleneck component was an integer. This ensured that any noise we observer in the product fill rates is system, rather than due to rounding base-stock levels.

The first system consisted of three components. We set the processing times for each component at the workstation to be exponentially distributed, but are different from one another so that the system in question falls outside the category of multiclass, product-form queueing networks discussed in [1] and [37]. Demand was represented by two independent arrival streams, each bringing one product order, and each with exponentially distributed inter-arrival times. The relevant demand

and processing rates are summarized in the table in Figure 4-2.

The first product arrival stream consisted of a requirement for two type-one components, two type-two component, and one type-three component. The second consisted of one type-one component, three type-two components, and two type-three components. The product orders from each stream are summarized in Table I. It is straightforward to calculate the aggregate component-level demand rates at $\lambda_1 = 4.0$, $\lambda_2 = 6.0$, and $\lambda_3 = 3.5$. Using these, and the processing rates given in the table in Figure 4-2 leads to nominal workstation utilization rates of $\rho = 0.958$.

We set the initial component base-stock levels so that component was the bottleneck component, and followed the generation procedure to develop cases, which we simulated for product fill rates. The results are summarized in the graph in Figure 4-2. Two things are apparent in the graph. First, the linear trade-off rule holds almost exactly. The points for the product fill rates of orders from both demand streams plot to essentially-straight lines. Both exhibit a slight upward drift as the promised delivery lead time increases. We expect that this has to due with the ability to better handle variability in production times that may arise. In other words, when the promised delivery lead time is small, a disruptive event at the workcenter. As the promised delivery lead time increases, the effects of a single bad processing time are felt less, and the level of service increases slightly. Second, the line for Product Stream One is shorter than that of Product Stream Two. This is a result of Product Stream One having been assigned a shorter promised lead time. When this value was reduced to zero, the promised lead time for orders from the second product stream was five.

Product Stream	Product Order	Probability	Comp. One Demand	Comp. Two Demand	Comp. Three Demand
One	1	1.0	2	2	1
Two	1	1.0	1	3	2

Table I: System I: The product order mix for two independent demand streams with exponential inter-arrival times.

3.2.2 System II

In the second system, we wish to test our prediction in the case where the variability in production rates was much lower than the variability in the demand structure. We accomplished this by not only using a high-variability arrival time distribution, but also by allowing the product streams to bring more than one kind of product order, chosen at random upon arrival.

Product Stream	Inter-Demand Distribution	Demand Rate	Demand SCV
One	Exponential	1.50	1.00
Two	Exponential	1.00	1.00
Component	Production Distribution	Production Rate	Production SCV
1	Exponential	13.00	1.00
2	Exponential	15.00	1.00
3	Exponential	14.00	1.00

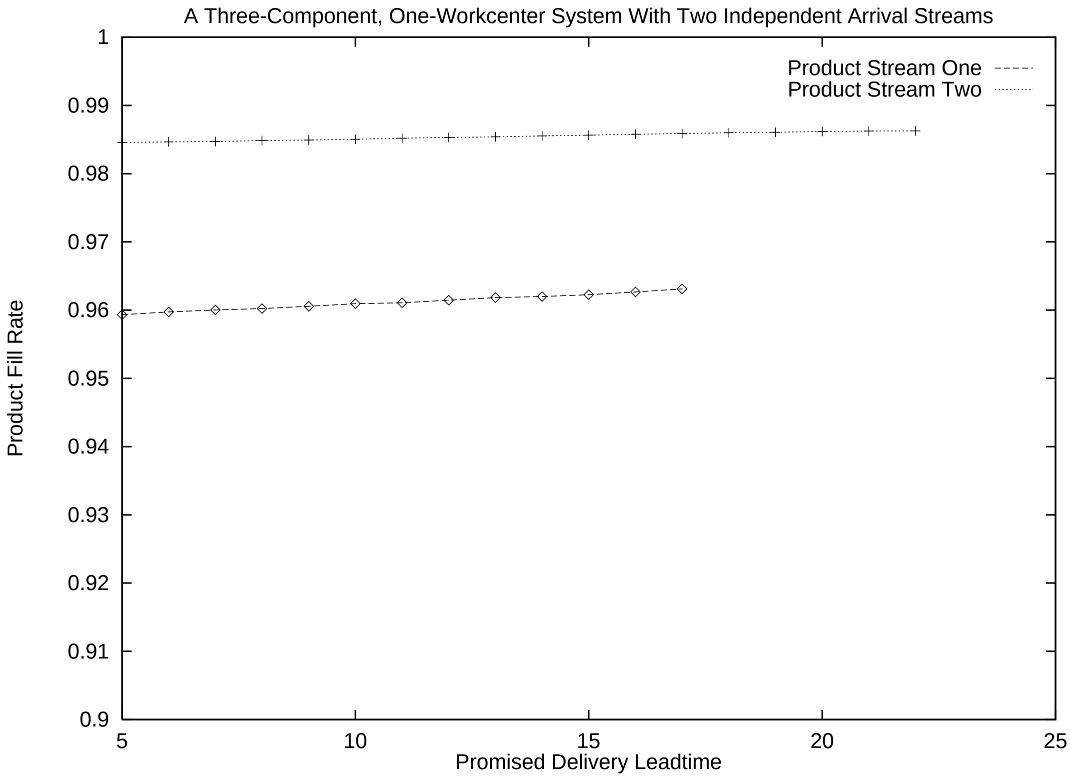

Figure 3-1: System I: Delivery Lead time/Production Rate Tradeoff for a system with exponentially distributed demand inter-arrival times and component production times, and workstation utilization rate $\rho = 0.958$.

The system we generated again consisted of two independent arrival streams placing orders for three components, and we set the processing times for each component at the workstation to be exponentially distributed. Demand was represented by two independent arrival streams, each bringing two different possible product orders, hyperexponentially distributed inter-arrival times. The relevant demand and processing rates are summarized in the table in Figure 4-3.

Unlike the system generated in Section 3.2.1, both the product arrival streams bring with them more than one kind of product order. The product orders from each stream are summarized in Table II. That is, for the first product stream, one-half the time the customer orders two type One components, one Type Two component, and two Type Three component. The other half of the time, he orders two Type One and Type Two components, and one Type Three component. For the second product stream, the customer will order two Type One and Type Three components and three Type Two components 30% of the time, and one Type One and Type Two component, and three Type Three components the remaining 70% of the time. In light of this product order structure, we use the values given in the table in Figure 4-3, and equation 2.2 to calculate the aggregate component-level demand rates as $\lambda_1 = 4.12$, $\lambda_2 = 3.88$, and $\lambda_3 = 5.17$. Using these, and the processing rates given in the table in Figure 4-3 leads to nominal workstation utilization rates of $\rho = 0.951$.

We set the initial component base-stock levels so that Component Two was the critical component, and followed the generation procedure to develop cases, which we simulated for product fill rates. The results are summarized in the graph in Figure 4-3. Because Component Two has a non-integer demand rate, we had to round the base-stock levels generated from the trade-off rule. Despite this, however, we still get lines that are very straight. As before, they lines plotted rise with the promised delivery lead time, and the drift only slightly more pronounced when compared to the results in Section 3.2.1. As conjectured, we expect that this is due to the systems ability to absorb a bad production time when then promised delivery lead time is long.

Product Stream	Product Order	Probability	Comp. One Demand	Comp. Two Demand	Comp. Three Demand
One	1	0.5	2	1	2
	2	0.5	2	2	1
Two	1	0.3	2	3	2
	2	0.7	1	1	3

Table II: System II: The product order mix for two independent demand streams with gamma-distributed inter-arrival times.

Product Stream	Inter-Demand Distribution	Demand Rate	Demand SCV
One	Hyperxponential	1.33	4.00
Two	Hyperxponential	1.18	8.00
Component	Production Distribution	Production Rate	Production SCV
1	Exponential	13.00	1.00
2	Exponential	15.00	1.00
3	Exponential	14.00	1.00

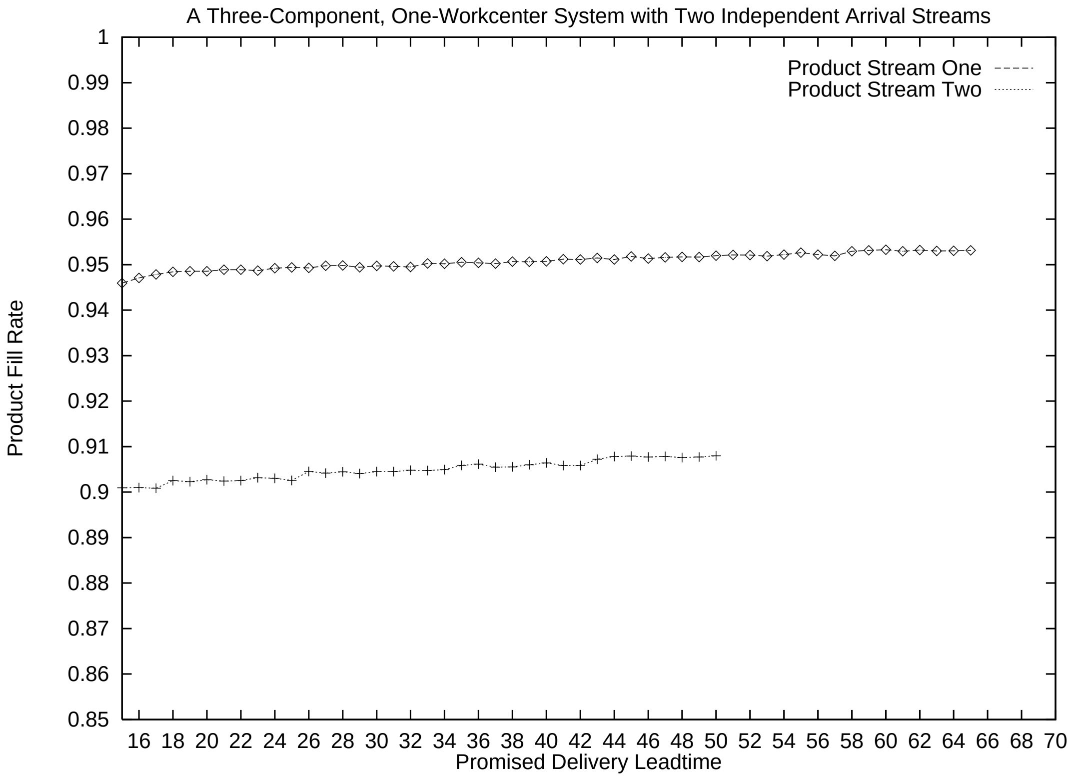

Figure 3-2: System II: Delivery Lead time/Production Rate Tradeoff for a system with hyperexponentially distributed demand inter-arrival times and exponentially-distributed component production times, and workstation utilization rate $\rho = 0.951$.

3.2.3 System III

In the third system, we tested our prediction in the case where the variability in production rates was much higher than the variability in the demand structure. We accomplished this by using high-variability production time distributions, The system we generated again consisted of two independent arrival streams placing orders for three components, and we set the processing times for each component at the workstation to be hyperexponentially distributed. Demand was represented by two independent arrival streams, each bringing two different possible product orders under exponentially distributed inter-arrival times. The relevant demand and processing rates are summarized in the table in Figure 4-4. As with the system in Section 3.2.2, we allowed each arrival stream to have a random product order mix, summarized in Table III. Using values given in the table in Figure 4-4, and equation 2.2, we calculate the aggregate component-level demand rates as $\lambda_1 = 3.95$, $\lambda_2 = 4.00$, and $\lambda_3 = 5.45$. Using these, and the processing rates given in the table in Figure 4-4 leads to nominal workstation utilization rates of $\rho = 0.970$.

We set the initial component base-stock levels so that Component Two was the critical component, and followed the generation procedure to develop cases, which we simulated for product fill rates. The results are summarized in the graph in Figure 4-4. In this case, as in Section 3.2.1, the arrival rate of the critical component is integer. This results in no rounding requirement when setting the base-stock levels. The lines plotted are still very straight, with some slight saw-toothing for the lower product fill rate stream. As before, we observe a slight upward drift in the lines with increasing promised lead time, which we attribute to the ability of the workcenter to recover from a long processing job. What is interesting to note, however, is that the drift does not seem to be substantially different from that observed in the systems in Sections 3.2.1 and 3.2.2, although the processing times have a much higher degree of variability. We do not have an explanation why this would be the case.

Product Stream	Product Order	Probability	Comp. One Demand	Comp. Two Demand	Comp. Three Demand
One	1	0.4	2	1	2
	2	0.6	2	2	1
Two	1	0.3	2	3	2
	2	0.7	1	1	3

Table III: System III: The product order mix for two independent demand streams with exponential inter-arrival times.

Product Stream	Inter-Demand Distribution	Demand Rate	Demand SCV
One	Exponential	1.00	1.00
Two	Exponential	1.50	1.00
Component	Production Distribution	Production Rate	Production SCV
1	Hyperexponential	13.33	4.00
2	Hyperexponential	12.50	8.00
3	Hyperexponential	15.38	16.00

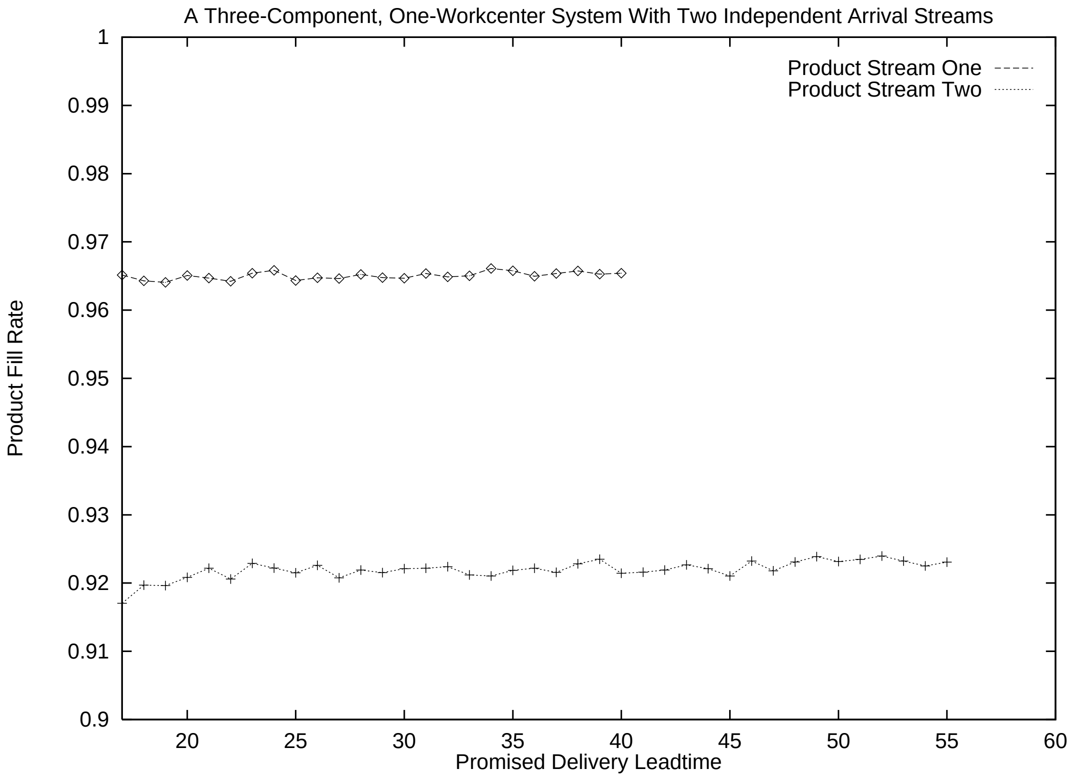

Figure 3-3: System III: Delivery Lead time/Production Rate Tradeoff for a system with exponentially distributed demand inter-arrival times and hyperexponentially distributed component production times, and workstation utilization rate $\rho = 0.970$.

3.2.4 System IV

In the fourth system, we again tested our prediction in the case where the variability in production rates was much lower than the variability in the demand structure, but we made the difference more extreme by replacing the exponential processing times of the system in Section 3.2.2 by low-variability Erlang distributions. The relevant demand and processing rates are summarized in the table in Figure 4-5. We allowed each arrival stream to have a random product order mix, summarized in Table IV. Using values given in the table in Figure 4-5, and equation 2.2, we calculate the aggregate component-level demand rates as $\lambda_1 = 2.63$, $\lambda_2 = 2.66$, and $\lambda_3 = 3.63$. Using these, and the processing rates given in the table in Figure 4-5 leads to nominal workstation utilization rates of $\rho = 0.971$.

In this system, we set Component Two to be the critical component, followed the generation procedure to develop cases, which we simulated for product fill rates. The results are summarized in the graph in Figure 4-5. In this case, as in Section 3.2.2, the arrival rate of the critical component is not an integer, so we rounded the generated base-stock levels. The lines plotted are still very straight, again with a slight upward drift. Indeed, given the complicated nature of the system, it is surprising how well the predicted trade-off rule performs.

Product Stream	Product Order	Probability	Comp. One Demand	Comp. Two Demand	Comp. Three Demand
One	1	0.4	2	1	2
	2	0.6	2	2	1
Two	1	0.3	2	3	2
	2	0.7	1	1	3

Table IV: System IV: The product order mix for two independent demand streams with gamma-distributed inter-arrival times.

3.2.5 System V

In the third fifth and final system, we again tested our prediction in the case where the variability in production rates was much higher than the variability in the demand structure, but we made the difference more extreme by replacing the exponential inter-demand times of the system in Section 3.2.3 by Erlang distributions. We also considered a system with three independent product arrival streams instead of two. The relevant demand and processing rates are summarized in the table in Figure 4-6. We allowed each arrival stream to have a random product order mix, summarized in

Product Stream	Inter-Demand Distribution	Demand Rate	Demand SCV
One	Hyperexponential	0.67	4.00
Two	Hyperexponential	1.00	8.00
Component	Production Distribution	Production Rate	Production SCV
1	2-Erlang	10.00	0.50
2	3-Erlang	9.80	0.33
3	4-Erlang	8.33	0.25

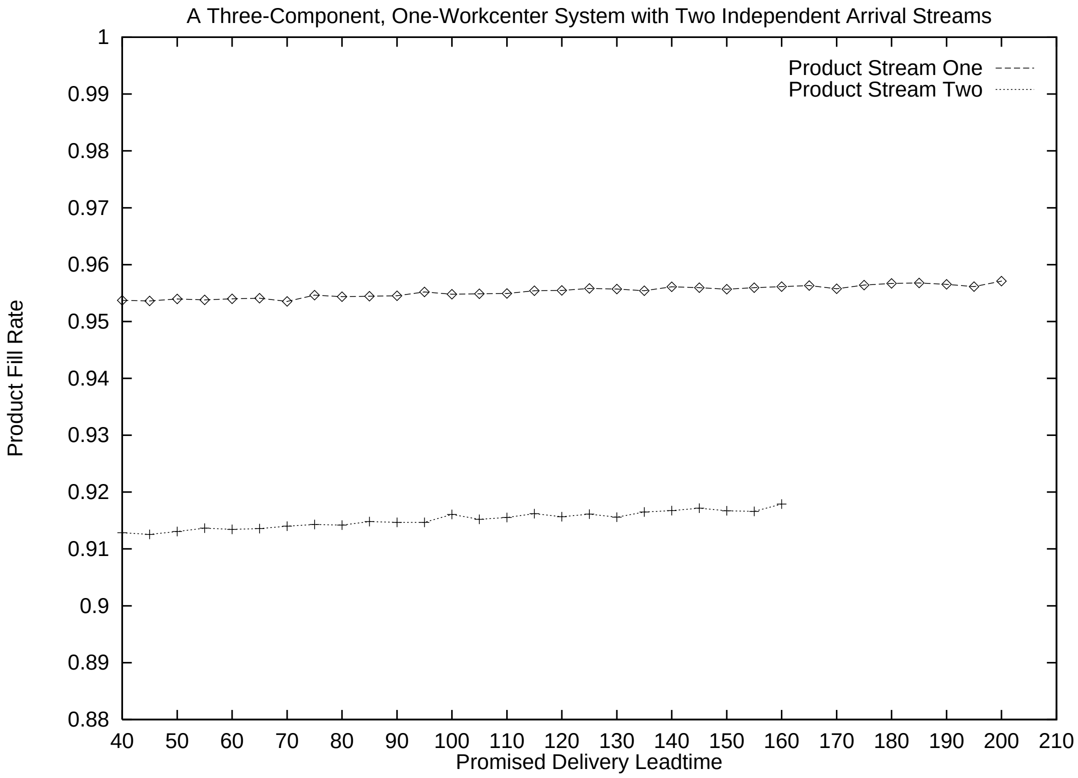

Figure 3-4: System IV: Delivery Lead time/Production Rate Tradeoff for a system with hyperexponentially- distributed demand inter-arrival times and Erlang-distributed component production times, and workstation utilization rate $\rho = 0.971$.

Table V. Using values given in the table in Figure 4-6, and equation 2.2, we calculate the aggregate component-level demand rates as $\lambda_1 = 4.30$, $\lambda_2 = 5.05$, and $\lambda_3 = 5.65$. Using these, and the processing rates given in the table in Figure 4-5 leads to nominal workstation utilization rates of $\rho = 0.958$.

In this system, we set Component One to be the critical component, followed the generation procedure to develop cases, which we simulated for product fill rates. The results are summarized in the graph in Figure 4-6. In this case, as in Sections 3.2.2 and 3.2.4, the arrival rate of the critical component is not an integer, so we rounded the generated base-stock levels. The lines plotted are still very straight, again with a slight upward drift. As with the system in Section 3.2.3, we observe some saw-toothing in the lower product fill rate streams, a phenomenon which becomes more pronounced as the desired product fill rate decreases. The behavior is consistent with the results found in Glasserman and Wang [21].

Product Stream	Product Order	Probability	Comp. One Demand	Comp. Two Demand	Comp. Three Demand
One	1	0.4	1	2	1
	2	0.6	2	2	1
Two	1	0.3	1	3	2
	2	0.7	2	2	4
Three	1	0.5	2	1	2
	2	0.5	2	2	3

Table V: System V: The product order mix for two independent demand streams with gamma-distributed inter-arrival times.

3.3 Product Fill Rates, Backorder

In order to test the accuracy of the performance measure approximations for product fill rates in the case where backorder is allowed, we simulated two systems, and compared the approximations to the simulated results.

3.3.1 System VI

The first system we simulated consisted of three components with two independent demand arrival streams. The relevant interarrival and processing time data is summarized in Table VI. The product mix is summarized in Table VII. In light of these,

Product Stream	Inter-Demand Distribution	Demand Rate	Demand SCV
One	2-Erlang	1.00	0.50
Two	3-Erlang	1.00	0.33
Three	4-Erlang	0.50	0.25
Component	Production Distribution	Production Rate	Production SCV
1	Hyperexponential	15.38	2.33
2	Hyperexponential	18.18	3.31
3	Hyperexponential	13.33	1.78

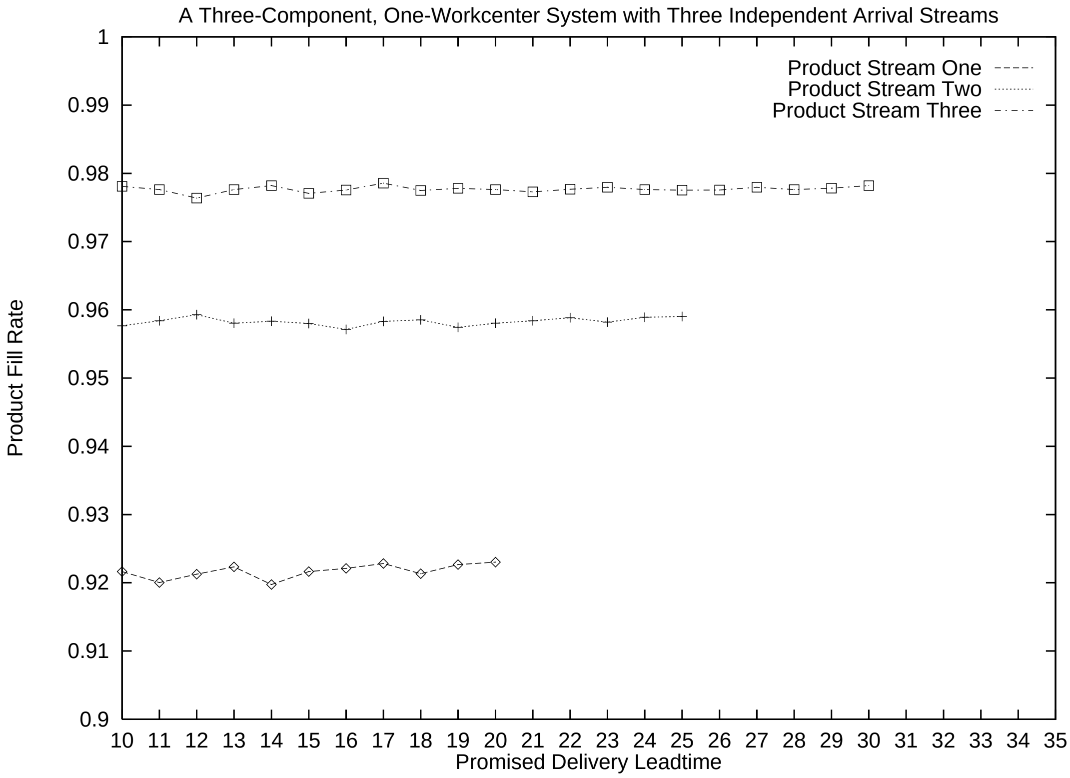

Figure 3-5: System V: Delivery Lead time/Production Rate Tradeoff for a system with exponentially distributed demand inter-arrival times and component production times, and workstation utilization rate $\rho = 0.958$.

Product Stream	Inter-Demand Distribution	Demand Rate	Demand SCV
One	Beta	1.1	1.50
Two	Beta	2.0	2.65
Component	**Production Distribution**	**Production Rate**	**Production SCV**
1	Exponential	13.00	1.00
2	2-Erlang	11.00	0.50
3	3-Erlang	15.00	0.33

Table VI: System VI: Interarrival and processing time distributions.

Product Stream	Product Order	Probability	Comp. One Demand	Comp. Two Demand	Comp. Three Demand
One	1	1	2	1	1
Two	1	1	1	2	1

Table VII: System VI: The product order mix for two independent demand streams with gamma-distributed inter-arrival times.

it is straight forward to calculate that $\lambda_1 = 4.16$,$\lambda_2 = 5.02$, and $\lambda_3 = 3.06$. The average utilization of the system is $\rho_0 = 0.980$.

In Case I of Table VIII we increase the first component base-stock until it is no longer the critical component, while holding the other two base-stock levels fixed. When the component base-stock level exceeds 20, it is no longer the critical component, and we would expect that further increases would have no effect on our product fill rates. In Case II we perform exactly the same experiment, this time with the critical component being the second. When its value exceeds 45, it ceases to be the bottleneck on the system, and further increases in initial base-stock level have a negligible effect on product fill rates. In both cases, the estimates for product fill rates are quite accurate, with almost all being within one percent of simulation.

Finally, Table IX summarizes the approximated lead times for System VI. The numbers in parentheses next to the simulation values are one-half the size of the 95% confidence interval, as percent of simulated value. The numbers in parentheses next to the heavy-traffic values are the percent error of the approximated value from simulation. In general, we find that the results of the approximation are quite accurate, always within 10% of simulation, and often within 5%.

Base-stock				Product 1		Product 2	
	s_1	s_2	s_3	Sim	HT	Sim	HT
I	10	30	30	0.951(0.4)	0.959(0.9)	0.951(0.4)	0.959(0.8)
	20	30	30	0.956(0.4)	0.964(0.8)	0.956(0.4)	0.964(0.8)
	30	30	30	0.959(0.4)	0.966(0.8)	0.958(0.4)	0.966(0.8)
	40	30	30	0.959(0.3)	0.966(0.7)	0.958(0.4)	0.966(0.8)
II	60	35	30	0.959(0.3)	0.968(0.9)	0.958(0.4)	0.968(1.0)
	60	40	30	0.961(0.3)	0.969(0.9)	0.960(0.3)	0.969(1.0)
	60	45	30	0.963(0.3)	0.971(0.8)	0.962(0.3)	0.971(0.9)
	60	50	30	0.964(0.3)	0.972(0.8)	0.964(0.3)	0.972(0.9)
	60	55	30	0.965(0.3)	0.972(0.7)	0.965(0.3)	0.972(0.7)

Table VIII: Product fill rates for System VI.

Base-stock				Product 1		Product 2	
	s_1	s_2	s_3	Sim	HT	Sim	HT
I	10	30	30	18.4(1.9)	17.6(-4.4)	18.3(1.9)	17.6(-3.6)
	20	30	30	16.5(2.1)	15.6(-5.3)	16.4(2.1)	15.6(-5.1)
	30	30	30	15.5(2.2)	14.7(-5.2)	15.8(2.2)	14.7(-6.9)
	40	30	30	15.4(2.2)	14.7(-4.4)	15.8(2.2)	14.7(-6.6)
II	60	35	30	15.4(2.2)	14.7(-4.4)	15.8(2.2)	14.7(-6.6)
	60	40	30	14.7(2.3)	14.0(-4.7)	15.0(2.3)	14.0(-6.9)
	60	45	30	14.0(2.4)	13.3(-5.0)	14.3(2.4)	13.3(-7.2)
	60	50	30	13.4(2.5)	12.7(-5.7)	13.7(2.5)	12.7(-7.6)
	60	55	30	13.1(2.6)	12.1(-7.1)	13.2(2.5)	12.1(-8.4)

Table IX: Average lead times for System VI.

3.3.2 System VII

Product Stream	Inter-Demand Distribution	Demand Rate	Demand SCV
One	2-Erlang	1.10	0.50
Two	3-Erlang	0.95	0.33
Three	4-Erlang	1.00	0.25
Component	Production Distribution	Production Rate	Production SCV
1	Exponential	16.50	1.00
2	Exponential	15.00	1.00
3	Exponential	14.50	1.00

Table X: Interarrival and processing time distributions for System VII.

Product Stream	Product Order	Probability	Comp. One Demand	Comp. Two Demand	Comp. Three Demand
One	1	0.4	1	1	1
	2	0.4	1	2	1
	3	0.2	3	2	1
Two	1	0.2	1	2	1
	2	0.3	2	2	1
	3	0.5	1	1	3
Three	1	0.1	3	2	2
	2	0.1	3	1	3
	3	0.8	2	2	2

Table XI: The product order mix for three independent demand streams with mixed inter-arrival times for System VII

The second system we simulated consisted of three components with three independent demand arrival streams. The relevant interarrival and processing time data is summarized in Table X. The product mix is summarized in Table XI. In light of these, it is straight forward to calculate that $\lambda_1 = 4.98$,$\lambda_2 = 4.90$, and $\lambda_3 = 5.10$. The average utilization of the system is $\rho_0 = 0.980$.

In Table XII no specific experiments were performed. Rather, we simulated to gauge the accuracy of the heavy-traffic approximation over a large number of combinations of base-stock levels. The number under the simulation result is one-half the length of the 95% confidence interval, as a percent of the simulated value. The number under the heavy-traffic approximation of the fill rate is the error of the approximation as percent of the simulated value. In general, the approximation of product fill rates is within one percent of the values given by discrete-event simulation, but is worse for smaller base-stock levels. As expected, the accuracy of the approximation increases with total base-stock level.

Table XIII records the approximation for lead times. The number in parentheses next to the simulation result is one-half the length of the 95% confidence interval, as a percent of the simulated value. The number in parentheses next to the heavy-traffic approximation of the fill rate is the error of the approximation as percent of the simulated value. In general, we find that the results are not very accurate in this case.

3.4 Performance Measures, Lost Sales

Finally, we wish to explore the behavior of the approximation in the case of lost sales. We consider a very simple system — called System VIII — with three independent product arrival streams, and three components. The relevant interarrival and processing time data is summarized in Table XIV. The product mix is summarized in Table XV. In light of these, it is straight forward to calculate that $\lambda_1 = 1.60$,$\lambda_2 = 1.35$, and $\lambda_3 = 1.25$. The average utilization of the system is $\rho_0 = 0.947$.

We wish to analyze the behavior of this system for different initial base-stock levels, and see how it behaves when these are changed. Table XVI gives the base-stock and allowable backlog levels for four different cases. We address each case one at a time.

In Case 1, we keep the initial base-stock levels of the second two components fixed, and increase the base-stock level of Component One in step sizes of two. We have set the base-stock levels so that Component One is initially the bottleneck, and becomes successively less so as its base-stock level increases. The status of Component One as the bottleneck is given in the third column of the table. For the first three configurations of base-stock levels, Component One is the bottleneck. In the fourth, the base-stock level of Component One has been increased enough so that Component Three becomes the bottleneck, and Component One moves into the second position. In the fifth and last configuration, Component One moves into the last position, changing places with item two.

We would expect that products that depend on the bottleneck item will have lower acceptance rates, in general, than those that do not. We computed product acceptance rates using our approximation, and compared the results with discrete event simulation. The results are recorded in Table XVII. Products One and Three depend on Component One, and when Component One is the bottleneck, they do

Base-Stock			**Product 1**		**Product 2**		**Product 3**	
s_1	s_2	s_3	Sim	HT	Sim	HT	Sim	HT
30	40	10	0.956	0.934	0.934	0.909	0.883	0.852
			0.3	-2.2	0.4	-2.7	0.4	-3.5
30	40	15	0.961	0.944	0.943	0.922	0.901	0.874
			0.3	-1.8	0.3	-2.2	0.4	-3.0
30	40	20	0.966	0.952	0.950	0.934	0.915	0.892
			0.3	-1.5	0.3	-1.8	0.4	-2.5
30	40	25	0.972	0.959	0.959	0.943	0.928	0.908
			0.2	-1.4	0.3	-1.6	0.3	-2.2
30	40	30	0.978	0.965	0.967	0.952	0.942	0.922
			0.2	-1.3	0.3	-1.6	0.3	-2.2
30	40	50	0.979	0.966	0.970	0.953	0.947	0.923
			0.2	-1.4	0.3	-1.8	0.3	-2.5
35	40	50	0.980	0.971	0.972	0.960	0.952	0.935
			0.2	-0.9	0.2	-1.3	0.3	-1.8
40	40	50	0.985	0.975	0.978	0.966	0.961	0.945
			0.1	-0.8	0.2	-1.2	0.3	-1.7
45	40	50	0.984	0.976	0.977	0.967	0.961	0.946
			0.1	-0.8	0.2	-1.1	0.2	-1.5
50	45	50	0.988	0.980	0.982	0.972	0.968	0.954
			0.2	-0.8	0.2	-1.1	0.3	-1.4
50	50	50	0.988	0.982	0.982	0.974	0.969	0.959
			0.2	-0.7	0.2	-0.8	0.3	-1.1
50	55	50	0.989	0.982	0.984	0.974	0.971	0.959
			0.1	-0.8	0.2	-1.0	0.2	-1.3
60	60	70	0.992	0.987	0.989	0.982	0.980	0.971
			0.1	-0.5	0.1	-0.6	0.2	-0.9
70	70	80	0.995	0.991	0.992	0.987	0.986	0.979
			0.0	-0.4	0.1	-0.5	0.1	-0.7

Table XII: Product acceptance rates for System VII.

have lower fill rates than product two, which does not depend on the bottleneck item. In the first configuration, where the allowable backlog level of Component One is very low, Component One is strong in its position as the bottleneck item. We would expect that Components Two and Three would then almost never stock out, and this is born out by the acceptance rate of Product Two, which is almost 100%. As Component One's capacity increases in configurations two and three, the acceptance rates of the products that depend on it increase, and the acceptance rate of Product Two, which does not depend on Component One, decreases. This is because Product Two requires Component Three, which is in second position. As Component One becomes less clearly the bottleneck on the system, Component Three stocks out more often, and the acceptance rate of products that depend on item three decline as well.

Table XVIII records the results of comparisons of simulation with the approximation for average buffer levels. The 95% confidence intervals are given after the simulation numbers in parentheses, and the value given is the half-length of the confidence interval as a percent of the simulation value. As the initial base-stock level of Component One increases, and as the fill rate of the products that depend on Component One increase, the average buffer level of Component One decreases. This is because more product orders, and thus more component orders are being filled.

In Case 2 we begin with Component Two as the bottleneck, and decrease its initial base-stock level. As expected, the acceptance rates of products that do not require Component Two increase and the acceptance rates of products that do require Component Two decrease. Also, the average buffer levels of all items increase as more products are turned away.

In Cases 3 and 4 we examine the effects of increasing initial base-stock level with fixed allowable backlog levels. We do so by increasing the initial base-stock level of item three gradually so that it becomes less and less the bottleneck on the system. Product One, which does not require item three, has high acceptance rates which decline slightly as Component Two, which it does require, becomes the bottleneck on the system. Products two and three see an increase in their acceptance rates as the capacity to accept orders for item three increases. Since acceptance rates are high already, even large changes the initial base-stock level of item three have negligible effects on the average base-stock levels of items one and two. The lead times of product one stays relatively static, while those for products two and three decrease as the base-stock level of item three increases. The same behavior in performance measure is displayed when Component One's initial base-stock level is increased until it is no longer the bottle neck.

Base-Stock			**Product 1**		**Product 2**		**Product 3**	
s_1	s_2	s_3	Sim	HT	Sim	HT	Sim	HT
30	40	10	4.0(2.0)	4.8(20.0)	4.0(1.9)	4.8(20.0)	4.0(1.9)	4.8(20.0)
30	40	15	3.4(1.9)	4.1(24.2)	3.4(1.9)	4.1(24.2)	3.4(1.9)	4.1(24.2)
30	40	20	2.8(2.4)	3.6(28.6)	2.9(2.3)	3.6(28.6)	2.9(2.3)	3.6(28.6)
30	40	25	2.4(2.5)	3.1(29.1)	2.4(2.5)	3.1(29.1)	2.4(2.5)	3.1(29.1)
30	40	20	2.0(3.5)	2.6(30.0)	2.0(3.5)	2.6(30.0)	2.0(3.4)	2.6(30.0)
30	40	25	1.8(2.4)	2.6(44.4)	1.8(2.5)	2.6(44.4)	1.8(2.4)	2.6(44.4)
35	40	50	1.6(3.9)	2.2(37.5)	1.6(3.9)	2.2(37.5)	1.6(3.8)	2.2(37.5)
40	40	50	1.3(3.8)	1.9(46.2)	1.3(3.8)	1.9(46.2)	1.3(3.8)	1.9(46.2)
45	40	50	1.3(3.7)	1.9(46.2)	1.3(3.7)	1.9(46.2)	1.3(3.7)	1.9(46.2)
50	45	50	1.4(4.2)	1.9(35.7)	1.4(4.2)	1.9(35.7)	1.4(4.2)	1.9(35.7)
50	50	50	1.1(5.5)	1.5(36.4)	1.1(5.4)	1.5(36.4)	1.1(5.4)	1.5(36.4)
50	55	50	1.0(4.1)	1.5(50.0)	1.0(4.0)	1.5(50.0)	1.0(4.0)	1.5(50.0)
60	60	70	0.7(5.7)	1.1(57.1)	0.7(5.7)	1.1(57.1)	0.7(5.7)	1.1(57.1)
70	70	80	0.5(7.5)	0.8(60.0)	0.5(7.4)	0.8(60.0)	0.5(7.4)	0.8(60.0)

Table XIII: Average lead times for System VII.

Product Stream	Inter-Demand Distribution	Demand Rate	Demand SCV
One	Exponential	1.10	1.00
Two	Exponential	0.95	1.00
Three	Exponential	1.00	1.00
Component	Production Distribution	Production Rate	Production SCV
1	Exponential	4.40	1.00
2	Exponential	4.00	1.00
3	Exponential	5.00	1.00

Table XIV: Interarrival and processing time distributions for System VIII.

Product Stream	Comp. One Demand	Comp. Two Demand	Comp. Three Demand
One	1	1	0
Two	0	1	1
Three	1	0	1

Table XV: The product order mix for three independent demand streams with exponential inter-arrival times for System VIII

Case 1

s_1	s_2	s_3	Bottleneck	2nd Item	3rd Item
12	16	14	Item 1	Item 3	Item 2
14	16	14	Item 1	Item 3	Item 2
16	16	14	Item 1	Item 3	Item 2
18	16	14	Item 3	Item 1	Item 2
20	16	14	Item 3	Item 2	Item 1

Case 2

s_1	s_2	s_3	Bottleneck	2nd Item	3rd Item
20	15	14	Item 2	Item 3	Item 1
20	14	14	Item 2	Item 3	Item 1
20	13	14	Item 2	Item 3	Item 1
20	12	14	Item 2	Item 3	Item 1

Case 3

s_1	s_2	s_3	Bottleneck	2nd Item	3rd Item
24	22	20	Item 3	Item 2	Item 1
24	24	20	Item 3	Item 1	Item 2
24	24	22	Item 1	Item 3	Item 2
24	24	24	Item 1	Item 2	Item 3
24	24	25	Item 1	Item 2	Item 3

Case 4

s_1	s_2	s_3	Bottleneck	2nd Item	3rd Item
30	24	25	Item 2	Item 1	Item 3
32	24	25	Item 2	Item 3	Item 1
34	24	25	Item 2	Item 3	Item 1
36	24	25	Item 2	Item 3	Item 1

Table XVI: Base-stock levels and bottleneck components in System VIII.

Case 1

Product 1		Product 2		Product 3	
Sim.	HT	Sim.	HT	Sim.	HT1
0.951	0.955(0.4)	0.995	0.985(-1.0)	0.950	0.953(0.3)
0.962	0.962(0.1)	0.986	0.983(-0.3)	0.958	0.960(0.2)
0.972	0.968(-0.4)	0.976	0.982(0.6)	0.965	0.966(0.1)
0.979	0.983(0.5)	0.968	0.966(-0.2)	0.970	0.966(-0.4)
0.982	0.985(0.3)	0.963	0.966(0.3)	0.972	0.968(-0.4)

Case 2

Product 1		Product 2		Product 3	
Sim.	HT	Sim.	HT	Sim.	HT1
0.977	0.971(-0.6)	0.960	0.966(0.7)	0.976	0.983(0.7)
0.971	0.968(-0.3)	0.957	0.964(0.7)	0.980	0.984(0.4)
0.964	0.965(0.1)	0.953	0.961(0.8)	0.984	0.985(0.1)
0.957	0.961(0.4)	0.949	0.957(0.8)	0.989	0.985(-0.4)

Case 3

Product 1		Product 2		Product 3	
Sim.	HT	Sim.	HT	Sim.	HT1
0.990	0.992(0.2)	0.982	0.983(0.1)	0.988	0.984(-0.4)
0.993	0.992(-0.1)	0.983	0.984(0.1)	0.986	0.984(-0.2)
0.990	0.988(-0.2)	0.987	0.991(0.4)	0.989	0.988(-0.1)
0.989	0.988(-0.1)	0.989	0.992(0.3)	0.991	0.989(-0.2)

Case 4

Product 1		Product 2		Product 3	
Sim.	HT	Sim.	HT	Sim.	HT1
0.988	0.988(0.0)	0.990	0.992(0.2)	0.992	0.989(-0.3)
0.989	0.989(0.0)	0.989	0.988(-0.1)	0.994	0.993(-0.1)
0.990	0.989(-0.1)	0.988	0.988(0.0)	0.995	0.994(-0.1)
0.990	0.990(0.0)	0.987	0.988(0.1)	0.995	0.994(-0.1)
0.991	0.990(-0.1)	0.987	0.988(0.1)	0.995	0.995(0.0)

Table XVII: Product acceptance rates for System VIII.

Case 1

Comp. 1			Comp. 2			Comp. 3		
Sim.	HT	% Error	Sim.	HT	% Error	Sim.	HT	% Error
5.2(0.7)	5.0	-3.6	4.6(0.7)	4.7	1.2	3.9(0.7)	4.0	2.8
4.4(1.1)	4.4	0.4	3.9(1.2)	4.2	6.4	3.2(1.2)	3.5	8.0
3.8(1.6)	3.8	0.3	3.4(1.6)	3.7	6.7	2.8(1.7)	3.1	7.8
3.4(1.9)	3.2	-8.4	3.2(1.7)	3.2	1.5	2.6(1.8)	2.7	4.6
3.3(2.0)	3.1	-3.5	3.1(1.8)	3.2	4.8	2.5(1.9)	2.7	7.7

Case 2

Comp. 1			Comp. 2			Comp. 3		
Sim.	HT	% Error	Sim.	HT	% Error	Sim.	HT	% Error
3.4(2.1)	3.2	-7.2	3.3(1.8)	3.4	3.1	2.6(2.0)	2.7	2.3
3.7(1.8)	3.5	-4.7	3.5(1.5)	3.6	3.6	2.8(1.7)	2.9	4.4
4.0(1.4)	3.8	-4.1	3.8(1.2)	3.9	3.1	3.0(1.3)	3.2	5.1

Case 3

Comp. 1			Comp. 2			Comp. 3		
Sim.	HT	% Error	Sim.	HT	% Error	Sim.	HT	% Error
4.4(1.2)	4.2	-5.3	4.2(1.0)	4.2	1.3	3.3(1.2)	3.5	3.8
5.4(2.2)	5.5	1.7	4.5(2.3)	4.8	6.3	3.1(2.9)	3.4	9.0
5.3(2.7)	5.5	3.1	6.3(2.0)	6.8	7.1	3.5(3.1)	3.4	-1.7
5.1(2.8)	5.1	0.0	6.2(2.0)	6.5	4.8	4.7(2.3)	5.0	5.8

Case 4

Comp. 1			Comp. 2			Comp. 3		
Sim.	HT	% Error	Sim.	HT	% Error	Sim.	HT	% Error
5.1(3.2)	5.1	-0.5	6.2(2.3)	6.5	4.3	6.7(2.0)	7.0	5.4
5.2(2.9)	5.1	-1.5	6.2(2.1)	6.5	3.6	7.7(1.6)	8.0	4.6
7.1(2.4)	7.0	-1.2	6.3(2.3)	6.4	2.8	7.7(1.7)	8.0	3.9
9.0(1.9)	9.0	-0.1	6.3(2.2)	6.4	2.5	7.7(1.7)	8.0	3.7
11.0(1.6)	11.0	-0.5	6.4(2.3)	6.4	1.0	7.8(1.8)	8.0	2.6
13.2(1.3)	13.0	-1.7	6.5(2.2)	6.4	-1.4	7.9(1.7)	8.0	0.6

Table XVIII: Average Finished Goods Inventory Levels for System VIII.

Chapter 4

A Production-Inventory System With Multiple Capacitated Production Resources

4.1 Introduction

A natural question that arises from Chapter 2 is whether or not the results are generalizable to an arbitrary number of components and workstations. In this Chapter we present exactly such an analysis. Following a procedure that has become relatively standard, we state a heavy-traffic limit theorem for a production-inventory system similar to the one studied in the full backorder case in Chapter 2, but with the relaxation of the assumption that all components are processed at a single workstation. We identify the limiting diffusion process, and derive some special properties specific to our case. We then present an extension of lead time-inventory tradeoff discussed in Chapter 2 in the case where we require that every product require at least one unit of every component. We provide numerical evidence in the form of simulation experiments similar in nature to those performed in Section 3.2.

4.2 System Model

In this chapter we consider an extension of the production-inventory system studied in Chapter 2, where we relax the assumption that all components are produced at a single workstation. Specifically, we propose to study a production-inventory system where multiple workcenters produce multiple components to stock, where they are held until they are assembled into orders upon customer demand. The system under consideration is depicted in Figure 4-1. Specifically, our system consists of c components that are produced at J different workcenters. Components are held in finished goods inventories, each of which has a finite, fixed initial base-stock level —

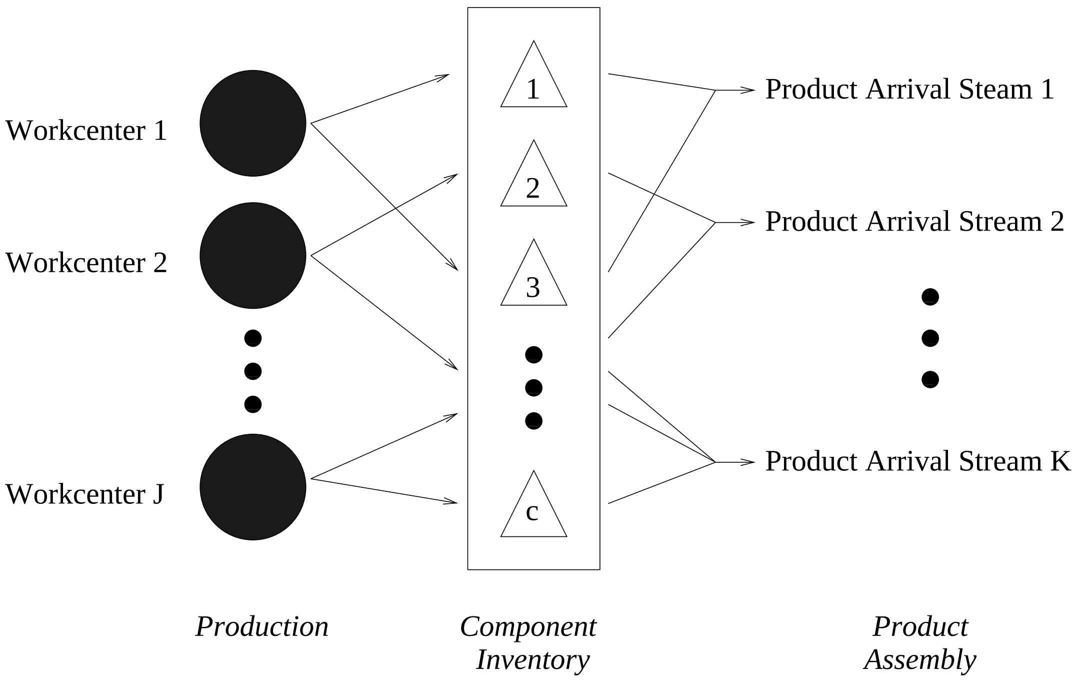

Figure 4-1: A production-inventory system with multiple capacitated resources.

where the base-stock level for component i is denoted s_i — and each of which may be infinitely backlogged. Demand in the form of customer orders arrives from K different demand streams, and we assume that every customer will demand at least one unit of every component. If all components ordered are on-hand, the order is filled immediately, and the customer departs. If at least one of the component orders cannot be filled, the entire product order is backlogged until all components ordered are available. Backlogged product orders are filled in a first-come first-serve (FCFS) manner. When component orders are placed — be they replacement orders for components delivered, or backlog orders — they are sent to the workcenters, where they are served on a FCFS basis. We allow different components to be serviced at different workcenters, and each workcenter can process more than one kind of component, but components must be processed at the workcenters to which they are defined. In particular, we do not allow for one workcenter to use excess capacity to help another, nor do we consider such polices as join the shortest queue.

Associated to every demand stream is a *promised delivery lead time*, a finite, fixed, non-negative length of time in which all product orders from the stream are promised to be filled. Our main performance measure of interest is the *product fill rate*, which is the proportion of time that a product order from a demand stream is filled within that demand stream's promised delivery lead time. The main focus of this chapter will be to investigate the relationship between the product fill rates as a function of the promised delivery lead times and the initial component base-stock levels.

4.3 Network Analysis

The following development closely follows that of [26]. In what follows, all vectors are taken to be column vectors. Let the system have c different components which are processed at d different workstations, where $c \geq d$. We denote the workstation that processes component order type i by $s(i)$. Define $\mathcal{C}(j)$ to be the *constituency* of workstation j, $j = 1, \ldots, J$. Then

$$\mathcal{C}(j) = \{i | s(i) = j, i = 1, \ldots, c\} \tag{4.1}$$

Define C to be the $d \times c$ *constituency matrix* in the following manner:

$$C_{ji} = \begin{cases} 1 & \text{if } i \in \mathcal{C}(j) \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

Then the c-dimensional row vector C_j^T represents the constituency of workstation j.

4.3.1 System Processes

The following defines two fundamental vector processes:

$$\begin{aligned} D(t) &: \text{ cumulative number of product demands in } [0, t] \\ S(n) &: \text{ sum of the first } n \text{ processing times} \end{aligned}$$

Here, D is mnemonic for "demand", and S mnemonic for "service". We will define the following system processes in terms of these two fundamental processes:

$$\begin{aligned} A(t) &: \text{ cumulative number of component demands in } [0, t], \\ L(t) &: \text{ cumulative amount of work sent to the workstations in } [0, t], \\ Y(t) &: \text{ cumulative amount of idle time at the workstations, in } [0, t], \\ W(t) &: \text{ instantaneous amount of work at the workstations at time } t \\ V(t) &: \text{ cumulative number of components to finish processing in } [0, t] \\ N(t) &: \text{ number of components in queue or in service at time } t \end{aligned}$$

We begin by describing the demand arrival process. We allow the system to have K different product order arrival streams. We wish to use these to define the component i demand process, $A_i(t)$, which is the cumulative total number of demands for component i to arrive in $[0,t]$. Let $D_k(t)$ denote the cumulative number of product orders to arrive from stream k in $[0,t]$, for $k=1,\ldots,K$. Our only requirement is that D obeys a functional Central Limit Theorem. That is, we require that there exists a c-dimensional vector $\alpha=(\alpha_1,\ldots,\alpha_c)^T$ and a $c\times c$ covariance matrix $\hat{K}$, such that

$$\frac{1}{n}\left(D(n^2\cdot)-n^2\alpha\cdot\right)\Longrightarrow D^*(\cdot)\sim BM(0,\hat{K})\quad\text{as } n\to\infty$$

Let $\{u_k(\ell),\ell=1,2,\ldots\}$ be a sequence of independent, identically distributed random variables with mean $1/\alpha_k$ and squared coefficient of variation c_k^2. Set

$$D_k(t)=max\left\{\ell\left|\sum_{j=1}^{\lfloor t\rfloor}u_k(j)\le t\right.\right\}\tag{4.3}$$

Defined in this manner, it follows from the Functional Central Limit Theorem for Renewal Processes (see Billingsley [3]) that α_k is the long-run average external arrival rate of product orders from stream k, and $\hat{K}$ is the asymptotic covariance matrix of the arrival streams. That is, if we define

$$\hat{K}=diag(\alpha_1^3c_1^2,\ldots,\alpha_K^3c_K^2),\tag{4.4}$$

then we have

$$\mathbf{E}[D_k(t)]\sim\alpha_kt\text{ and }\mathbf{Cov}[D(t)]\sim\hat{K}t\quad\text{as}\quad t\to\infty\tag{4.5}$$

where c_k^2 is the squared coefficient of variation of the k^{th} arrival stream.

Each arrival stream has $\ell=1,\ldots,L(k)$ possible product orders, where the probability that a product order from stream k is type ℓ is denoted $p_{k\ell}$. We denote the vector of probabilities for stream k by

$$p_k=(p_{k1},p_{k2},\ldots,p_{kL(k)})^T\tag{4.6}$$

Product type ℓ from stream k is denoted

$$q_\ell^k=(q_{1\ell}^k,q_{2\ell}^k,\ldots,q_{c\ell}^k)^T\tag{4.7}$$

Denote the $c\times L(k)$ product constituency matrix for stream k by Q_k, where

$$Q_k=\begin{pmatrix}q_{11}^k & q_{12}^k & \cdots & q_{1L(k)}^k\\ q_{21}^k & q_{22}^k & \cdots & q_{2L(k)}^k\\ \vdots & \vdots & \ddots & \vdots\\ q_{c1}^k & q_{c2}^k & \cdots & q_{cL(k)}^k\end{pmatrix}=\begin{pmatrix}| & | & | & |\\ q_1^k & q_2^k & \cdots & q_{L(k)}^k\\ | & | & | & |\end{pmatrix}\tag{4.8}$$

For demand stream k, let $\{\chi_k(j), j = 1, 2, \ldots\}$ be i.i.d. random vectors of dimension $L(k)$. Let e_ℓ be the $L(k)$-dimensional unit vector of zeros with a one in the ℓ^{th} position, and zeros elsewhere. The vector e_ℓ represents the arrival of product order type ℓ from stream k. That is, $Pr\left[\chi_k(j) = e_\ell\right] = p_{k\ell}$. It follows that

$$\mathbf{E}[\chi_k(j)] = p_k \quad \text{and} \quad \mathbf{Cov}[\chi_k(j)] = \Sigma^k,$$

where Σ^k is the $L(k) \times L(k)$ matrix with

$$\Sigma^k_{ij} = \mathbf{E}[(\chi_k(i) - \mathbf{E}[\chi_k(i)])(\chi_k(j) - \mathbf{E}[\chi_k(j)])] = \begin{cases} p_{ki}(1 - p_{ki}), & \text{if } i = j \\ -p_{ki}p_{kj}, & \text{if } i \neq j \end{cases} \tag{4.9}$$

Let X denote the following process:

$$X_k(t) = \sum_{j=1}^{\lfloor t \rfloor} \chi_k(j), \tag{4.10}$$

for $k = 1, \ldots, K$. Then we can write the component demand process A in terms of X by

$$A(t) = Q_1 X_1(D_1(t)) + \ldots + Q_K X_K(D_K(t)). \tag{4.11}$$

For each k, $X_k(D_k(t))$ is a random sum of random vectors, and its covariance matrix is given by

$$\mathbf{Cov}[X_k(D_k(t))] = \mathbf{E}[D_k(t)]\Sigma^k + \mathbf{Var}[D_k(t)]\theta_k\theta_k^T, \tag{4.12}$$

where Σ^k is as given above, and $\theta_k\theta_k^T$ is the $L(k) \times L(k)$ matrix with

$$[\theta_k\theta_k^T]_{ij} = p_{ki}p_{kj}. \tag{4.13}$$

Define the $L(k) \times L(k)$ matrix Ω_k by:

$$[\Omega_k]_{ij} = \begin{cases} \alpha_k p_{ki}(1 - p_{ki}) + \alpha_k^2 c_k^2 p_{ki}p_{ki}, & \text{if } i = j \\ -\alpha_k p_{ki}p_{kj} + \alpha_k^2 c_k^2 p_{ki}p_{kj}, & \text{if } i \neq j \end{cases} \tag{4.14}$$

Then it follows that $\mathbf{Cov}[X_k(D_k(t))] \sim \Omega_k t$, as $t \to \infty$. Define the $c \times c$ matrix H_k as follows:

$$\hat{H}_k \equiv Q_k \Omega_k Q_k^T. \tag{4.15}$$

Then

$$\mathbf{Cov}[Q_k X_k(D_k(t))] \sim \hat{H}_k t. \tag{4.16}$$

Let r_{ik} denote the expected type i component order batch size from stream k.

That is,

$$r_{ik} = \sum_{j=1}^{L_k} p_{kj} q_{ji}^k \tag{4.17}$$

Then the c-dimensional vector of expected component batch sizes from stream k, is denoted r_k, and given by

$$r_k = Q_k p_k. \tag{4.18}$$

Then from (4.5) and (4.18) it follows that $\mathbf{E}[Q_k X_k(D_k(t))] \sim \alpha_k r_k t$, as $t \to \infty$ We can write the $c \times K$ matrix of expected batch sizes as

$$R = \begin{pmatrix} | & | & | & | \\ r^1 & r^2 & \cdots & r^K \\ | & | & | & | \end{pmatrix} \tag{4.19}$$

Then we have

$$\mathbf{E}[A(t)] \sim R\alpha t \quad \text{as} \quad t \to \infty \tag{4.20}$$
$$\tag{4.21}$$

Next, we define the service vector process, S. Let $\{v_i(\ell), \ell = 1, 2, \ldots\}$ be a sequence of i.i.d. service times for component type i. Assume, further, that

$$\mathbf{E}[v_i(1)] = m_i \tag{4.22}$$
$$\mathbf{SCV}[v_i(1)] = s_i^2 \tag{4.23}$$

Define the process S,

$$S_i(n) = \sum_{j=1}^{\lfloor t \rfloor} v_i(j), \tag{4.24}$$

the c-dimensional vector m

$$m = (m_1, m_2, \ldots, m_c)^T, \tag{4.25}$$

and the matrix $\hat{J}$

$$\hat{J} = diag(m_1 s_1^2, m_2 s_2^2, \ldots, m_c s_c^2). \tag{4.26}$$

Then we have

$$\frac{1}{n}\Big(S(n^2\cdot) - n^2 m\cdot\Big) \Longrightarrow S^*(\cdot) \sim BM(0, \hat{J}) \quad \text{as } n \to \infty.$$

In light of the definitions above, we have $S_i(A_i(t))$ is the sum of the service times of all orders for component i to arrive in $[0, t]$.

Next, define the d-dimensional vector process $L(t)$ to be the *load* sent to the workstations in $[0, t]$. That is, $L(t)$ represents the cumulative total amount of work, as represented by the processing times required for component orders that have arrived in $[0, t]$. Written in terms of the component demand and service processes, this is

$$L(t) = CS(A(t)) \tag{4.27}$$

Next, we define the *immediate workload process*, denoted $W(t)$, to be the amount of work Now, if $L(t)$ represents the cumulative total amount of work sent to the workstations in $[0, t]$, then $W(t)$, the immediate workload process at time t is represented as

$$W(t) = [L(t) - t] + Y(t), \tag{4.28}$$

where

$$Y_j(t) = - \inf_{0 \leq s \leq t} \{L_j(s) - s\} \tag{4.29}$$

It can be shown that

$$\begin{aligned}
&W_j(t) \text{ is the amount of work at workcenter } j \text{ at time } t \\
&Y_j(\cdot) \text{ is continuous and non-decreasing with } Y_j(0) = 0 \\
&Y_j(\cdot) \text{ increases only at times } t \text{ when } W_j(t) = 0
\end{aligned}$$

Define $\tau_j(t)$ to be the time when all the component orders to have arrived at workstation j by time t will have completed processing. Then this can be written as

$$\tau_j(t) \equiv W_j(t) + t. \tag{4.30}$$

Next, define $V_i(t)$ to be the total number of type i component order to have complete service — at workstation $s(i)$ — by time t. Then

$$V_i(\tau_{s(i)}(t)) = A_i(t) \tag{4.31}$$

Next, let $N_i(t)$ denote the number of type i components at workstation $s(i)$ at time t, including the component in service, if any. Then

$$N(t) = A(t) - V(t) \tag{4.32}$$

Last, define the vector s to be the c-dimensional vector of component base-stock levels:

$$s = (s_1, s_2, \ldots, s_c)^T \tag{4.33}$$

Let $Z_i(t)$ denote the level of component i inventory held at time t, where a negative value denotes that the component is in backorder. Then

$$Z(t) = s - N(t). \tag{4.34}$$

4.3.2 System Parameters

Define the vector

$$\lambda = (\lambda_1, \ldots, \lambda_c)^T, \tag{4.35}$$

and set

$$\lambda = R\alpha, \tag{4.36}$$

where R is the expected component order batch size, give in Equation (4.19), and α is the long-run rate of demand of the demand streams. Then Then λ_i is the long-run average total arrival rate of component i orders to the workstation where it is processed.

Define

$$\rho_j = \sum_{\{i:i\in C(j)\}} \lambda_i m_i \tag{4.37}$$

The value ρ_j represents the traffic intensity of workcenter j; it represents the long-run average proportion of time that the workcenter is expected to be busy. If we define

$$\theta_j = 1 - \rho_j, \tag{4.38}$$

then θ_j represents the long-run average proportion of time that the workcenter is expected to be idle.

Let M be the matrix $M = diag(m)$. Then the utilization vector ρ can be written

$$\rho = CM\lambda \tag{4.39}$$

Last, if we denote the c-dimensional vector of 1's by e; that is,

$$e = (1, 1, \ldots, 1)^T, \tag{4.40}$$

then we can write

$$\theta = \rho - e \tag{4.41}$$

Above, we showed that

$$\mathbf{E}[A(t)] \sim \lambda t \quad \text{as} \quad t \to \infty \tag{4.42}$$

$$\mathbf{Cov}[A(t)] \sim \Gamma t \quad \text{as} \quad t \to \infty \tag{4.43}$$

It follows from the functional Strong Law of Large Numbers (SLLN) that

$$\mathbf{E}[L(t)] \sim \rho t \quad \text{as} \quad t \to \infty \tag{4.44}$$
$$\mathbf{E}[V(t)] \sim \lambda t \quad \text{as} \quad t \to \infty \tag{4.45}$$

4.3.3 Centering the Processes

chap:multi:subsec:centeringtheprocesses

Define the following centered processes.

$$\begin{array}{cc} \hat{A}(t) = A(t) - \lambda t & \hat{S}(t) = S(t) - m\lfloor t \rfloor \\ \hat{V}(t) = V(t) - \lambda t & \hat{D}(t) = D(t) - \alpha t \\ \hat{L}(t) = L(t) - \rho t & \hat{\tau}(t) = \tau(t) - et \end{array}$$

Also, define the following $c \times J$ matrix Λ by

$$\Lambda_{ij} = \begin{cases} \lambda_i & \text{if } i \in \mathcal{C}(j) \\ 0 & \text{otherwise.} \end{cases} \tag{4.46}$$

From these identities, straightforward algebraic manipulation gives

$$\hat{A}(t) = \hat{U}(t) + R\hat{D}(t) \tag{4.47}$$
$$\hat{U}(t) = Q_1\hat{X}_1(D_1(t)) + \ldots + Q_K\hat{X}_K(D_K(t)) \tag{4.48}$$
$$\hat{L}(t) = C[\hat{S}(A(t)) + M\hat{A}(t)] \tag{4.49}$$
$$W(t) = \hat{L}(t) - \theta t + Y(t) \tag{4.50}$$
$$N(t) = \hat{A}(t) - \hat{V}(t) \tag{4.51}$$
$$\hat{V}(\tau(t)) + \Lambda\hat{\tau}(t) = \hat{A}(t) \tag{4.52}$$
$$\hat{\tau}(t) = W(t) \tag{4.53}$$
$$Z(t) = s - N(t) \tag{4.54}$$

Furthermore, the following properties of the system processes are true

$$W(t) \geq 0 \tag{4.55}$$
$$\hat{Y}(\cdot) \text{ is continuous and non-decreasing with } \hat{Y}(0) = 0 \tag{4.56}$$
$$\hat{Y}_j(t) \text{ increases only at times } t \text{ for which } W_j(t) = 0 \tag{4.57}$$

4.3.4 Scaling the Processes

We now scale the processes in Equations (4.47)-(4.54) in the usual manner,

$$\tilde{W}^n(t) \equiv \frac{1}{n}W(n^2t) \quad \text{and} \quad \overline{D}^n(t) \equiv \frac{1}{n^2}D(n^2t)$$

Also, make the following definitions:

$$\begin{aligned} \tilde{\theta}^n &\equiv n\theta & (4.58) \\ \tilde{s}^n &\equiv ns. & (4.59) \end{aligned}$$

By applying the scalings given in (4.58) to the system processes defined in Equations (4.47)-(4.54), and using the definitions in (4.58) and (4.59), the scaled system is represented by the following equations:

$$\begin{aligned}
&\tilde{A}^n(t) = \tilde{U}^n(t) + R\tilde{D}^n(t) && (4.60)\\
&\tilde{U}^n(t) = Q_1\tilde{X}_1^n(\overline{D}_1^n(t)) + \ldots + Q_K\tilde{X}_K^n(\overline{D}_K^n(t)) && (4.61)\\
&\tilde{L}^n(t) = C[\tilde{S}^n(\overline{A}^n(t)) + M\tilde{A}^n(t)] && (4.62)\\
&\tilde{W}^n(t) = \tilde{L}^n(t) - \tilde{\theta}^n t + \tilde{Y}^n(t) && (4.63)\\
&\tilde{N}^n(t) = \tilde{A}^n(t) - \tilde{V}^n(t) && (4.64)\\
&\tilde{V}^n(\overline{\tau}^n(t)) + \Lambda\tilde{\tau}^n(t) = \tilde{A}^n(t) && (4.65)\\
&\tilde{\tau}^n(t) = \tilde{W}^n(t) && (4.66)\\
&\tilde{Z}^n(t) = \tilde{s}^n - \tilde{N}^n(t) && (4.67)\\
&\tilde{W}^n(t) \geq 0 && (4.68)\\
&\tilde{Y}^n(\cdot) \text{ is continuous and non-decreasing with } \tilde{Y}^n(0) = 0 && (4.69)\\
&\tilde{Y}_j^n(t) \text{ increases only at times } t \text{ for which } \tilde{W}_j^n(t) = 0 && (4.70)
\end{aligned}$$

4.3.5 The Scaling Convention

For a fixed value of n, we refer to the system whose physics are described in (4.60) to (4.70) as the n^{th} *scaled network*. The idea is then to let n grow without bound to obtain a sequence of converging systems whose limit is a system which is tractable. In order to do so, we must make the notion of the index precise. In this case, n be thought of in the following manner. Define

$$\epsilon \equiv \max_{1\leq j\leq d}(1-\rho_j) \tag{4.71}$$

Then set

$$n \equiv \sqrt{\frac{1}{\epsilon}} \tag{4.72}$$

Observe that ϵ is defined as the proportion of time the *least* utilized station is idle. That is, the workcenter that is the most under capacitated indexes the system. With this scaling convention, letting $n \to \infty$ is equivalent to letting the system utilization of *all* the workstations approach unity. That is, our limit regime is one in which all the workstations are busy 100% of the time. We call this the *heavy-traffic limit.*

In order to avoid simplistic, or degenerate limiting behavior, we need to make two specific assumptions. In the first, we wish to classify the rate at which the sequence of networks converge to the heavy-traffic limit. Specifically, we assume the following:

$$n(e - \rho) \to \theta^*, \ 0 < \theta^* < \infty \quad \text{as} \quad n \to \infty. \tag{4.73}$$

Roughly speaking, (4.73) says that the workstation utilization rates do not approach unity too quickly.

Our second assumption is that as the workstations become ever more busy, (implying a degradation of service) the component base-stock levels will, in turn, increase (implying higher service levels). Specifically, we require that

$$\frac{s_i^n}{n} \to \beta_i^*, \ 0 < \beta_i^* < \infty \quad \text{as} \quad n \to \infty, \ i = 1, \ldots, c. \tag{4.74}$$

4.3.6 The Limiting System

Under the scaling convention given in (4.58), and assumptions (4.73) and (4.74), the from the functional SLLN

Lemma 1 *As $n \to \infty$,*

$$\overline{D}_k^n(t) \to \alpha_k t, \ k = 1, \ldots, K, \tag{4.75}$$
$$\overline{A}^n(t) \to \lambda t \tag{4.76}$$
$$\overline{\tau}^n(t) \to t, \tag{4.77}$$
$$\tag{4.78}$$

where the convergence is over compact sets in the uniform norm. With this result, it can be shown that, as $n \to \infty$, and in light of (4.73) and (4.74), the $n^t h$ system, as defined by (4.60) to (4.70), converges in distribution to a limiting system defined by the following:

$$A^*(t) = U^*(t) + RD^*(t) \tag{4.79}$$

$$U^*(t) = Q_1 X_1^*(\alpha_1 t) + \ldots + Q_K X_K^*(\alpha_K t) \tag{4.80}$$
$$L^*(t) = C[S^*(\lambda t) + MA^*(t)] \tag{4.81}$$
$$W^*(t) = L^*(t) - \theta^* t + Y^*(t) \tag{4.82}$$
$$N^*(t) = A^*(t) - V^*(t) \tag{4.83}$$
$$V^*(t) + \Lambda\tau^*(t) = A^*(t) \tag{4.84}$$
$$\tau^*(t) = W^*(t) \tag{4.85}$$
$$Z^*(t) = s^* - N^*(t) \tag{4.86}$$
$$W^*(t) \geq 0 \tag{4.87}$$
$$Y^*(\cdot) \text{ is continuous and non-decreasing with } Y^*(0) = 0 \tag{4.88}$$
$$Y_j^*(t) \text{ increases only at times } t \text{ for which } W_j^*(t) = 0 \tag{4.89}$$

In addition, if we define the matrix $\hat{B}$ by

$$\hat{B} \equiv diag(\lambda_1 m_1^2 s_1^2, \ldots, \lambda_c m_c^2 s_c^2), \tag{4.90}$$

and the matrix $\hat{H}$ by

$$\hat{H} \equiv \sum_{k=1}^{K} \alpha_k \hat{H}_k, \tag{4.91}$$

then the following are also true.

$$S^*(\lambda t) \text{ is a driftless Brownian motion with covariance matrix } \hat{B} \tag{4.92}$$
$$D^*(t) \text{ is a driftless Brownian motion with covariance matrix } \hat{K} \tag{4.93}$$
$$U^*(t) \text{ is a driftless Brownian motion with covariance matrix } \hat{H} \tag{4.94}$$
$$S^*,\ D^*, \text{ and } U^* \text{ are independent} \tag{4.95}$$

We can simplify (4.79)-(4.89) substantially by making the following manipulations. First, combine (4.85), (4.84), and (4.83) to attain

$$N^*(t) = \Lambda W^*(t).$$

Next, rewrite (4.81) by substituting (4.80) into (4.79), and then substituting the resulting expression into (4.81). Finally, rewrite (4.82) by setting

$$\xi^*(t) = C[S^*(\lambda t) + M(Q_1 X_1^*(\alpha_1 t) + \ldots + Q_K X_K^*(\alpha_K t) + RD^*(t))] - \theta^* t.$$

It follows from (4.92), (4.93), (4.94), and (4.95) that ξ^* is a d-dimensional Brownian motion with drift vector $-\theta^*$, and covariance matrix

$$\Gamma \equiv C\hat{B}C^T + CM\hat{H}(CM)^T + CMR\hat{K}(CMR)^T. \tag{4.96}$$

The final result is the following theorem.

Theorem 4 *As $n \to \infty$, under the scaling convention given in (4.58) and assumptions (4.73) and (4.74), the scaled n^{th} network defined by (4.60)-(4.70) converges weakly to the system $(W^*, Y^*, \xi^*, N^*, Z^*)$, defined by*

$$W^*(t) = \xi^*(t) + Y^*(t) \tag{4.97}$$

$$\xi^*(t) \textit{ is a d-dimensional } BM(-\theta^*, \Gamma) \tag{4.98}$$

$$Y^*(\cdot) \textit{ is continuous and non-decreasing with } Y^*(0) = 0 \tag{4.99}$$

$$Y_j^*(t) \textit{ increases only at times } t \textit{ for which } W_j^*(t) = 0 \tag{4.100}$$

$$N^*(t) = \Lambda W^*(t) \tag{4.101}$$

$$Z^*(t) = \beta^* - N^*(t) \tag{4.102}$$

The rigorous proof of this theorem is an extension of that found in Peterson [48], and the details are omitted.

4.4 The Limiting Process

An important question to ask is what sort of process do (4.97)- (4.102) define, if any. The following definition is adapted from Williams [61]

Definition 1 (Semimartingale Reflecting Brownian Motion in the Orthant) *Let d be a positive integer, S_+^d represent the d-dimensional orthant, θ^* a d-dimensional constant vector, Γ a $d \times d$ non-degenerate covariance matrix, and R be a $d \times d$ matrix. Then an SRBM associated with the data $(S_+^d, \theta^*, \Gamma, R)$, written $SRBM(S_+^d, \theta^*, \Gamma, R)$ is a d-dimensional process W such that*

$$\begin{aligned}
&(a) \quad Z(t) = X(t) + RY(t) \in S_+^d, \textit{ for all } t \geq 0 \\
&(b) \quad X \textit{ is a d-dimensional } BM(\theta, \Gamma) \\
&(c) \quad Y \textit{ is continuous and non-decreasing with } Y(0) = 0 \\
&(d) \quad \int_0^t 1_{F_j}(Z(s))dY_j(s) = Y_j(t), \textit{ for all } t \geq 0,
\end{aligned}$$

where F_j is the j^{th} face; i.e. $F_j \equiv \{x \in S_+^d : x_j = 0\}$. The matrix R is known as the Reflection matrix.

Our process satisfies the requirements of the definition, and we can identify the limiting process as an SRBM on the positive orthant with the identity as the reflection matrix.

Our goal is to see whether the lead time-inventory trade off derived in Chapter 2 holds in the case where we allow for multiple workcenters. In order to do so, we would like to investigate the distribution function of the diffusion process that approximates our system in the limiting procedure we have followed in this chapter. Unfortunately, the density function for reflected Brownian motion is is only explicitly known a few cases. In particular, we would be interested to know when the density function of the SRBM might have a product form solution (that is, a density function that is just the product of the marginal densities). Harrison and Williams [29] provided a criterion known as the *Skew Symmetry Condition* for multiclass feedforward networks, of which ours is a special case. For our purposes, their Theorem 3.1 of [29] can be stated as follows.

Theorem 5 *The SRBM W has a product form stationary distribution if and only if*

$$\Gamma_{ij} = \frac{1}{2}\Gamma_{jj} \qquad \textit{for } 1 \le j < i \le d.$$

Investigation of the covariance matrix Γ, defined in (4.96) shows that a necessary and sufficient condition for the density function to have a product form solution is that there is no product order that requires more than one component. This case is, however, not of interest to us.

There are only a few cases in which closed-form solutions for density functions of SRBMs that are not of product form are known. Harrison [31] was able to calculate it for a two-dimensional SRBM corresponding to a special tandem queue. Varma [56] was able to find an explicit form of the density function in a special case of our system. He considered a two-component, two-workcenter production-inventory system where a single arrival stream brought one product order that requested one unit of Component One and one unit of Component Two. Under the assumption that the variance of the arrival process and the variances in the production processes were the same, and the mean processing times of both components identical, he was able to solve the partial differential equations derived in [27]. His solution does not, however, admit to a closed-form cumulative distribution function, making the sort of direct analysis done in Section 2.4.7 unavailable. While we do not know the form of the joint density function of our SRBM, we do know something about the marginal densities. We will see in Section 4.4.1 that they are exponential.

4.4.1 The Basic Adjoint Relationship

Define the differential operators L and D_i, as applied to the function f, in the following manner

$$Lf = \frac{1}{2}\sum_{i=1}^{d}\sum_{j=1}^{d}\Gamma_{ij}\frac{\partial^2 f}{\partial x_i \partial x_j} + \sum_{i=1}^{d}\theta_i^*\frac{\partial f}{\partial x_i} \tag{4.103}$$

and

$$D_j f = v_j \cdot \bigtriangledown f, \text{ for } j = 1, \ldots, d. \tag{4.104}$$

Suppose the process Z, as defined in definition 1, has a stationary distribution π. Then the stationary distribution π is unique, and has a density function p_0 relative to Lebesgue measure on S_+^d. Furthermore, for $j = 1, \ldots, d$, the exists a finite Borel measure ν_j on face F_j that has a $d-1$-dimensional density function relative to surface measure σ_j on F_j. Let $C_b^2(S_+^d)$ denote the space of continuous, bounded, real-valued functions defined on S_+^d that have bounded continuous first and second partial derivatives on S_+^d, and suppose that the process Z satisfies definition 1. Then a necessary (see Dai [9]) and sufficient (see Dai and Kurtz [10]) for a function $p = (p_0; p_1, p_2, \ldots, p_j)$ to be the density function of the SRBM that it satisfy, for all test functions $f \in C_b^2(S_+^d)$ — that is, all bounded, twice-continuously differentiable functions defined on the positive orthant — the following *basic adjoint relationship* (BAR):

$$\int_{S_+^d} Lf \; p_o dx + \sum_{j=1}^{d}\int_{F_j} D_j f \; p_j d\sigma_j = 0, \text{ for all } f \in C_b^2(S_+^d). \tag{4.105}$$

Since the reflection matrix of our diffusion is just the identity matrix, this becomes

$$\int_{S_+^d} Lf \; p_o dx + \sum_{j=1}^{d}\int_{F_j} \frac{\partial f}{\partial x_j} \; p_j d\sigma_j = 0, \text{ for all } f \in C_b^2(S_+^d). \tag{4.106}$$

If we assume that the moment generating function of the density function is finite in a neighborhood of the origin, then we can relax the boundedness assumption, and extend the class of functions for which the BAR holds to polynomials. Then by taking the function $f(x) = e^{-\gamma_i x_i}$, $i = 1, \ldots, d$, we can show that the Laplace transform of the i^{th} marginal density function is exponentially distributed in the parameter $2\theta/\Gamma_{ii}$.

4.5 The Lead time-Inventory Trade Off

In Section 4.4.1, we were able to show that the marginal densities of the multi-dimensional diffusion that is the limiting process as our scaling parameter approaches infinity are exponentially distributed. Following the argument of Section 2.4.7, we can conclude that the j^{th} component of the limiting process W^* is distributed as an

RBM with drift θ_j,variance Γ_{jj} and state-space $[0, \infty)$. By unscaling the process, it is possible to verify that $nW_j^*(n^{-2}\cdot)$ has an exponential distribution with parameter

$$\tau_j \equiv \frac{2(1-\rho_j)}{\Gamma_{jj}}. \tag{4.107}$$

If we let $R_j(t)$ denote the response time at the j^{th} workstation at time t, then we can follow the derivation exactly as done in Chapter 2 to write

$$R_j(t) \approx \left(nW_j^*(n^{-2}t) - \min_{i \in \mathcal{C}(j)} \left\{\frac{s_i}{\lambda_i}\right\}\right)^+ .$$

Observe that there is one fundamental difference between the expression above, and that analogous expression in Section 2.4.7: there, our expression for R_k related to the k^{th} product order, while here we are referring to the response time at workcenter j, at time t.

If we let $R_j(\infty)$ denote the steady state response time at the j^{th} workstation, and let T denote the random variable whose distribution is exponential with parameter τ_j, then as done in Section 2.4.7, we are lead to the following approximation:

$$Pr\{R_j(\infty) > x\} \quad \approx \quad e^{-\tau_j\left(x + \min_{i\in\mathcal{C}(j)} s_i/\lambda_i\right)}. \tag{4.108}$$

Proceeding in Section 2.4.9, if we let $\mathcal{I}_j = \{\ell : \frac{s_\ell}{\lambda_\ell} = \min_{i\in\mathcal{C}(j)} \frac{s_i}{\lambda_i}\}$ be the set of "critical" components for workcenter j, then when promised lead time for a product that has components requiring processing at workcenter j is changed, the base-stock level of critical components should be changed according to the rule

$$\Delta s_i = -\lambda_i \Delta x, \qquad i \in \mathcal{I}_j. \tag{4.109}$$

With the relationships in (4.108) and (4.109) in mind, we can now present our extension of the linear trade off rule discussed in Chapter 4. If we restrict ourselves to the case that every product order requires at least one unit of every component, then every product order has the same set of critical components. We then hypothesize that if we increase the base-stock levels of the critical components at *all* workcenters in heavy-traffic by one unit, and reduce the promised delivery lead times of all *product streams* according to the rule in (4.109), then the fill rate will remain constant over all product streams. We expect this to be true because we are raising (lowering) the base-stock levels (promised delivery lead times) in tandem at all workcenters where (4.109) holds. We then know that the marginal probability that the promised delivery lead times is exceeded at any one of the workstations remains the same. It then seems reasonable to expect that the overall probability that the promised delivery lead time is exceeded does not change, either. While we are unable to justify such a claim rigorously, we present numerical evidence in Section 4.6 that this does hold.

4.6 Numerical Results

In Section 4.5, we hypothesized that in the special case where we require all products be composed of at least one unit of all components, then by applying the linear trade off (4.109) to the individual workcenters, we will preserve a constant product fill rate. In order to test this hypothesis we used discrete event simulation, and followed a procedure similar to that used in Chapter 3. We simulated five different systems, consisting of a mix of various different inter-arrival and service-time distributions, as well as a mix in the number of components, workcenters, and product arrival streams. We then simulated various combinations of promised delivery lead times and initial component base-stock levels according to our linear trade-off rule. To do so, we would begin with the case where the promised delivery lead time of one of the product arrival streams was zero, and determine the component base-stock levels that would give a certain product fill rate. Usually, we looked for a mix where orders from one product stream would be filled within the promised delivery lead time 95% of the time, while orders for another would be filled within a given promised delivery lead time within 90% of the time. The actual numbers vary with the system tested, and are essentially arbitrary. That is, we did not keep a specific real product or industry in mind when doing the numerical tests.

Once we established a baseline, we would then increase the promised delivery lead times by a constant increment, and decrease the initial component base-stock levels in accordance with the corresponding aggregate component order rate. For each pair of points generated, we would simulate the system for product fill rates. If the theory is correct, then these product fill rates should plot to a straight line. The results of these experiments are illustrated in Figures 4-2 through 4-6. In each system, a single point represents a full discrete-event simulation experiment. Each simulation experiment took approximately one hour to run on a dual-processor 400 Mhz Pentium II system running Linux. The event-based discrete-event simulator was custom-written in C to test these systems.

As we did for the single-workcenter case, each simulation experiment consisted of 30 different, independent runs of 1.3 million time units, with the data from the first 300,000 time units discarded as an initial transient. We used an approximate 95% confidence interval to ensure that the simulated product fill rates were exact to the second or third decimal place. As with all the simulation experiments in this dissertation, the length of the simulation runs was chosen so that we could be sure that the transient period did not affect the results and that the period of data collection was long enough to give confidence intervals that gave at least three significant digits of accuracy to the right of the decimal place. To do so, we followed a simple procedure of picking a simulation length and transient period at random, and the doubling it until the results from one iteration to the next were essentially identical.

4.6.1 System I

The first system we tested consisted of three components, the first two of which were processed at one workstation, and the third of which was processed at a second workstation. The processing times for each component at its workstation were exponentially distributed, but were different from one another so that the system in question falls outside the category of multiclass, product-form queueing networks discussed in [1] and [37]. Demand was represented by two independent arrival streams, each bringing one product order, and each with exponentially distributed inter-arrival times. The relevant demand and processing rates are summarized in the table in Figure 4-2.

The first product arrival stream brought a product demand that consisted of two Type One components, one Type Two component, and two Type Three components. The second product arrival stream brought a product demand that consisted of one Type One component, two Type Two components, and two Type Three components. It is straightforward to calculate the aggregate component-level demand rates at $\lambda_1 = 4.5$, $\lambda_2 = 6.0$, and $\lambda_3 = 7.0$. Using these, and the processing rates given in the table in Figure 4-2 leads to nominal workstation utilization rates of $\rho_1 = \rho_2 = 0.976$.

We set the initial component base-stock levels so that Components Two and Three were the bottleneck components at their respective workstations. We then computed the locus of points as predicted by the linear trade-off rule, and simulated the resulting system for product fill rates. The results are summarized in the graph in Figure 4-2. Since the arrival rates of the bottleneck components are integer, the base-stock levels produced by the trade-off do not need rounding. In this case, the linear trade-off rule holds almost exactly. The points for the product fill rates of orders from both demand streams plot to essentially-straight lines. Both exhibit a slight upward drift as the promised delivery lead time increases. As with the single workcenter case, we expect that this has to due with the ability to better handle variability in production times that may arise. In other words, when the promised delivery lead time is small, a disruptive event at one of the workcenters, such as a component order taking a particularly long time to process, will be felt more acutely, and a degradation in service results. As the promised delivery lead time increases, this effects of a single bad processing time are felt less, and the level of service increases slightly.

4.6.2 System II

The second system we tested consisted of four components which are processed at two workstations. Components one and two are processed at the first workstation, and components three and four are processed at the second workstation. The processing times at the workstations were exponentially distributed, and as in the case of System I, we set the processing rates to be different to ensure that the underlying queueing system does not have a product-form density function. Demand was represented

Product Stream	Inter-Demand Distribution	Demand Rate	Demand SCV
One	Exponential	1.00	1.00
Two	Exponential	2.50	1.00
Component	Production Distribution	Production Rate	Production SCV
1	Exponential	10.00	1.00
2	Exponential	11.40	1.00
3	Exponential	7.17	1.00

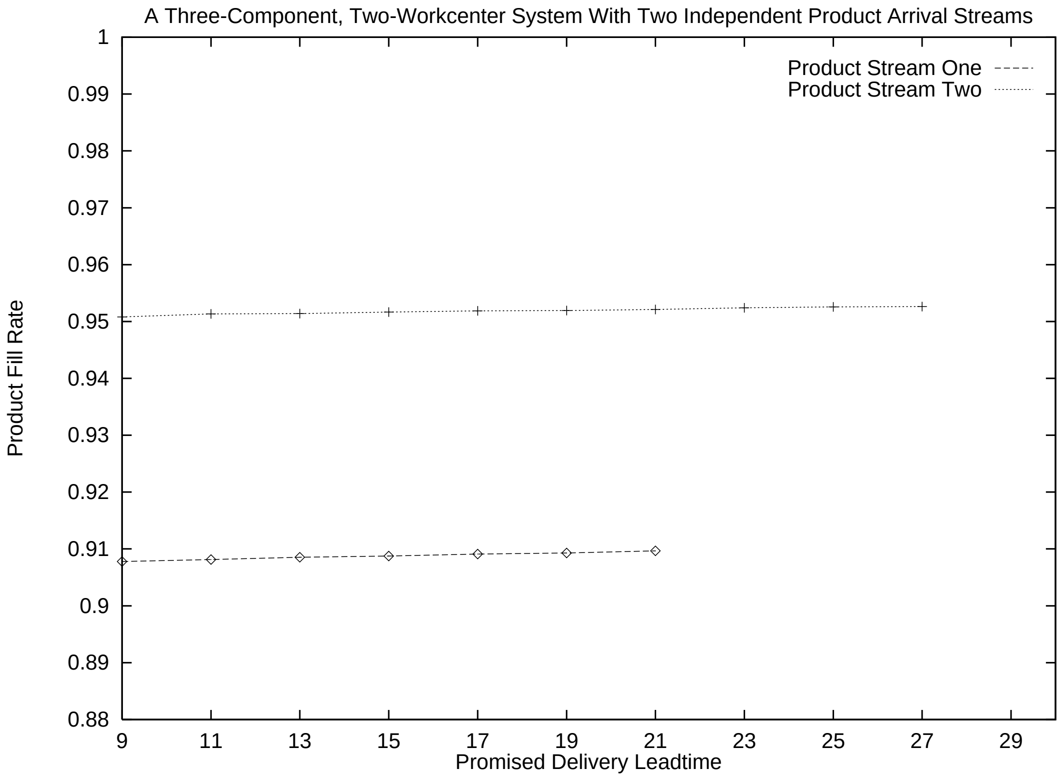

Figure 4-2: System I: Delivery Lead time/Production Rate Tradeoff for a system with exponentially distributed demand inter-arrival times and component production times, and workstation utilization rates $\rho_1 = \rho_2 = 0.976$.

by two different independent arrival streams, with gamma distributed inter-demand times. The relevant parameters are summarized in the table in Figure 4-3.

Product Stream	Product Order	Probability	Comp. One Demand	Comp. Two Demand	Comp. Three Demand	Comp. Four Demand
One	1	0.5	1	2	3	4
	2	0.5	2	2	1	2
Two	1	0.3	2	1	3	2
	2	0.7	4	2	1	2

Table I: System II: The product order mix for two independent demand streams with gamma-distributed inter-arrival times.

Each of the two demand streams bring with them two different possible product orders. At an instant of demand arriving from stream one, one-half of the time the component order mix is for one type-one component, two type-two components, three type-three components, and four type-four components. The other half of the time the component order mix is for two type-one components, two type-two components, one type-three component, and two type-four components. Then product mix for demand stream two is defined in an analogous manner, and both product order mixes are are summarized in Table I. From the given data, it is straightforward to calculate the aggregate component-level demand rates to be $\lambda_1 = 3.943$, $\lambda_2 = 3.638$, $\lambda_3 = 3.580$, and $\lambda_4 = 5.143$. It is then simple to calculate the nominal utilization rates at the workstations to be $\rho_1 = 0.968$ and $\rho_2 = 0.948$.

In order to generate the the points, we initially set the promised delivery lead time for product orders from demand stream one to be six, and the promised delivery lead time for product orders from demand stream two to be zero. We then set the initial component base-stock levels so that components one and four were the bottleneck components at their respective workstations. Then, using the linear trade-off rule, we increased the promised delivery lead times for both product streams in increments of one, and decreased the relevant initial component base-stock levels according to the linear trade-off rule. For this system, this entailed decreasing the base-stock level for component one by 3.943, and for component four by 5.143 for ever unit increase in the promised delivery lead times. Since these numbers are not integer, we had to round the resulting initial component base-stock levels to the nearest integer. We did not change the component two and three initial base-stock levels at all. For each unit increment in the promised delivery lead time, and corresponding base-stock distribution, we simulated for the product fill rates. The results are summarized in the graph in Figure 4-3. While not as sharp as the results in Subsection 4.6.1, the results are still quite strong. The product fill rates for both product arrival streams plot to essentially-straight lines. We expect that the small kinks observed are due to the rounding of the component base-stock levels. Notice that the the slight upward drift of product order fill rates, described in the previous section, also occurs here. As before, we expect that this is a result of the system being able to better weather disruptive events at a workcenter when the promised delivery lead time is longer.

4.6.3 System III

The third system we tested consisted of three components which are processed at three workstations. Component One is processed at the first workcenter, Component Two at the second, and Component Three at the third. The processing time for Component One is exponentially distributed, while those of Component Two and Component Three have an Erlang distribution. Demand is represented by two different independent arrival streams, with gamma distributed inter-demand times. The relevant parameters are summarized in the table in Figure 4-4. We allowed each arrival stream to have a random product order mix, summarized in Table II. Using values given in the table in Figure 4-4, and equation 2.2, we calculate the aggregate component-level demand rates as $\lambda_1 = 4.2$, $\lambda_2 = 4.63$, and $\lambda_3 = 4.78$. Using these, and the processing rates given in the table in Figure 4-4 leads to nominal workstation utilization rates of $\rho_1 = 0.966$, $\rho_2 = 0.973$, and $\rho_3 = 0.976$.

Since each component is processed at its own workcenter, each acts as the bottleneck component. We followed the standard generation procedure, and simulated the systems generated. The results are summarized in the graph in Figure 4-4. In general, the simulated product fill rates plot to straight lines, with some oscillation in the results. Since none of the arrival rates of component demand are integer, we had to round the base-stock levels to the nearest integer during the generation procedure. We believe that this rounding may account for some of the saw-toothed behavior of the lines. However, the results are still quite strong.

4.6.4 System IV

The fourth system we tested was very similar in structure to the system investigated in Section 4.6.3, but we made the change of having the processing times be less variable than the inter-arrival times. The fourth system consisted of three components which are processed at three workstations. Component One is processed at the first workcenter, Component Two at the second, and Component Three at the third. The processing time for all components is exponentially distributed. Demand is represented by two different independent arrival streams, with hyperexponentially distributed inter-demand times. The relevant parameters are summarized in the table in Figure 4-5. We allowed each arrival stream to have a random product order mix, summarized in Table III. Using values given in the table in Figure 4-5, we calculate

Product Stream	Inter-Demand Distribution	Demand Rate	Demand SCV
One	Gamma	1.33	0.667
Two	Gamma	0.57	0.286
Component	Production Distribution	Production Rate	Production SCV
1	Exponential	8.80	1.00
2	Exponential	7.00	1.00
3	Exponential	9.50	1.00
4	Exponential	9.00	1.00

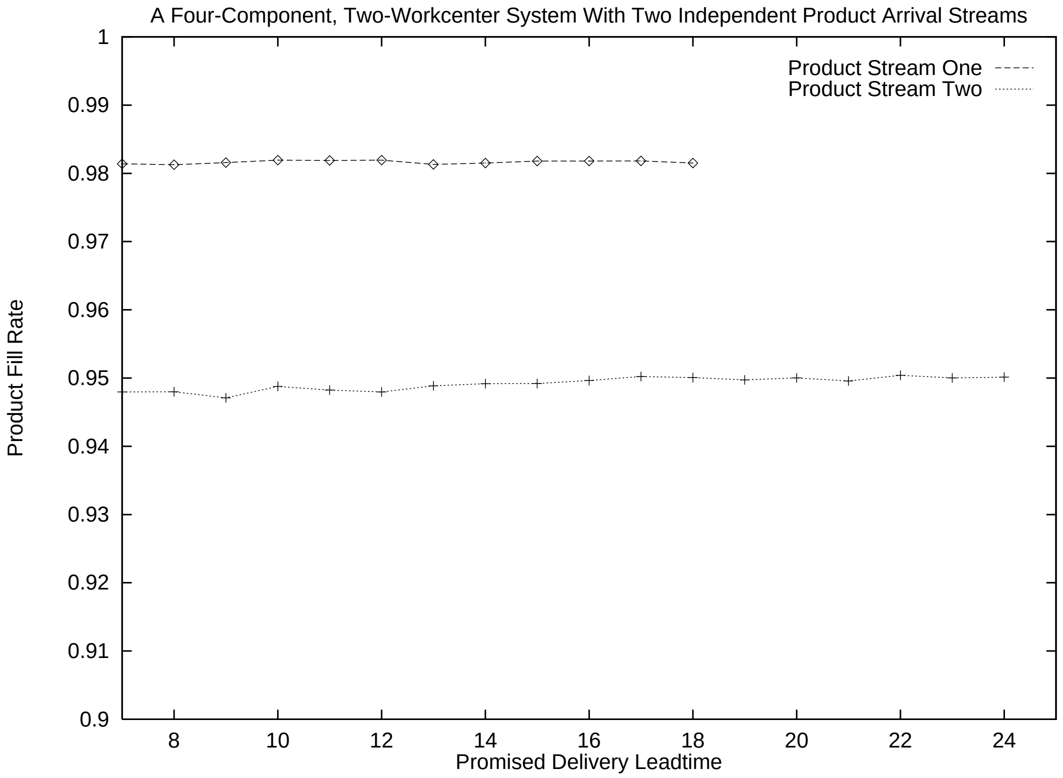

Figure 4-3: System II: Delivery Lead time/Production Rate Tradeoff for a system with gamma distributed demand inter-arrival times, exponentially distributed component production times, and workstation utilization rates $\rho_1 = 0.968$ and $\rho_2 = 0.948$.

Product Stream	Product Order	Probability	Comp. One Demand	Comp. Two Demand	Comp. Three Demand
One	1	0.3	1	2	3
	2	0.2	2	2	1
	3	0.5	3	3	4
Two	1	0.4	2	2	1
	2	0.4	4	3	2
	3	0.2	1	4	6

Table II: System III: The product order mix for two independent demand streams with gamma-distributed inter-arrival times.

the aggregate component-level demand rates as $\lambda_1 = 3.43$, $\lambda_2 = 3.46$, and $\lambda_3 = 3.07$. Using these, and the processing rates given in the table in Figure 4-5 leads to nominal workstation utilization rates of $\rho_1 = 0.954$, $\rho_2 = 0.937$, and $\rho_3 = 0.958$.

Since each component is processed at its own workcenter, each acts as the bottleneck component. We followed the standard generation procedure, and simulated the systems generated. The results are summarized in the graph in Figure 4-5. In general, the simulated product fill rates plot to straight lines, with a very small amount of oscillation in the results. Since none of the arrival rates of component demand are integer, we had to round the base-stock levels to the nearest integer during the generation procedure. Despite this rounding, the results are very strong.

Product Stream	Product Order	Probability	Comp. One Demand	Comp. Two Demand	Comp. Three Demand
One	1	0.5	3	2	3
	2	0.5	2	2	1
Two	1	0.4	2	2	1
	2	0.4	1	3	2
	3	0.2	1	1	2

Table III: System IV: The product mix for two independent demand streams with hyperexponential-distributed inter-arrival times.

4.6.5 System V

The last system we tested consists of a The fourth system consisted of three components which are processed at two workstations. Component One is processed at the first workcenter, while Component Two and Component Three are processed at the second. Component One processing times have a gamma distribution, Component Two processing times are exponentially distributed, and Component Three processing times are hyperexponentially distributed. Demand is represented by three different independent arrival streams, the first with hyperexponentially distributed inter-demand times, the second with gamma-distributed interarrival times, and the third with exponentially distributed inter-demand times. The relevant parameters are summarized in the table in Figure 4-6. We allowed each arrival stream to have a random product order mix, summarized in Table IV. Using values given in the table in Figure 4-6, and equation 2.2, we calculate the aggregate component-level demand rates as $\lambda_1 = 3.93$, $\lambda_2 = 5.90$, and $\lambda_3 = 4.06$. Using these, and the processing rates given in the table in Figure 4-6 leads to nominal workstation utilization rates of $\rho_1 = 0.944$, and $\rho_2 = 0.963$.

The only component processed at the first workcenter is Component One, so it is the bottleneck component there. We set the base-stock levels so that Component Two is the bottleneck at the second workcenter. Following the standard generation procedure, simulating the systems for product fill rate, and plotting the results lead to the points in Figure 4-5. Here the results are very strong. There is almost no saw-tooth behavior in the lines plotted, despite the mix of inter-demand and processing time distributions.

Product Stream	Product Order	Probability	Comp. One Demand	Comp. Two Demand	Comp. Three Demand
One	1	0.5	3	2	3
	2	0.5	2	2	1
Two	1	0.4	2	2	1
	2	0.6	1	3	2
Three	1	0.5	1	1	2
	2	0.5	2	3	1

Table IV: System V: The product mix for three independent demand streams with variously-distributed inter-arrival times.

4.7 Conclusions

In this chapter we extended the results derived for the case of a single capacitated resource described in Chapter 2 to the case where we allow different components to be processed at different workcenters, with the restriction that all products require at least one unit of all components. We were unable to present a complete, rigorous analysis as we did in Chapter 2 because of the intractability of the distribution function of the limiting process in multiple dimensions. We were able to show, however, that the marginal distributions of the limiting process were exponentially distributed, and that the linear trade off rule held at each workcenter. We hypothesized that it was enough to apply the linear trade off rule at each workcenter in isolation to preserve a constant product fill rate. We tested our hypothesis with detailed discrete event simulation, and found that it held even for very complicated systems.

Product Stream	Inter-Demand Distribution	Demand Rate	Demand SCV
One	Gamma	0.85	0.45
Two	Gamma	0.90	0.24
Component	Production Distribution	Production Rate	Production SCV
1	Exponential	4.35	1.00
2	2-Erlang	4.76	0.50
3	3-Erlang	4.90	0.33

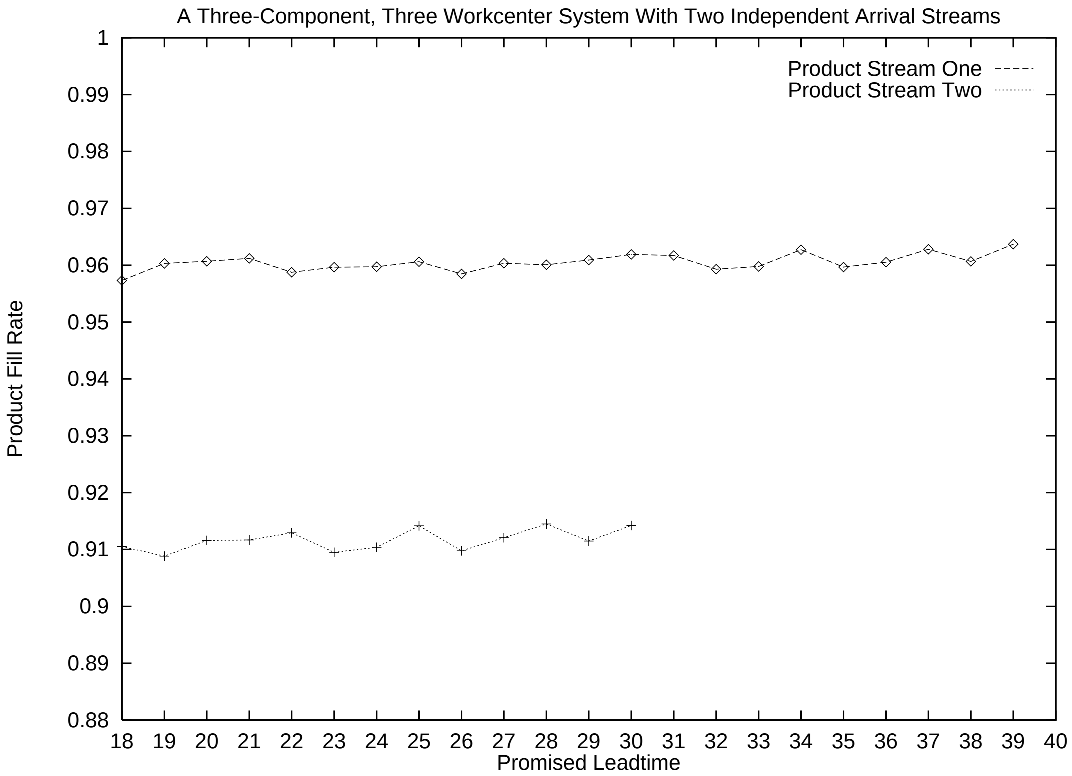

Figure 4-4: System III: Delivery Lead time/Production Rate Tradeoff for a system with gamma distributed demand inter-arrival times, Erlang distributed component production times, and workstation utilization rates $\rho_1 = 0.966$, $\rho_2 = 0.973$, and $\rho_3 = 0.976$.

Product Stream	Inter-Demand Distribution	Demand Rate	Demand SCV
One	Hyperexponential	1.00	4.00
Two	Hyperexponential	0.67	4.00
Component	Production Distribution	Production Rate	Production SCV
1	Exponential	3.60	1.00
2	Exponential	3.70	1.00
3	Exponential	3.20	1.00

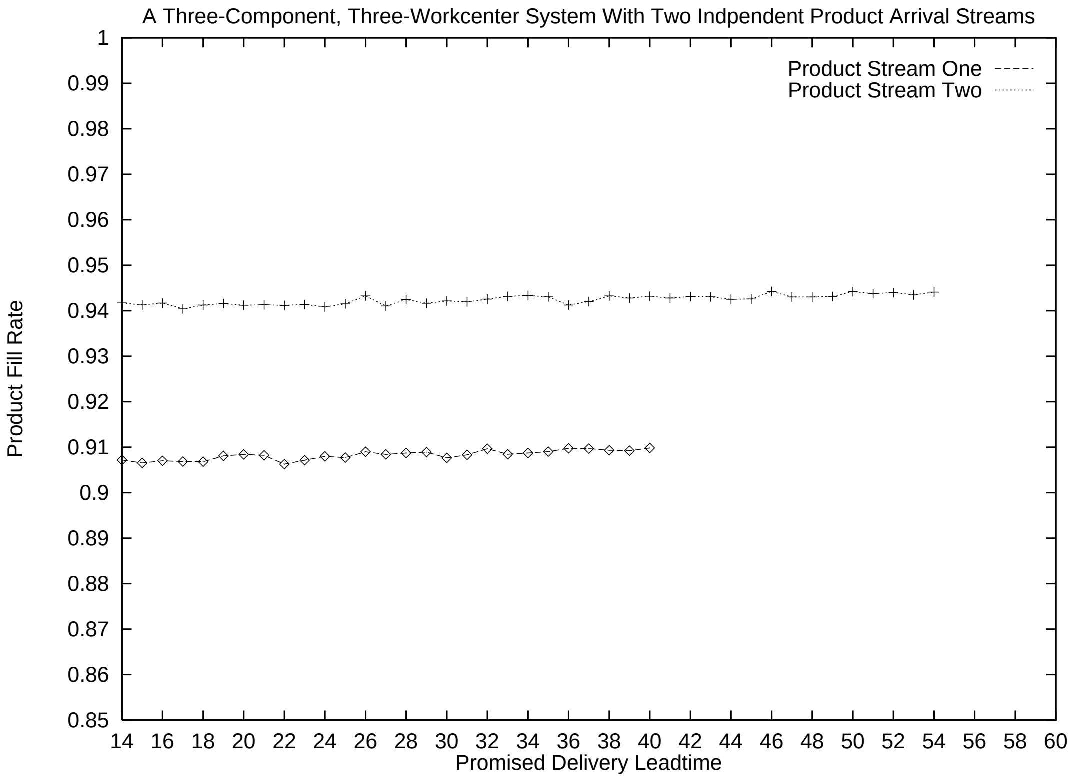

Figure 4-5: System IV: Delivery Lead time/Production Rate Tradeoff for a system with hyperexponential-distributed demand inter-arrival times, exponential distributed component production times, and workstation utilization rates $\rho_1 = 0.954$, $\rho_2 = 0.937$, and $\rho_3 = 0.958$.

Product Stream	Inter-Demand Distribution	Demand Rate	Demand SCV
One	Gamma	0.33	—-
Two	Exponential	1.50	1.00
Three	Hyperexponential	0.67	7.15
Component	Production Distribution	Production Rate	Production SCV
1	Hyperexponential	4.17	17.36
2	Gamma	8.00	——
3	Exponential	18.00	1.00

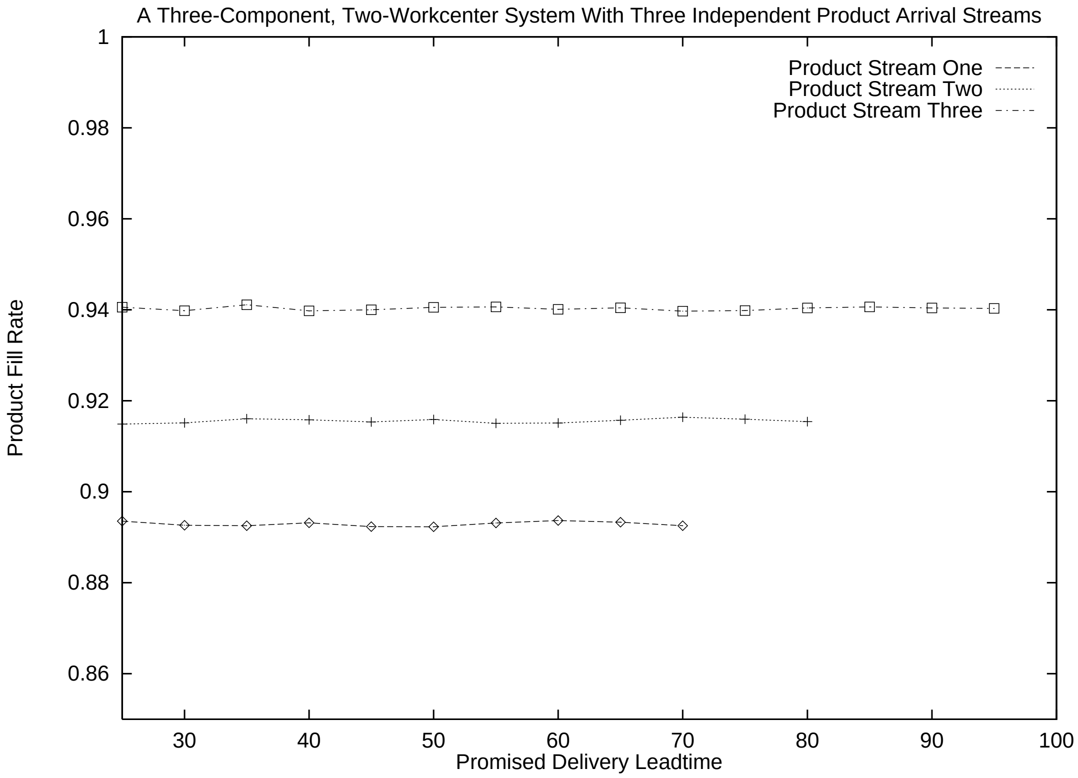

Figure 4-6: System V: Delivery Lead time/Production Rate Tradeoff for a system with variously-distributed demand inter-arrival times, variously-distributed component production times, and workstation utilization rates $\rho_1 = 0.944$ and $\rho_2 = 0.963$.

Chapter 5

Multiclass processing lines

5.1 Introduction

In this chapter we present a model and a preliminary analysis of a processing line with finite buffers, unreliable machines, and two different distinct component classes. While the methods presented appear to be generalizable to more than two part types, we restrict our attention to only two in order to make precise all of the issues involved in the decomposition.

The basic method described in Chapter 1 — the decomposition of a long line into analytically tractable components — has been widely applied to a large variety of manufacturing systems under differing assumptions. However, all of the decompositions done to date using the methods described in the preceding section have one thing in common: they assume there is only one part type. The contribution of this work will be to extend the method described above to a processing line with more than one part type. The importance of this extension to the body of literature already developed should be stressed. Since most manufacturing systems consists of machines that do manufacture more than one type of product, it is essential that models be developed to reflect this.

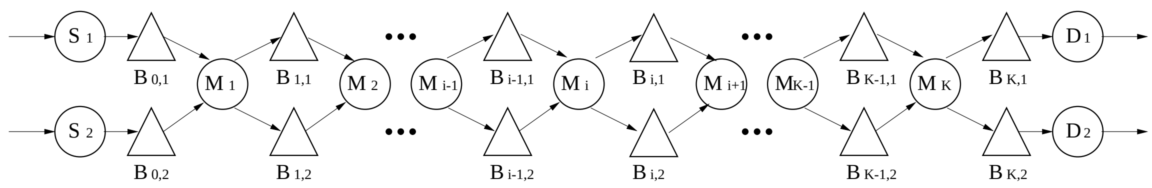

Figure 5-1: A two-part type flow line

In order to illustrate the issues involved in modeling a transfer line with more than one part type, we will begin by restricting ourselves to the case where we assume that there are two part types.

5.2 A Markov Chain Model of a Multiclass Line

The processing line model under consideration is illustrated in Figure 5-1. The line has four distinct components: the machines that produce parts — called *processing machines* — denoted by the circles labeled M; finite-capacity storage areas for work-in-process inventory, denoted by the triangles labeled B; two machines that represent the behavior of unreliable supply, denoted by the circles labeled S_1 and S_2; and two machines representing the demand process for the two part types, denoted by the circles labeled D_1 and D_2, respectively.

We assume that each processing machine in the line can operate on both part types, and that there is no set-up penalty incurred when the machine switches production from one part type to another. When machine M_i completes work on a part, it is placed in a buffer downstream of the machine. Each part type has a distinct buffer. Thus, a Type 1 produced at machine M_i would be placed in the buffer denoted by $B_{i,1}$, while a Type 2 part produced at the same machine would be placed in buffer $B_{i,2}$. The purpose of the supply and demand machines is not necessarily to model only unreliable raw material supply and finished goods demand, but to model any upstream of downstream behavior that is unpredictable, and can affect the system.

5.2.1 Finite Storage Buffers

As in the case with the single part type processing line, we assume that each buffer has finite capacity, denoted $N_{i,j}$ for buffer (i,j). We denote the current level of buffer (i,j) at the end of time step t by $n_{i,j}(t)$. As before, $0 \leq n_{i,j}(t) \leq N_{i,j}$, for all (i,j), and all $t \geq 0$. A machine is said to be *starved* for a given part type if the upstream buffer corresponding to that part type is empty. It is *blocked* for a given part type if the corresponding downstream buffer is full.

We make the assumption that the supply machines are never starved, and the demand machines are never blocked.

5.2.2 Unreliable, Homogeneous Machines with Perfect Yield

All machines in the line, including supply and demand machines, are assumed to be unreliable. We let α indicate the state of a machine. If $\alpha = 1$, then the machine is said to be *up* or *operational*. If $\alpha = 0$, the machine is said to be *down*, *failed*, or *under repair*. We let $\alpha_{1,1}(t)$ denote the state of supply machine S_1 at the end of time step t We define $\alpha_{1,2}(t)$ similarly for S_2. For D_1, and D_2, we let the corresponding state variables be $\alpha_{K+1,1}(t)$ and $\alpha_{K+1,2}(t)$. For processing machine M_i, the state variable representing the state of the machine at the end of time step t is written $\alpha_i(t)$.

We make the assumption that all the machines in the line, including the supply and demand machines, have *homogeneous processing times* — that is, all the processing times of the machines are the same, and are assumed to be scaled to unity.

Furthermore, we assume that the yield of all machines is 100%. That is, we do not allow for the scrapping or rework of parts.

5.2.3 Uncorrelated, Operation Dependent Failures

All machines, including the machines representing supply and demand, have *operation dependent failures*. That is, we assume that the machines cannot fail if they are idle. In the case of the supply machines, this means that the supply machines cannot fail if they are blocked. This also means that the demand machines cannot fail if they are starved. In the case of the processing machines, the machine cannot fail if it is either starved or blocked for type one, and starved or blocked for type two. We further assume that all failures are uncorrelated.

All machines are assumed to have geometrically distributed up and down times. For the supply machines, we let $r_{1,1}$ and $r_{1,2}$ represent the probability that machine S_1 and S_2 are up in time step $t+1$, given they were down in time step t, respectively. Likewise, $p_{1,1}$ and $p_{1,2}$ represent the probability that machine S_1 and S_2 are down in time step $t+1$, given they were up *and* not blocked in time step t. For the demand machines D_1 and D_2, the corresponding parameters are written $r_{K+1,1}$, $p_{K+1,1}$, $r_{K+1,2}$, and $p_{K+1,2}$. Formally, these are

$$
\begin{aligned}
r_{1,1} &= Pr\left[\alpha_{1,1}(t+1)=1|\alpha_{1,1}(t)=0\right] \\
r_{1,2} &= Pr\left[\alpha_{1,2}(t+1)=1|\alpha_{1,2}(t)=0\right] \\
p_{1,1} &= Pr\left[\alpha_{1,1}(t+1)=0|\alpha_{1,1}(t)=1 \cap n_{0,1}(t)<N_{0,1}\right] \\
p_{1,2} &= Pr\left[\alpha_{1,2}(t+1)=0|\alpha_{1,2}(t)=1 \cap n_{0,2}(t)<N_{0,2}\right] \\
r_{K+1,1} &= Pr\left[\alpha_{K+1,1}(t+1)=1|\alpha_{K+1,1}(t)=0\right] \\
r_{K+1,2} &= Pr\left[\alpha_{K+1,2}(t+1)=1|\alpha_{K+1,2}(t)=0\right] \\
p_{K+1,1} &= Pr\left[\alpha_{K+1,1}(t+1)=0|\alpha_{K+1,1}(t)=1 \cap n_{K,1}(t)>0\right] \\
p_{K+1,2} &= Pr\left[\alpha_{K+1,2}(t+1)=0|\alpha_{K+1,2}(t)=1 \cap n_{K,2}(t)>0\right]
\end{aligned}
$$

We assume that the probability that processing machine M_i fails is the same, regardless of the part type the machine is working on. That is, we define

$$
\begin{aligned}
p_i = Pr\big[\alpha_{i,1}(t+1)=0|&\{\alpha_{i,1}(t)=1 \cap n_{i-1,1}(t)>0 \cap n_{i,1}(t)<N_{i,1}\} \cup \\
&\{\alpha_{i,1}(t)=1 \cap (n_{i-1,1}(t)=0 \cup n_{i,1}(t)=N_{i,1}) \\
&\cap\, n_{i-1,2}(t)>0 \cap n_{i,2}(t)<N_{i,2}\}\big].
\end{aligned}
$$

For processing machine M_i, the probability of a repair is defined to be

$$r_i = Pr\,[\alpha_i(t+1) = 1 | \alpha_i(t) = 0].$$

5.2.4 System State

The state of the entire system at the end of time step t is written

$$\begin{aligned}\mathcal{S} \;=\; & (\alpha_{1,1}(t), \alpha_{1,2}(t), \alpha_1(t), \ldots, \alpha_K(t), \alpha_{K+1,1}(t), \alpha_{K+1,2}(t), \\ & n_{0,1}(t), n_{0,2}(t), n_{1,1}(t), n_{1,2}(t), \ldots, n_{K+1,1}(t), n_{K+1,2}(t))\end{aligned}$$

5.2.5 Static Priority Policy

Since each machine in the production line now must choose which part to work on when it has a choice, we are required to state a scheduling policy. Our assumption is that each machine will work on part type one whenever it can (i.e. whenever the machine is up, the upstream buffer for type one is not empty, and the downstream buffer for type one is not full). Therefore, a machine will only work on a type two part if it is up, and either blocked or starved for type on parts. We suppose that the ranking of parts at each stage is done according to some sort of $c\mu$ rule.

5.2.6 Conservation of Flow

Since the processing line has no mechanism for the destruction of parts, in steady state we require that each machine in the line makes the same number of type one and type two parts. If we denote the throughput of type j parts at machine i by $E_j(i)$, for $i = 1, \ldots, K$, and the throughput for the demand machine for part type j by E_{d_j}, then we must have

$$E_j(1) = E_j(2) = \ldots = E_j(K) = E_{d_j},$$

for $j = 1, 2$ This equation is a generalization of the Conservation of Flow equation for a single-part transfer line, given in Equation (1.5).

5.2.7 Repair Frequency Equals Failure Frequency

In order to perform the decomposition procedure for the two-part transfer line analogous to the procedure described in Chapter 1, we must classify some fundamental relationships of the system we are modeling, such as the Conservation of Flow relationships given in Section 5.2.6. This section formalizes the observation that for every failure of a machine in the line, there will be a repair. On a first glance, we would expect that for every time the machine fails making part type one, there would be a repair in which the machine would make a part type one, and for every failure during a part type two, there would be a repair that produces a part type two. However, in light of our priority scheme, we see that this is not correct. Consider the case where M_i is making a part type two because M_{i+1} has failed, and $B_{i,1}$ has filled up. Suppose M_i now fails while making a part type two. While the machine is down, M_{i+1} is repaired, and begins depleting the parts stored in $B_{i,1}$. In this case, when M_i is repaired, it will immediately begin producing type one parts, although it failed producing type two parts.

We now make this formal for the two part case. We consider the following probabilities, where all events are defined in the same time step t.

$$\begin{aligned}
E_{i,1} &= Pr\left[\alpha_i(t) = 1 \cap n_{i-1,1}(t) > 0 \cap n_{i,1}(t) < N_{i,1}\right] \\
E_{i,2} &= Pr[\alpha_i(t) = 1 \cap (n_{i-1,1}(t) = 0 \cup n_{i,1}(t) = N_{i,1}) \\
&\qquad \cap\, n_{i-1,2}(t) > 0 \cap n_{i,2}(t) < N_{i,2}] \\
D_{i,1} &= Pr\left[\alpha_i(t) = 0 \cap n_{i-1,1}(t) > 0 \cap n_{i,1}(t) < N_{i,1}\right] \\
D_{i,2} &= Pr[\alpha_i(t) = 0 \cap (n_{i-1,1}(t) = 0 \cup n_{i,1}(t) = N_{i,1}) \\
&\qquad \cap\, n_{i-1,2}(t) > 0 \cap n_{i,2}(t) < N_{i,2}] \\
I_i &= Pr[\alpha_i(t) = 1 \cap (n_{i-1,1}(t) = 0 \cup n_{i,1}(t) = N_{i,1}) \\
&\qquad \cap\, (n_{i-1,2}(t) = 0 \cup n_{i,1}(t) = N_{i,2})]
\end{aligned}$$

Here, the event $E_{i,1}$ corresponds to the event that machine M_i is up, and not blocked or starved for Type 1 parts at time t; $E_{i,2}$ the event that M_i is up, either blocked or starved for Type 1 parts, and not blocked or starved for Type 2 parts, at time t; $D_{i,1}$ the event that M_i is down, and not blocked or starved for Type 1, at time t; $D_{i,2}$ the event that M_i is down, blocked or starved for Type 1, and not blocked or starved for Type 2, at time t; and I_i the event that M_i is up, and either blocked or starved for both Type 1 and Type 2 parts. Since the real machine in the long processing line can only fail if it is operational, then it cannot fail if it is either blocked or starved for both parts. Therefore,

$$Pr\left[\alpha_i = 0 \cap (n_{i-1,1} = 0 \cup n_{i,1} = N_{i,1}) \cap (n_{i-1,2} = 0 \cup n_{i,1} = N_{i,2})\right] = 0.$$

It follows that the states defined in the five probability expressions from a mutually exclusive, collectively exhaustive list of all possible combinations of machine and buffer states. Therefore,

$$E_{i,1} + E_{i,2} + D_{i,1} + D_{i,2} + I_i = 1 \tag{5.1}$$

In steady state, for every failure of machine M_i, there is a repair of machine M_i. Therefore, in steady state we have

$$(E_{i,1} + E_{i,2})p_i = (D_{i,1} + D_{i,2})r_i \tag{5.2}$$

We now come upon a subtle, but important point. In the single part type decomposition, it can be proved that in steady state, for every failure making a part, there is a repair. Unfortunately, the same does not hold for a processing line with more than one part type under our current scheduling rule. That is, if a machine comes up making type one, we cannot say for sure that it went down making a type one. This is because it is possible that the machine went down making a type two, while starved or blocked for type one (a necessary condition for the machine to be able to produce type two parts), and while the machine was down, the blockage or starvation for type one that affected is removed by a repair up- or downstream of the machine. That is, while we can assert that the relationship expressed in (5.2) holds true, we are unable to assert that either

$$p_i E_{i,1} = r_i D_{i,1}, \tag{5.3}$$

or

$$p_i E_{i,2} = r_i D_{i,2} \tag{5.4}$$

are true. Therefore, the compact flow rate-idle time relationship observed in the single part-type case no longer holds. This presents us with a difficulty in deriving the multiple part type decomposition. To overcome this, we will *assume* that (5.3) and (5.4) do hold. This will restrict the cases where the decomposition is accurate to those where the relationships do hold closely. Specifically, we expect good performance when the machines in the processing line do not stay down too relative to the amount of time they are blocked or starved for Type 1 parts. We then expect that almost every time a machine fails while making a Type 2 part, it will come up making a Type 2 part.

5.2.8 Flow Rate-Idle Time

In Section 5.2.7 we stated that the difficulty in deriving an uptime-downtime relationship for type one lay in the fact that if the processing machine came up after a failure making a type one, we could not determine whether it went down making a type one or a type two. In this case, given our current state, we cannot say anything about the past. However, if the machine goes down making a type one, then we know that it will come up making a type one, for it is impossible for the machine to become either blocked or starved for type one while it is down.

For the second part type, we have a similar problem. If the machine fails while making a part type two, we cannot say for certain that it will come back up making a part type two. That is, we have no information about the future, given our current state. However, if we observe the machine coming up making a type two, then we know that it went down making a type two. That is, given our current state, we do have information about the past.

These subtleties — that for part type one, we know something about the future, but not about the past, and for part type two, we know something about the past, but not the future — make the derivation of exact flow rate-idle time relationships difficult. We will, instead, resort to approximate relationships.

The quantity $E_1(i)$ represents the probability that machine M_i will make a type one part in a give time step. We know that machine M_i will make a type one part at the end of time $t+1$ if

- M_i is not starved for type one at time t,
- M_i is not blocked for type one at time t, and
- M_i is up at the end of time $t+1$.

This probability is expressed as follows

$$E_1(i) \quad = \quad Pr\left[\alpha_i(t+1) = 1 \cap n_{i-1,1}(t) > 0 \cap n_{i,1}(t) < N_{i,1}\right] \tag{5.5}$$

In the above quantity, we have an expression that is evaluated at two different time steps. In the single part type processing line, it is possible, by means of conditioning using a law of total probability expansion, to write the corresponding expression in terms of the same time step. The procedure relies on the fact that, in the single part type case, the number of failures while making part type one equals the number of repairs that come up making type one. That is, in the single part type case, we can show that $E_1(i) = E_{i,1}$, where $E_{i,1}$ was defined in Section 5.2.7, and that this relationship is exact. Due to the issues discussed at the beginning of this section, we are unable to assert this at the part-type level when the system produces more than one part type. However, under the assumption that (5.3) holds, we proceed as follows.

$$\begin{aligned}
E_{i,1} &= Pr\left[\alpha_i(t+1) = 1 \cap n_{i-1,1}(t) > 0 \cap n_{i,1}(t) < N_{i,1}\right] \\
&= Pr\left[\alpha_i(t+1) = 1 | \alpha_i(t) = 1 \cap n_{i-1,1}(t) > 0 \cap n_{i,1}(t) < N_{i,1}\right] \\
&\quad \times Pr\left[\alpha_i(t) = 1 \cap n_{i-1,1}(t) > 0 \cap n_{i,1}(t) < N_{i,1}\right] \\
&+ Pr\left[\alpha_i(t+1) = 1 | \alpha_i(t) = 0 \cap n_{i-1,1}(t) > 0 \cap n_{i,1}(t) < N_{i,1}\right] \\
&\quad \times Pr\left[\alpha_i(t) = 0 \cap n_{i-1,1}(t) > 0 \cap n_{i,1}(t) < N_{i,1}\right] \\
&= (1-p_i) Pr\left[\alpha_i(t) = 1 \cap n_{i-1,1}(t) > 0 \cap n_{i,1}(t) < N_{i,1}\right] \\
&\quad + r_i Pr\left[\alpha_i(t) = 0 \cap n_{i-1,1}(t) > 0 \cap n_{i,1}(t) < N_{i,1}\right]
\end{aligned}$$

$$\begin{aligned} &= (1-p_i)E_{i,1} + r_i D_{i,1} \\ &\approx E_{i,1}, \end{aligned}$$

where the last step follows from the assumption that (5.3) holds approximately. Again, we emphasize that in the single part-type case, this relationship is exact. We therefore make the following approximation.

$$E_1(i) \approx Pr\left[\alpha_i(t) = 1 \cap n_{i-1,1}(t) > 0 \cap n_{i,1}(t) < N_{i,1}\right] \tag{5.6}$$

Under the assumption that (5.3) holds, it is possible to rewrite the right hand side of (5.6) to get

$$E_1(i) \approx e_i Pr\left[n_{i-1,1}(t) > 0 \cap n_{i,1}(t) < N_{i,1}\right]. \tag{5.7}$$

We follow a similar line of argument for the Type 2 production rate. Let the quantity $E_2(i)$ denote the probability that that machine M_i makes a type two part in a given time step. From our assumptions, we know that machine M_i will make a type two part at time $t+1$ if

- M_i is either blocked or starved for type one at time t,
- M_i is not starved for type two at time t,
- M_i is not blocked for type two at time t, and
- M_i is up at the end of time $t+1$.

This probability is expressed as follows,

$$\begin{aligned} E_2(i) \;=\; & Pr\left[\alpha_i(t+1) = 1 \cap (n_{i-1,1}(t) = 0 \cup n_{i,1}(t) = N_{i,1}) \right. & (5.8) \\ & \left. \cap\, n_{i-1,2}(t) > 0 \cap n_{i,2}(t) < N_{i,2}\right] & \\ & & (5.9) \end{aligned}$$

As with the case of the production rate for part type one, we have an expression written in terms of two different time steps. Using the assumption in (5.4), we can derive the following approximation in a manner similar to that done for $E_1(i)$.

$$\begin{aligned} E_2(i) \;\approx\; & Pr\left[\alpha_i(t) = 1 \cap (n_{i-1,1}(t) = 0 \cup n_{i,1}(t) = N_{i,1}) \right. \\ & \left. \cap\, n_{i-1,2}(t) > 0 \cap n_{i,2}(t) < N_{i,2}\right] \end{aligned} \tag{5.10}$$

In order to make the decomposition tractable, we make the following approximation.

$$E_2(i) \approx e_i Pr\left[(n_{i-1,1}(t) = 0 \cup n_{i,1}(t) = N_{i,1})\right. \tag{5.11}$$

$$\cap\, n_{i-1,2}(t) > 0 \cap n_{i,2}(t) < N_{i,2}] \,.$$

We take (5.11) to be our expression for the production rate of Type 2 parts.

5.3 Approximation of the Model

The system defined in Section 5.2 is a Markov chain, and it is in principal possible to write out a state transition diagram. However, this is not practical, because the size of the state space is too large. In order evaluate the system, it is necessary to find an approximation to the Markov chain that models it.

As was done with the single part-type transfer line, we intend to break down the larger system into smaller, analytically tractable pieces, and capture the local behavior of the long line, as seen by an observer in a buffer, by choosing appropriate parameters of the two-machine sub-lines. This decomposition procedure is represented in Figure 5-2. As discussed earlier, the idea is to fool an observer in a buffer in the long, multiclass processing line into thinking he is in a two-machine line. In the figure, the inflow and outflow behavior of material an observer in buffer $B_{i,1}$ would see is modeled by the two-machine, one-part line $L_{i,1}$. Close observation of the dynamics of the long line, however, show the necessity for a new two-machine line model. The reasoning is as follows.

Suppose that we take the point of view of an observer in the buffer $B_{i,1}$. We misinform this observer: we lead him to believe that he is watching the flow in the only buffer in a two-machine, one buffer, one-part type system. Let us assume that the observer sees that the outflow from his buffer has ceased, but the inflow has not. (The reason for the stoppage of outflow is irrelevant — it could be due to a failure of machine M_{i+1}, or the failure of a machine further downstream, leading to a blockage of M_{i+1} for type one parts.) Eventually, unless the outflow resumes or the inflow ceases, the buffer $B_{i,1}$ will fill up. According to our scheduling rule, M_i will immediately begin making Type 2 parts, if it is able to. Suppose it does, and that M_i fails while making a type two part.Now suppose that while M_i is down, the outflow from $B_{i,1}$ begins again. Then the sequence of events that the observer will see are that the outflow ceased, the buffer filled up, but when the outflow began again, the inflow did not. As far as the observer in the buffer is concerned, the machine upstream of him failed while it was blocked!

There is a subtlety here that must be paid close attention to . While this apparent idleness failure is behavior that an observer in a buffer sees, it is important to realize that the real machines do not fail when they are idle. It only appears to the observer that the machine has failed during an idle period, because the observer believes that he is in a two-machine, one-part type line. Therefore, while in our previous model we assumed that both the real machines and the pseudo-machines in the two-machine sub-lines had operation dependent failures, we must relax that assumption for the two-machine sub-lines in the multiple part type case. Thus, a new two-machine line model is in order. We present a discrete-time, discrete-state Markov model of precisely such a line in Chapter 6.

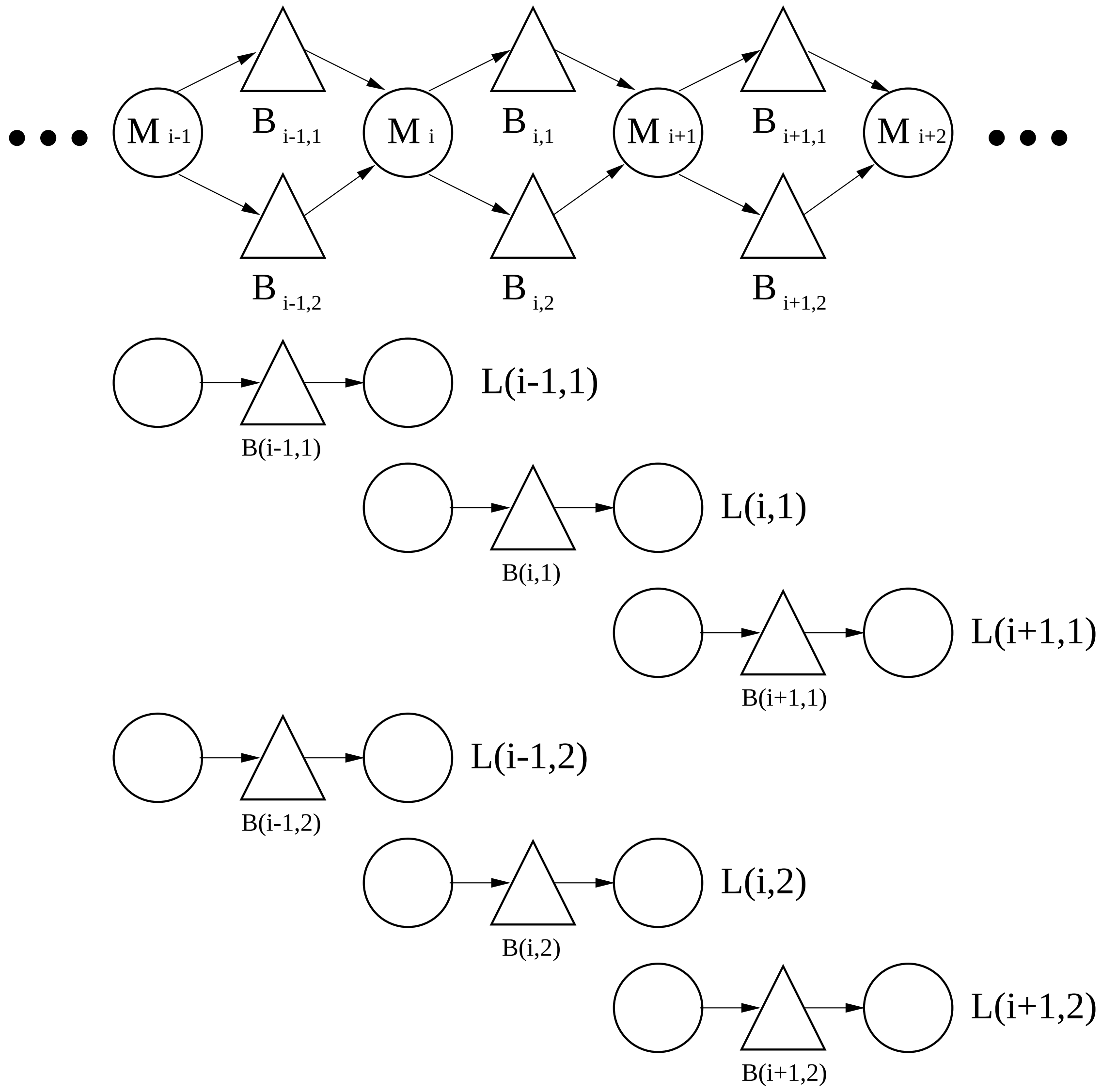

Figure 5-2: The decomposition of a long two-part processing line into two-machine, one-part sub-lines.

Chapter 6

A New Two-Machine Line Model

6.1 Introduction

As was discussed in Section 5.3, in order to decompose the Markov chain model of the two part-type processing line, we need a new two-machine line. In this chapter we present the analysis of a Markov model of a two-machine transfer line with a finite intermediate storage area and two unreliable machines that are no longer restricted to failing only if they are not blocked or starved. The line in question is similar to the deterministic processing time model described in Chapter 3 of Gershwin [16], with the important difference that we remove the assumption that machines in the line can only fail when they are operating on a part. This leads to changes in the balance equations, fundamental identities such as the flow rate-idle time and conservation of flow, and behavior of the two-machine line model. The general format of this chapter is similar to that of Section 3.6 in Gershwin [16].

6.2 Model Assumptions and Notation

The two-machine line is illustrated in Figure 6-1. As is our convention, the machines are denoted by circles, and the buffer by a triangle. We denote the upstream machine by M_u, and the downstream machine by M_d. We denote the size of the intermediate buffer by N, and the current level of the intermediate buffer by n. It follows that $0 \leq n \leq N$.

Material flows into the upstream machine from an infinite supply, is processed by the machine, and when processing is complete, the material is placed into the buffer until it is processed by the downstream machine. Upon finishing processing by the downstream machine, the part leaves the line. We assume that there is always room for the downstream machine to unload a part it has just completed processing. We make the assumption that there is only one class of parts produced by the line, and that the production time at each of the machines is identical, and equal to one.

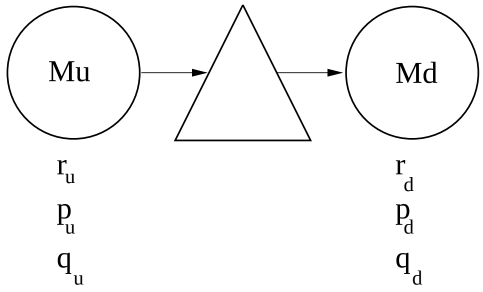

Figure 6-1: A two-machine transfer line with idleness failures.

The machines are unreliable. We assume that the machines can fail while they are either operating on a part, or while they are idle, but we do not assume that the probabilities of failure are identical. In particular, we assume that the probability that M_u fails while it is working on a part, given it is not blocked, is sp_u, and the probability that it fails while it is blocked is q_u. Note that we do *not* assume that there is a new failure mode, but only that there is a new way of reaching the failure mode. We define the quantities p_d and q_d for M_d similarly. Finally, we denote the probabilities that M_u and M_d are repaired while they are down by r_u and r_d, respectively. We call a failure that takes place while the machine is operating on a part an *operational failure*, and a failure that occurs when the machine is idle an *idleness failure*.

Finally, we define the state of the two-machine line to be $s = (n, \alpha_u, \alpha_d)$, where n is the level of the intermediate buffer ($0 \leq n \leq N$), α_u is 0 if M_u is down, and 1 if M_u is up, and α_d is 0 if M_d is down, and 1 if M_d is up.

6.3 Balance Equations

In this section we define the balance equations for the Markov model. The relaxation of the requirement that machines only fail while they are operating on a part leads states that were previously transient now to have positive probability. However, not all of the previously transient states now have positive measure. The following states are still transient.

- (0,1,0) is transient because it cannot be reached from any other state. If $\alpha_u(t+1) = 1$ and $\alpha_d(t+1) = 0$, then $n(t+1) = n(t) + 1$. That is, if the upstream machine is repaired in time step $t+1$, but the downstream machine remains down, then the part produced upon the repair of M_u will be unloaded into the intermediate buffer.

- (0,1,1) is transient because it cannot be reached from any other state. If $\alpha_u(t+1) = 1$ and $\alpha_d(t+1) = 1$, but $n(t) = 0$, then M_d will be unable to produce a part at time step $t+1$ because it was starved at time t. However, M_u will produce a part in time step $t+1$ because of the assumption that it is never starved, and the fact that it is not blocked at time step t.

- (N,0,1) is transient. If $\alpha_d(t+1) = 1$ and $\alpha_u(t+1) = 0$, and $n(t) > 0$, then M_d will produce a part in time step $t+1$ because it was not starved in times step t, and is never blocked, by assumption.

- (N,1,1) is transient. If $\alpha_u(t+1) = 1$ and $\alpha_d(t+1) = 1$, and $n(t) > 0$, then M_d will produce a part in time step $t+1$ because it was not starved at time step t, and is assumed never to be blocked, but M_u will not produce a time step in time $t+1$ because it was blocked at time step t.

The system can reach state $(0,0,0)$ from the state $(0,0,0)$ if neither the upstream nor downstream machines are repaired. This occurs with probability $(1-r_u)(1-r_d)$. Similarly, the system can reach state $(0,0,0)$ from the state $(0,0,1)$ if the downstream machine does not incur an idleness failure, and the upstream machine is not repaired. This happens with probability $(1-r_u)q_d$. Since the state $(0,0,0)$ cannot be reached from any other non-transient states, the transition equation for the state $(0,0,0)$ is given by

$$\mathbf{p}(0,0,0) = (1-r_u)(1-r_d)\mathbf{p}(0,0,0) + (1-r_u)q_d\mathbf{p}(0,0,1).$$

Making similar arguments for all the possible states of the Markov chain leads to the following balance equations:

Lower Boundary Equations, $n \leq 1$

$$\begin{aligned}
\mathbf{p}(0,0,0) &= (1-r_u)(1-r_d)\mathbf{p}(0,0,0) + (1-r_u)q_d\mathbf{p}(0,0,1) && (6.1)\\
\mathbf{p}(1,1,0) &= r_u q_d\mathbf{p}(0,0,1) + r_u(1-r_d)\mathbf{p}(0,0,0) && (6.2)\\
\mathbf{p}(0,0,1) &= (1-r_u)r_d\mathbf{p}(0,0,0) + p_u r_d\mathbf{p}(1,1,0) && (6.3)\\
&\quad +(1-r_u)(1-q_d)\mathbf{p}(0,0,1) + (1-r_u)r_d\mathbf{p}(1,0,0)\\
&\quad +(1-r_u)(1-p_d)\mathbf{p}(1,0,1) + p_u(1-p_d)\mathbf{p}(1,1,1)\\
\mathbf{p}(1,0,0) &= p_u(1-r_d)\mathbf{p}(1,1,0) + (1-r_u)(1-r_d)\mathbf{p}(1,0,0) && (6.4)\\
&\quad +(1-r_u)p_d\mathbf{p}(1,0,1) + p_u p_d\mathbf{p}(1,1,1)\\
\mathbf{p}(1,0,1) &= (1-r_u)r_d\mathbf{p}(2,0,0) + (1-r_u)(1-p_d)\mathbf{p}(2,0,1) && (6.5)\\
&\quad +p_u r_d\mathbf{p}(2,1,0) + p_u(1-p_d)\mathbf{p}(2,1,1)\\
\mathbf{p}(1,1,1) &= r_u r_d\mathbf{p}(0,0,0) + (1-p_u)r_d\mathbf{p}(1,1,0) && (6.6)\\
&\quad +r_u(1-q_d)\mathbf{p}(0,0,1) + r_u r_d\mathbf{p}(1,0,0)\\
&\quad +r_u(1-p_d)\mathbf{p}(1,0,1) + (1-p_u)(1-p_d)\mathbf{p}(1,1,1)\\
\mathbf{p}(2,1,0) &= (1-p_u)(1-r_d)\mathbf{p}(1,1,0) + r_u(1-r_d)\mathbf{p}(1,0,0) && (6.7)\\
&\quad +r_u p_d\mathbf{p}(1,0,1) + (1-p_u)p_d\mathbf{p}(1,1,1)
\end{aligned}$$

These equations describe the relationships between states in the two-machine line model. We take the state $(0,0,0)$ as an example. Equation (6.1) says that state $(0,0,0)$ can only be reached from two other states: itself, and $(0,0,1)$. The probability of the transition from state $(0,0,0)$ to state $(0,0,0)$ is the probability that neither the upstream, nor the downstream machine is repaired. Since repairs of the machines are independent events, this is just the product of the probabilities that the machines are not repaired, or $(1-r_u)(1-r_d)$. The probability of the transition from state $(0,0,1)$ to $(0,0,0)$ is the probability that the upstream machine was not repaired, and the downstream machine underwent an idleness failure. This is given as $(1-r_u)q_d$. Equations (6.2) through (6.7) are derived following similar analysis.

Internal Equations,$2 \leq n \leq N-2$ The internal equations are identical to the internal equations in Gershwin [16]. They are given here for completeness.

$$\begin{aligned}
\mathbf{p}(n,0,0) &= (1-r_u)(1-r_d)\mathbf{p}(n,0,0)+(1-r_u)p_d\mathbf{p}(n,0,1) && (6.8)\\
&\quad +p_u(1-r_d)\mathbf{p}(n,1,0)+p_u p_d\mathbf{p}(n,1,1)\\
\mathbf{p}(n,0,1) &= (1-r_u)r_d\mathbf{p}(n+1,0,0)+(1-r_u)(1-p_d)\mathbf{p}(n+1,0,1) && (6.9)\\
&\quad +p_u r_d\mathbf{p}(n+1,1,0)+p_u(1-p_d)\mathbf{p}(n+1,1,1)\\
\mathbf{p}(n,1,0) &= r_u(1-r_d)\mathbf{p}(n-1,0,0)+r_u p_d\mathbf{p}(n-1,0,1) && (6.10)\\
&\quad (1-p_u)(1-r_d)\mathbf{p}(n-1,1,0)+(1-p_u)p_d\mathbf{p}(n-1,1,1)\\
\mathbf{p}(n,1,1) &= r_u r_d\mathbf{p}(n,0,0)+r_u(1-p_d)\mathbf{p}(n,0,1)+(1-p_u)r_d\mathbf{p}(n,1,0) && (6.11)\\
&\quad (1-p_u)(1-p_d)\mathbf{p}(n,1,1)
\end{aligned}$$

Upper Boundary Equations, $n \geq N-1$ The upper boundary equations are derived following exactly the same reasoning procedure as was used for the lower boundary equations.

$$\begin{aligned}
\mathbf{p}(N-2,0,1) &= (1-r_u)(1-p_d)\mathbf{p}(N-1,0,1) && (6.12)\\
&\quad +(1-r_u)r_d\mathbf{p}(N-1,0,0)\\
&\quad +p_u r_d\mathbf{p}(N-1,1,0)+p_u(1-p_d)\mathbf{p}(N-1,1,1)\\
\mathbf{p}(N-1,0,1) &= (1-r_u)r_d\mathbf{p}(N,0,0)+q_u r_d\mathbf{p}(N,1,0) && (6.13)\\
\mathbf{p}(N-1,0,0) &= (1-r_u)p_d\mathbf{p}(N-1,0,1) && (6.14)\\
&\quad +(1-r_u)(1-r_d)\mathbf{p}(N-1,0,0)\\
&\quad +p_u(1-r_d)\mathbf{p}(N-1,1,0)+p_u p_d\mathbf{p}(N-1,1,1)\\
\mathbf{p}(N-1,1,0) &= r_u(1-r_d)\mathbf{p}(N-2,0,0)+r_u p_d\mathbf{p}(N-2,1,1) && (6.15)
\end{aligned}$$

$$
\begin{aligned}
& +(1-p_u)(1-r_d)\mathbf{p}(N-2,1,0) \\
& +(1-p_u)p_d\mathbf{p}(N-2,1,1) \\
\mathbf{p}(N-1,1,1) &= r_u r_d \mathbf{p}(N-1,0,0) + (1-p_u)r_d\mathbf{p}(N-1,1,0) \qquad (6.16) \\
& +(1-p_u)(1-p_d)\mathbf{p}(N-1,1,1) + (1-q_u)r_d\mathbf{p}(N,1,0) \\
& +r_u r_d\mathbf{p}(N,0,0) + r_u(1-p_d)\mathbf{p}(N-1,0,1) \\
\mathbf{p}(N,1,0) &= r_u(1-r_d)\mathbf{p}(N-1,0,0) + r_u p_d\mathbf{p}(N-1,0,1) \qquad (6.17) \\
& +r_u(1-r_d)\mathbf{p}(N-1,0,0) + (1-p_u)(1-r_d)\mathbf{p}(N-1,1,0) \\
& +(1-p_u)r_d\mathbf{p}(N-1,1,1) + (1-q_u)(1-r_d)\mathbf{p}(N,1,0) \\
\mathbf{p}(N,0,0) &= (1-r_u)(1-r_d)\mathbf{p}(N,0,0) + q_u(1-r_d)\mathbf{p}(N,1,0) \qquad (6.18)
\end{aligned}
$$

Normalization Finally, since the total probability of all states summed must be unity, we have the following normalization equation.

$$1 = \sum_{\alpha_u=0}^{1} \sum_{\alpha_d=0}^{1} \sum_{n=0}^{N} \mathbf{p}(n, \alpha_u, \alpha_d) \qquad (6.19)$$

Equations (6.1) through (6.19) define a probability mass function for the Markov chain model of the two-machine transfer line. Before we turn our attention to their solution, however, we address the issue of the performance measures of the line, as well as the expression of several of the fundamental physical properties of the line. We begin with performance measures.

6.4 Performance Measures

The main performance measures of interest in the new two-machine line model are the *throughput*, *average buffer level*, *probability of starvation*, and *probability of blockage.*

The throughput of the upstream machine is the probability that the upstream machine is working in time step $t+1$, and not blocked at time t. This is can be expressed as

$$E_u = Pr\left[\alpha_u(t+1) = 1 \cap n(t) < N\right] \qquad (6.20)$$

Similarly, the throughput of the downstream machine is the probability that the downstream machine is working at time $t+1$ and not starved at time t. This is given by

$$E_d = Pr\left[\alpha_d(t+1) = 1 \cap n(t) > 0\right] \qquad (6.21)$$

The average buffer level is denoted $\overline{n}$, and is given by

$$\overline{n} = \sum_{\alpha_u=0}^{1} \sum_{\alpha_d=0}^{1} \sum_{n=0}^{N} n\mathbf{p}(n, \alpha_u, \alpha_d) \tag{6.22}$$

The probability that the downstream machine is starved, denoted p_s, is the probability that the machine is up, but that there is no part in the buffer. Since the state $(0,1,1)$ is transient, this is given by

$$p_s = \mathbf{p}(0,0,1). \tag{6.23}$$

Finally, the probability that the upstream machine is blocked, denoted p_b, is the probability that the machine us up, and that the intermediate buffer is full. Since the state $(N,1,1)$ is transient, this is given by

$$p_b = \mathbf{p}(N,1,0) \tag{6.24}$$

6.5 Identities

The underlying physical nature of the transfer line being modeled, and the logic of the of the model of the tandem line itself gives us some identities that must hold true in order for the model to be accurate. These are as follows.

6.5.1 Repair Frequency Equals Failure Frequency

For every machine failure, there is also a repair. That is, for both the up and downstream machines in the two-machine line, the conditional probability that a machine is repaired, given it is down, times the probability that it is down is equal to the conditional probability that the machine fails times the probability that the machine is up. For the upstream machine, this is expressed as

$$\begin{aligned} r_u Pr\left[\{\alpha_u = 0\} \cap \{n < N\}\right] &= p_u Pr\left[\{\alpha_u = 1\} \cap \{n < N\}\right] \\ &+ q_u Pr\left[\{\alpha_u = 1\} \cap \{n = N\}\right] \end{aligned} \tag{6.25}$$

Similarly, for the downstream machine we have

$$\begin{aligned} r_d Pr\left[\{\alpha_d = 0\} \cap \{n > 0\}\right] &= p_d Pr\left[\{\alpha_u = 1\} \cap \{n > 0\}\right] \\ &+ q_d Pr\left[\{\alpha_u = 1\} \cap \{n = 0\}\right] \end{aligned} \tag{6.26}$$

We can use Equations (6.25) and (6.26) to derive alternative expressions for Equations (6.20) and (6.21). Observe that the upstream machine, E_u produces a part in time step $t+1$ if it is up at the end of time step $t+1$, and was not blocked at the end of time step t. We can then write E_u as follows:

$$E_u = Pr\left[\alpha_u(t+1) = 1 \cap n(t) < N\right]. \tag{6.27}$$

Observe that this expression has both time step $t+1$ and time step t in it. We proceed by conditioning on events occurring in time step t to write Equation (6.27) in terms of events occurring entirely in time step t. By doing so, we will be able to express the production rate of the upstream machine entirely in terms of the state probabilities, which are defined only on one time step.

$$\begin{aligned}
E_u &= Pr\left[\alpha_u(t+1) = 1 \cap n(t) < N\right] \\
&= Pr\left[\alpha_u(t+1) = 1 | \alpha_u(t+1) = 1 \cap n(t) < N\right] \\
&\quad \times Pr\left[\alpha_u(t+1) = 1 \cap n(t) < N\right] \\
&\quad + Pr\left[\alpha_u(t+1) = 1 | \alpha_u(t+1) = 0 \cap n(t) < N\right] \\
&\quad \times Pr\left[\alpha_u(t+1) = 0 \cap n(t) < N\right] \\
&= (1 - p_u) Pr\left[\alpha_u(t+1) = 1 \cap n(t) < N\right] \\
&\quad + r_u Pr\left[\alpha_u(t+1) = 0 \cap n(t) < N\right] \\
&= Pr\left[\alpha_u(t+1) = 1 \cap n(t) < N\right] + q_u Pr\left[\alpha_u(t) = 1 \cap n(t) = N\right] \\
&\quad - r_u Pr\left[\alpha_u(t) = 0 \cap n(t) = N\right]
\end{aligned}$$

Observing that

$$Pr\left[\alpha_u(t) = 1 \cap n(t) = N\right] = \mathbf{p}(N10),$$

and that

$$Pr\left[\alpha_u(t) = 0 \cap n(t) = N\right] = \mathbf{p}(N00),$$

and following the same derivation procedure for E_d in Equation (6.26), we arrive at

$$E_u = Pr\left[\alpha_u(t) = 1 \cap n(t) < N\right] + q_u \mathbf{p}(N10) - r_u \mathbf{p}(N00) \tag{6.28}$$

$$E_d = Pr\left[\alpha_d(t) = 1 \cap n(t) > 0\right] + q_d \mathbf{p}(001) - r_d \mathbf{p}(000) \tag{6.29}$$

These expressions will be useful in the derivation of the resumption of flow equations in the next chapter.

6.5.2 Conservation of Flow

Since we provide no mechanism for the creation or destruction of material, and since we assume perfect yield, then the probability that the upstream machine makes a part in any given time step must be equal to the probability that the downstream machine makes a part in any given time step. That is, we require that $E_u = E_d$. We prove this from the balance equations, and Equations (6.20) and (6.21). We do so by exploiting an interesting fact having to do with the balance equations first observed by Gershwin [16]. If we add Equations (6.1), (6.2),(6.3),(6.4),(6.6), and (6.7), and cancel everything that can be canceled, we find:

$$\mathbf{p}(101) = \mathbf{p}(210)$$

Furthermore, if we perform an analogous operation with the internal equations, we find that

$$\mathbf{p}(n+1,10) = \mathbf{p}(n01), \quad \text{for } n = 1,2,\ldots,N-2$$

Now, from Equations (6.28) and (6.29), we proceed as follows:

$$\begin{aligned}
E_u - E_d &= Pr\left[\alpha_u(t) = 1 \cap n(t) < N\right] + q_u\mathbf{p}(N10) - r_u\mathbf{p}(N00) \\
&\quad -Pr\left[\alpha_d(t) = 1 \cap n(t) > 0\right] - q_d\mathbf{p}(001) + r_d\mathbf{p}(000) \\
&= \sum_{n=0}^{N-1}\mathbf{p}(n10) + \sum_{n=0}^{N-1}\mathbf{p}(n11) - \sum_{n=1}^{N}\mathbf{p}(n01) + \sum_{n=1}^{N}\mathbf{p}(n11) \\
&\quad +q_u\mathbf{p}(N10) - r_u\mathbf{p}(N00) - q_d\mathbf{p}(001) + r_d\mathbf{p}(000) \\
&= \sum_{n=0}^{N-1}\mathbf{p}(n10) - \sum_{n=1}^{N}\mathbf{p}(n01) \\
&\quad +q_u\mathbf{p}(N10) - r_u\mathbf{p}(N00) - q_d\mathbf{p}(001) + r_d\mathbf{p}(000) \\
&= \mathbf{p}(110) - \mathbf{p}(N-1,01) + q_u\mathbf{p}(N10) - r_u\mathbf{p}(N00) - q_d\mathbf{p}(001) - r_d\mathbf{p}(000)
\end{aligned}$$

In what remains above, we now make the following substitutions.

$$\mathbf{p}(000) = \frac{(1-r_u)q_d}{r_u + r_d - r_u r_d}\mathbf{p}(001) \tag{6.30}$$

$$\mathbf{p}(110) = \frac{r_u q_d}{r_u + r_d - r_u r_d}\mathbf{p}(001) \tag{6.31}$$

$$\mathbf{p}(N00) = \frac{q_u(1-r_d)}{r_u + r_d - r_u r_d}\mathbf{p}(N10) \tag{6.32}$$

$$\mathbf{p}(N-1,01) = \frac{q_u r_d}{r_u + r_d - r_u r_d}\mathbf{p}(N10) \tag{6.33}$$

By canceling terms that can be canceled, and rewriting, we obtain the following:

$$E_u - E_d = q_u\left(1 - \frac{r_d}{r_u + r_d - r_u r_d} - \frac{r_u(1-r_d)}{r_u + r_d - r_u r_d}\right)\mathbf{p}(N10)$$

$$
\begin{aligned}
&+q_d \left(\frac{r_u}{r_u + r_d - r_u r_d} + \frac{r_d(1-r_u)}{r_u + r_d - r_u r_d} - 1 \right) \mathbf{p}(001) \\
= \;& 0
\end{aligned}
$$

6.5.3 Flow Rate-Idle Time

We now derive the flow rate-idle time relationships for the two-machine line. Define the following quantities:

$$
\begin{aligned}
A_u &= Pr\left[\alpha_u(t) = 1 \cap n(t) < N\right] \\
D_u &= Pr\left[\alpha_u(t) = 0 \cap n(t) < N\right] \\
F_u &= Pr\left[\alpha_u(t) = 0 \cap n(t) = N\right] \\
p_b &= Pr\left[\alpha_u(t) = 1 \cap n(t) = N\right]
\end{aligned}
$$

Observe that all the events in the above expressions are evaluated at the same time step. These quantities have the following interpretations. A_u is the probability that the upstream machine is up, and not blocked at time step t. D_u is the probability that the upstream machine is down and not blocked at time step t. F_u is the probability that the upstream machine is down and blocked at time step t. Last, p_b is the probability that the upstream machine is up and blocked at time step t. It follows that

$$A_u + D_u + F_u + p_b = 1 \tag{6.34}$$

Now, rewrite Equations (6.25) and (6.28) using the above notation to get

$$
\begin{aligned}
r_u D_u + r_u F_u &= p_u A_u + q_u p_b \\
E_u &= A_u + q_u p_b - r_u F_u
\end{aligned}
$$

Some simple algebraic manipulation of these equations yields

$$
\begin{aligned}
D_u &= \frac{p_u}{r_u} E_u - (1-p_u) F_u + \frac{q_u}{r_u}(1-p_u) p_b && (6.35) \\
A_u &= E_u - q_u p_b + r_u F_u && (6.36)
\end{aligned}
$$

Now, from Equation (6.34), we have

$$1 - p_b - F_u = A_u + D_u. \tag{6.37}$$

By substituting the expressions in (6.35) and (6.36) into (6.37) and manipulating algebraically, we arrive at

$$E_u = e_u(1 - p_b) - e_u q_u p_b \left(\frac{r_u + r_d - p_u r_d - 2 r_u r_d}{r_u (r_u + r_d - r_u r_d)} \right). \tag{6.38}$$

In a similar manner, it is possible to derive the following expression for the downstream machine.

$$E_d = e_d(1 - p_s) - e_d q_d p_s \left(\frac{r_u + r_d - r_u p_d - 2 r_u r_d}{r_d (r_u + r_d - r_u r_d)} \right). \tag{6.39}$$

Now, observe that if the idleness failure probabilities in either of the above equations are zero, then that expression reduces to that found in Gershwin [16]. That is, they reduce to the standard flow rate-idle time relationship for the discrete-time, discrete-state two-machine transfer line model. Furthermore, since we have an added opportunity for either the upstream or downstream machines to fail, we would expect the E's of both machines to be lower in the modified two-machine line. It can be readily verified that if both r_u and r_d are less than 0.5, this occurs.

6.5.4 Solution of the Balance Equations

We now solve the balance equations for the probability mass function for the Markov model of the two-machine line. To begin, observe that the internal balance equations given above are identical to those given in Gershwin [16]. This leads us to immediately deduce that the internal equations will be of the form

$$\mathbf{p}(n, \alpha_u, \alpha_d) = C_1 X_1^n Y_{11}^{\alpha_u} Y_{21}^{\alpha_d} + C_2 X_2^n Y_{21}^{\alpha_u} Y_{22}^{\alpha_d}, \tag{6.40}$$

where $X_1, X_2, Y_{11}, Y_{12}, Y_{21}$ and Y_{22} are exactly as given on page 86 of Gershwin [16]. Now, every term in the right-hand side of Equation (6.5) is of internal form, so we can conclude that

$$\mathbf{p}(101) = C_1 X_1 Y_{21} + C_2 X_2 Y_{22}.$$

Furthermore, as it was shown in Section 6.5.2, $\mathbf{p}(210) = \mathbf{p}(101)$. Since the state $(2, 1, 0)$ is internal, we have

$$\mathbf{p}(210) = C_1 X_1^2 Y_{11} + C_2 X_2^2 Y_{12}$$

By subtracting $\mathbf{p}(101)$ from $\mathbf{p}(210)$ and collecting terms, we get

$$C_1 \left(Y_{11} - Y_{21} \right) + C_2 \frac{Y_{22}}{Y_{12}} \left(\frac{Y_{22}}{Y_{12}} Y_{12} - Y_{22} \right) = 0$$

from which it follows immediately that

$$C_1\left(Y_{11} - Y_{21}\right) = 0.$$

This implies that $C_1 = 0$, and that the solution for internal states is given by

$$\mathbf{p}(n, \alpha_u, \alpha_d) = CX^n Y_1^{\alpha_u} Y_2^{\alpha_d},$$

where

$$\begin{aligned}
Y_1 &= \frac{R - r_u p_d}{P - p_u r_d} && (6.41)\\
Y_2 &= \frac{R - p_u r_d}{P - r_u p_d} && (6.42)\\
X &= \frac{Y_2}{Y_1} && (6.43)\\
R &= r_u + r_d - r_u r_d && (6.44)\\
P &= p_u + p_d - p_u p_d && (6.45)
\end{aligned}$$

With the solution of the internal states at hand, we turn our attention to the solution for the boundary states. We begin with the lower boundary states. There are a total of eight different lower boundary states. Two of these, $(0,1,0)$ and $(0,1,1)$, are transient states, leaving a total of six non-transient states. For these six states, we have six equations, given by Equations (6.1) through (6.5). We can write $\mathbf{p}(000)$ and $\mathbf{p}(110)$ in terms of $\mathbf{p}(001)$, as in Equations (6.30) and (6.31), leaving a total of four equations for four states. Remembering that state $(1,0,1)$ has internal form gives three equations for three states. For the sake of convenience, we define the following quantities.

$$\begin{aligned}
I_1 &= q_d(r_u - r_d(1 - p_u)) && (6.46)\\
J_1 &= q_d(1 - r_u - r_d p_u) && (6.47)
\end{aligned}$$

By rewriting and collecting terms for Equations (6.3), (6.4), and (6.6), we arrive at the following:

$$\begin{aligned}
(1 - r_u)(1 - p_d)CXY_2 &= \frac{r_u(J_1 + R)}{R}\mathbf{p}(001) - (1 - r_u)r_d\mathbf{p}(100) - p_u(1 - p_d)\mathbf{p}(111)\\
(1 - r_u)p_d CXY_2 &= -\frac{p_u(1 - r_d)r_u q_d}{R}\mathbf{p}(001) + R\mathbf{p}(100) - p_u p_d\mathbf{p}(111)
\end{aligned}$$

$$r_u(1-p_d)CXY_2 = -\frac{r_u(I_1+R)}{R}\mathbf{p}(001) - r_u r_d \mathbf{p}(100) + P\mathbf{p}(111)$$

We have three equations in three unknowns, for which we are guaranteed a unique solution. It is complicated, however, and we defer its exposition until Section 6.6.

By following a similar procedure as that described above, it is possible to proceed in exactly the same manner to find expressions for the upper boundary equations. If we define the following quantities

$$\begin{aligned} I_2 &= r_u(1+r_d^2-2r_d+r_d p_d) \\ J_2 &= X_2^{N-1}Y_1 \end{aligned}$$

and observing that the state $(N-1,1,0)$ is of internal form, the by making the substitutions defined in Equations (6.32) and (6.33), we arrive that the following equations for $\mathbf{p}(N-1,00)$, $\mathbf{p}(N-1,11)$ and $\mathbf{p}(N10)$.

$$\begin{aligned} (1-p_u)(1-r_d)CJ_2 &= \frac{q_u(1-r_d)R-I_2}{R}\mathbf{p}(N10) - r_u(1-r_d)\mathbf{p}(N-1,00) \\ &\quad -(1-p_u)p_d\mathbf{p}(N-1,11) \\ (1-p_u)r_dCJ_2 &= -(1-q_u)r_d - \frac{r_u r_d q_u(1-r_d) - r_u(1-p_d)q_u r_d}{R}\mathbf{p}(N10) \\ &\quad -r_u r_d\mathbf{p}(N-1,00) + P\mathbf{p}(N-1,11) \\ p_u(1-r_d)CJ_2 &= -\frac{(1-r_u)q_u p_d r_d}{R}\mathbf{p}(N10) + R\mathbf{p}(N-1,00) \\ &\quad -p_u p_d\mathbf{p}(N-1,11) \end{aligned}$$

Again, we have three equations in three unknowns, which yields a unique solution.

6.6 Summary of Steady-State Probabilities

The probability mass function for the Markov chain of the two-machine transfer line model are summarized as follows.

$$\mathbf{p}(0,0,0) = C\frac{(1-r_u)q_d}{r_u+r_d-r_u r_d}p_s \tag{6.48}$$

$$\mathbf{p}(0,1,0) = 0 \tag{6.49}$$

$$\mathbf{p}(0,0,1) = C\frac{bfl-bki+chk-cel+jei-jhf}{aei-ahf-dbi+hdc-edc+gbf} \tag{6.50}$$

$$\mathbf{p}(0,1,1) = 0 \tag{6.51}$$

$$\mathbf{p}(1,0,0) = -C\frac{afl - aki - dcl - fgj + kgc + dji}{aei - ahf - dbi + hdc - edc + gbf} \tag{6.52}$$

$$\mathbf{p}(1,1,0) = C\frac{r_u q_d}{r_u + r_d - r_u r_d} p_s \tag{6.53}$$

$$\mathbf{p}(1,0,1) = CXY_2 \tag{6.54}$$

$$\mathbf{p}(1,1,1) = C\frac{hdj - ahk + ael - egj - dbl + gbk}{aei - ahf - dbi + hdc - edc + gbf} \tag{6.55}$$

$$\mathbf{p}(N-1,0,0) = -C\frac{a'f'l' - a'k'i' - d'c'l' - f'g'j' + k'g'c' + d'j'i'}{a'e'i' - a'h'f' - d'b'i' + h'd'c' - e'd'c' + g'b'f'} \tag{6.56}$$

$$\mathbf{p}(N-1,0,1) = C\frac{q_u r_d}{r_u + r_d - r_u r_d} p_b \tag{6.57}$$

$$\mathbf{p}(N-1,1,0) = CX_2^{N-2} Y_2 \tag{6.58}$$

$$\mathbf{p}(N-1,1,1) = C\frac{h'd'j' - a'h'k' + a'e'l' - e'g'j' - d'b'l' + g'b'k'}{a'e'i' - a'h'f' - d'b'i' + h'd'c' - e'd'c' + g'b'f'} \tag{6.59}$$

$$\mathbf{p}(N,0,0) = C\frac{r_u(1 - r_d)}{r_u + r_d - r_u r_d} p_b \tag{6.60}$$

$$\mathbf{p}(N,1,0) = C\frac{b'f'l' - b'k'i' + c'h'k' - c'e'l' + j'e'i' - j'h'f'}{a'e'i' - a'h'f' - d'b'i' + h'd'c' - e'd'c' + g'b'f'} \tag{6.61}$$

$$\mathbf{p}(N,0,1) = 0 \tag{6.62}$$

$$\mathbf{p}(N,1,1) = 0 \tag{6.63}$$

$$\mathbf{p}(n,\alpha_u,\alpha_d) = CX^n Y_1^{\alpha_u} Y_2^{\alpha_d}, 2 \le n \le N-2; \alpha_u = 0,1; \alpha_d = 0,1 \tag{6.64}$$

where Y_1, Y_2, X, R, and P are given by Equations (6.41), (6.42), (6.43), (6.44), and (6.45), the normalizing constant C is given by

$$C = \left(\sum_{\alpha_u=0}^{1} \sum_{\alpha_d=0}^{1} \sum_{n=0}^{N} \mathbf{p}(n,\alpha_u,\alpha_d) \right)^{-1}, \tag{6.65}$$

and the values for $a, a', b, b', \ldots$ are given in Table I.

$a = \frac{r_u(J+R)}{R}$	$b = -(1-r_u)r_d$	$c = -p_u(1-p_d)$
$d = \frac{-p_u(1-r_d)r_u q_d}{R}$	$e = R$	$f = -p_u p_d$
$g = \frac{-r_u(I+R)}{R}$	$h = -r_u r_d$	$i = P$
$j = (1-r_u)(1-p_d)XY_2$	$k = (1-r_u)p_d XY_2$	$l = r_u(1-p_d)XY_2$
$I = q_d((1-p_u)r_d - r_u)$	$J = q_d(1-r_u-p_u r_d)$	
$a' = \frac{q_u((1-r_d)R-I')+r_d R}{R}$	$b' = -r_u(1-r_d)$	$c' = -(1-p_u)p_d$
$d' = \frac{-(1-q_u)r_d R - q_u r_u r_d(1-r_d) - q_u r_u r_d(1-p_d)}{R}$	$e' = -r_u r_d$	$f' = P$
$g' = \frac{-(1-r_u)q_u r_d p_d}{R}$	$h' = R$	$i' = -p_u p_d$
$j = (1-r_u)(1-p_d)XY_2$	$k = (1-r_u)p_d XY_2$	$l = r_u(1-p_d)XY_2$
$I' = r_u(1+r_d^2-2r_d+r_d p_d)$		

Table I: Values in PMF

Chapter 7

Decomposition Derivation

7.1 Introduction

Following a similar approach as the decomposition for the single part type case, we present a decomposition for the processing line with two part types. As before, we take the position of an observer in a buffer. We then seek to capture the upstream and downstream behavior, as seen by the observer, in the parameters of a two machine line. As before, we will exploit the relationships inherent in the model, namely conservation of flow and the flow rate-idle time relationship.

7.2 Notation

The decomposition of the two-part type line is illustrated in Figure 7-1. For each buffer $B_{i,j}$, we have a representative two-machine line, denoted $L(i,j)$. The upstream machine is denoted $M_u(i,j)$, and the downstream machine is denoted $M_d(i,j)$. We denote the repair, failure, and idleness failure parameters of $M_u(i,j)$ by $r_u(i,j)$, $p_u(i,j)$ and $q_u(i,j)$, respectively. Likewise, we denote the repair, failure, and idleness failure parameters of $M_d(i,j)$ by $r_d(i,j)$, $p_d(i,j)$ and $q_d(i,j)$, respectively. We take the size of the buffer $B(i,j)$ in $L(i,j)$, denoted $N(i,j)$, to be the same as $B_{i,j}$. We denote the current level of the buffer in $L(i,j)$ by $n(i,j)(t)$.

For a long line with K processing machines, we will have $2(K+1)$ buffers, and thus $2(K+1)$ two-machine lines in the decomposition. Each two-machine line has seven parameters that determine its behavior. These are the six machine parameters, plus the size of the intermediate buffer. Since we have fixed the size of the intermediate buffer of the two-machine line, we are left with six unknown parameters for each two-machine line to be determined in the decomposition. Thus, we have $12(K+1)$ unknowns, and will need $12(K+1)$ parameters to guarantee a unique solution.

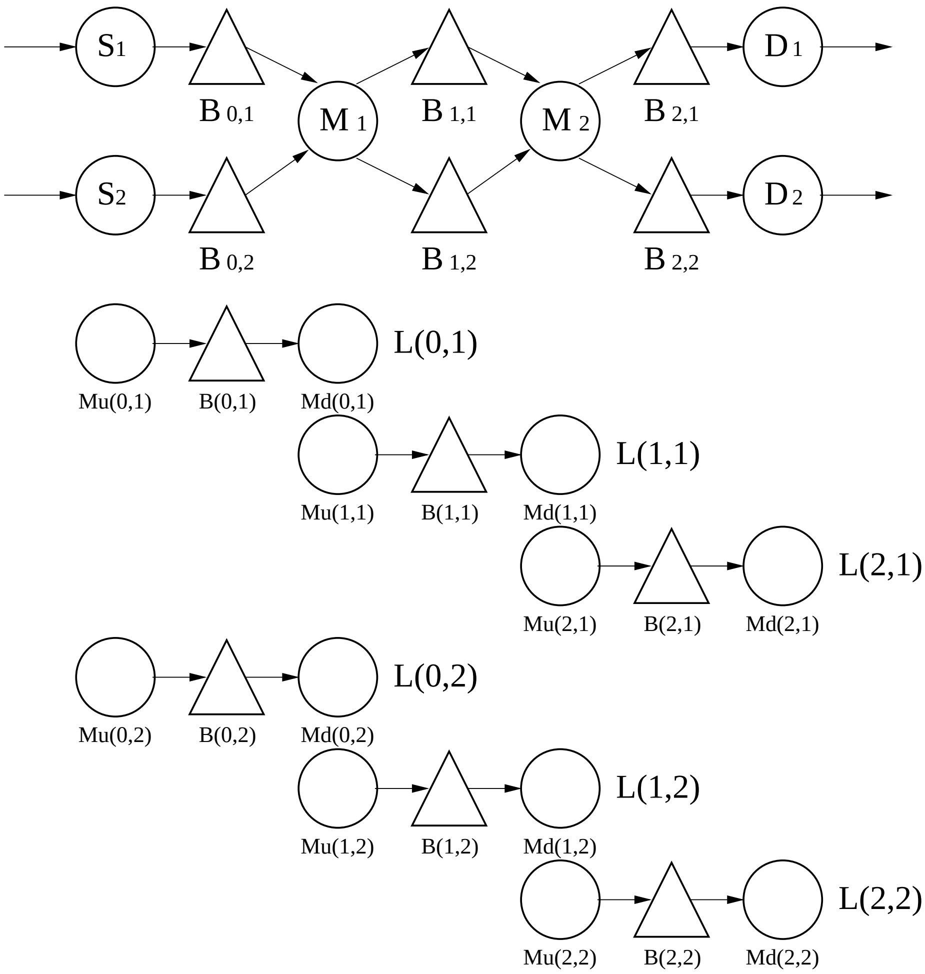

Figure 7-1: Decomposition of a two-part type processing line, with two processing machines.

7.3 An important approximation

chap:derivation:sec:approximation

In order to facilitate the derivation of the decomposition, we need to make a crucial approximation. We will assume that the probability that a machine M_i is simultaneously starved and blocked for a given part is negligible. That is, we assume that

Approximation. $Pr\left[n_{i-1,j}(t) = 0 \cap n_{i,j}(t) = N_{i,j}\right] \approx 0.$ □

We justify this approximation with the following argument. In order for a machine to be both starved and blocked for a part simultaneously, it is necessary that at some point, the machine had exactly one part in the upstream buffer, and exactly one space in the downstream buffer. We expect that this is a rare occurrence because in order for it to happen, the buffer upstream of machine M_i would have to have exactly one part in it, the buffer downstream of machine M_i would have to have exactly one space for a part. Furthermore, the upstream machine must be unable to process parts to place in its downstream buffer, and the downstream machine must be unable to process parts, depleting the stores of its upstream buffer. Since the probability of a machine failing is assumed to be small — on the order of one percent — that probability that all three of these occurrences happen at the same time is likely to be quite low. In fact, testing this hypothesis using discrete event simulation has shown that the approximation holds in many systems with moderate sized buffers.

7.4 Conservation of Flow

As in the case of the single part type decomposition, we require that flow be conserved, since there is no mechanism for the creation or destruction of material. In the case with two part types, we expect the flow of each type to be conserved. That is, for $j = 1, 2$, we require that

$$E(i,j) = E(0,j) \quad \text{for } i = 1, \ldots, K. \tag{7.1}$$

Equation (7.1) gives $2K$ equations.

7.5 Resumption of Flow Equations, Type One

The resumption of flow equations are used to calculate the values of $r_u(i,j)$ for $j = 1, 2$ and $i = 1, \ldots, K$, giving $2K$ equations, and $r_d(i,j)$ for $j = 1, 2$ and $i = 0, \ldots, K-1$, giving another $2K$ equations, for a total of $4K$ equations. We begin with the calculation of an expression for $r_u(i, 1)$, $i = 1, \ldots, K$.

It is important to understand the role played by the resumption of flow equations. The purpose of decomposing a long processing line into sub-lines is that the Markov chain model for the long line is not tractable — either analytically or numerically — while that of the two-machine sub-line is. The idea is then to set the parameters of the upstream and downstream pseudo-machines of the sub-line to capture the behavior of the long line upstream and downstream of the buffer being modeled. Therefore, the upstream pseudo-machine in a sub-line is meant to capture the nature of flow of material into the buffer from the line upstream. The resumption of flow equation of the upstream pseudo-machine then needs to be a representation of this behavior. To illustrate this, we take the case of the deterministic long line decomposition for a processing line with a single part type, given in Gershwin [16]. In that case, flow of material from upstream of a buffer could have stopped for one of two reasons: either the upstream machine failed, or the upstream machine became starved for material, and ceased producing parts. Therefore, the repair probability of the upstream pseudo-machine for the sub-line capturing the flow behavior would be a weighted sum of the probability that the upstream machine was repaired, given that it was down, plus the probability that the machine became unstarved, given that it was starved. Likewise, the repair probability of the downstream pseudo-machine for the same sub-line would be the weighted sum of the probability that the downstream machine was repaired, given that it was down, plus the probability that the downstream machine became unblocked, given that it was blocked.

In the case of the deterministic long line decomposition given in Gershwin (1995), the resumption of flow equations are

$$\begin{aligned} r_u(i) &= r_i\left(1 - p_s(i-1)\frac{r_u(i)}{p_u(i)E(i)}\right) \\ &\quad + r_u(i-1)p_s(i-1)\frac{r_u(i)}{p_u(i)E(i)} \end{aligned} \tag{7.2}$$

$$r_d(i-1) = r_i\left(1 - p_b(i)\frac{r_d(i)}{p_d(i)E(i)}\right) + r_d(i)p_b(i)\frac{r_d(i)}{p_d(i)E(i)} \tag{7.3}$$

Here, we have suppressed the notational dependence on part type, as there is only one. The resumption of flow equations we derive for the multiple part type case will be of similar spirit; namely, they will be weighted sums of the probabilities of transition out of a certain state, given the system was in this state. In this case, such a state will be any reason that flow is not entering a given buffer.

7.5.1 Some Useful Quantities

The main technique in deriving the decomposition is to relate events in the real line with events in the two-machine sub-lines. Since the two-machine sub-lines model the behavior of material flowing into and out of the buffers in the real line, the states when buffers in the real line are empty or full should correspond to states in the relevant two-machine sub-lines where the intermediate buffer is empty or full. We begin by associating the state $(0, 0, 0)$ in the two-machine line $L(i-1, 1)$ with a state in the processing line. The empty buffer $B_{i-1,1}$ is captured in the buffer of the two-machine line being empty. The downstream machine of the two-machine line being down can be due to only two possible events: either the real machine is up and blocked, or the real machine is down. Since $B_{i-1,1}$ is empty, we cannot have $B_{i,1}$ full, as this violates our assumption that no machine is simultaneously blocked and starved. Finally, the upstream machine being down is due to the fact that the state $(0, 1, 0)$ in the two-machine line solution is transient. So we have

$$\mathbf{p}(i-1, 1, ; 000) = Pr\left[\alpha_i(t) = 0 \cap n_{i-1,1}(t-1) = 0 \cap n_{i,1}(t-1) < N_{i,1}\right]. \tag{7.4}$$

Following a similar line of reasoning leads to the following definitions.

$$\begin{aligned}
\mathbf{p}(i-1,1;001) &= Pr\left[\alpha_i(t)=1 \cap n_{i-1,1}(t-1)=0 \cap n_{i,1}(t-1)<N_{i,1}\right] && (7.5)\\
\mathbf{p}(i,1;N_{i,1},00) &= Pr\left[\alpha_i(t)=0 \cap n_{i-1,1}(t-1)>0 \cap n_{i,1}(t-1)=N_{i,1}\right] && (7.6)\\
\mathbf{p}(i,1;N_{i,1},10) &= Pr\left[\alpha_i(t)=1 \cap n_{i-1,1}(t-1)>0 \cap n_{i,1}(t-1)=N_{i,1}\right] && (7.7)
\end{aligned}$$

We note that the quantities in Equations (7.5) and (7.7) correspond to $p_s(i-1,1)$ and $p_b(i,1)$, respectively. That is, $\mathbf{p}(i-1,1;001)$ is precisely the state in which the downstream machine in line $L(i-1,1)$ is starved, and $\mathbf{p}(i,1;N_{i,1},10)$ is precisely the state in which the upstream machine in line $L(i,1)$ is blocked.

7.5.2 Type One, Upstream

The parameter $r_u(i,1)$ is the repair probability of the upstream machine of the two-machine sub-line $L(i,1)$. That is, it is the probability that $M_u(i,1)$ makes a part in time step $t+1$, given it did not make one, and was not blocked, in time step t. There are three ways that M_i could have not produced a type one part in time step t, given that it was not blocked for type one at time $t-1$. They are:

- M_i was down at time t, and not starved for type one at time $t-1$,
- M_i was down at time t, and starved for type one at time $t-1$, or
- M_i was up at time t, and starved for type one at time $t-1$

Observe that the first and the third points are identical to those in the discrete time, discrete material transfer line decomposition given in Gershwin [16], while the second point is new. If we are considering a processing line making only one part type, and assume the the machines have operation-dependent failures, then the probability of the second point occurring is zero, as the a machine being simultaneously down and starved implies that the machine failed while idle.

Written in terms of the long-line parameters, this is gives the expression

$$\begin{aligned}
r_u(i,1) = Pr\big[&\alpha_i(t+1)=1 \cap n_{i-1,1}(t)>0 \cap n_{i,1}(t-1)<N_{i,1} \big| \\
&(\{\alpha_i(t)=0 \cap n_{i-1,1}(t-1)>0\} \cup \\
&\{\alpha_i(t)=0 \cap n_{i-1,1}(t-1)=0\} \cup \\
&\{\alpha_i(t)=1 \cap n_{i-1,1}(t-1)=0\}) \cap n_{i,1}(t-1)<N_{i,1}\big]
\end{aligned}$$

If we define the following events

$$U = \{\alpha_i(t+1)=1 \cap n_{i-1,1}(t)>0 \cap n_{i,1}(t-1)<N_{i,1}\}$$

$$\begin{aligned} X &= \{\alpha_i(t) = 0 \cap n_{i-1,1}(t-1) > 0 \cap n_{i,1}(t-1) < N_{i,1}\} \\ Y &= \{\alpha_i(t) = 0 \cap n_{i-1,1}(t-1) = 0 \cap n_{i,1}(t-1) < N_{i,1}\} \\ Z &= \{\alpha_i(t) = 1 \cap n_{i-1,1}(t-1) = 0 \cap n_{i,1}(t-1) < N_{i,1}\} \end{aligned}$$

We note that the events X, Y, and Z are mutually exclusive. Then we can rewrite $r_u(i,1)$ as

$$\begin{aligned} r_u(i,1) &= Pr\,[U|X \cup Y \cup Z] \\ &= Pr\,[U|X]Pr\,[X|X \cup Y \cup Z] + Pr\,[U|Y]Pr\,[Y|X \cup Y \cup Z] \\ &\quad + Pr\,[U|Z]Pr\,[Z|X \cup Y \cup Z] \\ &= \frac{Pr\,[U|X]Pr\,[X] + Pr\,[U|Y]Pr\,[Y] + Pr\,[U|Z]Pr\,[Z]}{Pr\,[X \cup Y \cup Z]} \end{aligned}$$

We need to calculate these quantities. We begin with the conditional probability that we make a part at the end of time step $t+1$, given that we did not make one at the end of time step t because M_i was down and not blocked or starved for type ones.

$$\begin{aligned} Pr\,[U|X] &= Pr\,[\alpha_i(t+1) = 1 \cap n_{i-1,1}(t) > 0 \cap n_{i,1}(t-1) < N_{i,1}| \\ &\qquad \alpha_i(t) = 0 \cap n_{i-1,1}(t-1) > 0 \cap n_{i,1}(t-1) < N_{i,1}] \\ &= Pr\,[\alpha_i(t+1) = 1|\alpha_i(t) = 0] \\ &= r_i. \end{aligned}$$

Here, the last step follows from the definition of r_i.

Next, we calculate the conditional probability that M_i makes a type one part in step $t+1$, given it did not make one in step t because the machine was down at the end of time t, and starved for type one parts in step $t-1$. This is the probability that M_i is repaired, and M_{i-1} makes a type one part.

$$\begin{aligned} Pr\,[U|Y] &= Pr\,[\alpha_i(t+1) = 1 \cap n_{i-1,1}(t) > 0 \cap n_{i,1}(t-1) < N_{i,1}| \\ &\qquad \alpha_i(t) = 0 \cap n_{i-1,1}(t-1) = 0 \cap n_{i,1}(t-1) < N_{i,1}] \\ &\approx r_i r_u(i-1,1). \end{aligned}$$

Here, the approximation comes from two events occurring. First, machine M_i has been repaired, which occurs with probability r_i. Second, buffer $B_{i-1,1}$ has had a part delivered to it, so that it ceases to be empty. That is, an observer in the buffer has seen flow resume. This could be due to a repair of M_{i-1}, or because M_{i-1} has become unstarved for Type 1 parts due to a repair upstream. The observer is unable to distinguish which event occurred, but the probability of flow resuming is given by $R_u(i-1,1)$.

Next, we calculate the conditional probability that M_i makes a type one part in step $t+1$, given it did not make one in step t because it was up, but starved for type one parts.

$$\begin{aligned} Pr\,[U|Z] &= Pr\,[\alpha_i(t+1) = 1 \cap n_{i-1,1}(t) > 0 \cap n_{i,1}(t-1) < N_{i,1}| \\ &\qquad \alpha_i(t) = 1 \cap n_{i-1,1}(t-1) = 0 \cap n_{i,1}(t-1) < N_{i,1}] \\ &\approx r_u(i-1,1)(1-q_d(i-1,1)). \end{aligned}$$

In this derivation, two events have occurred simultaneously. First, flow has resumed into the upstream buffer. This occurs with probability $r_u(i-1,1)$. Second, machine M_i did not fail while idle. This corresponds to the machine not failing while making a Type 2 part. The observer in the Type 1 buffer does not even know if M_i was working on a Type 2 part, but he observes that the machine did not undergo an idleness failure. This occurs with probability $1-q_d(i-1,1)$.

Next we calculate the probability of the event X. That is, we wish to calculate the probability that M_i is down, and not blocked or starved. In order to do so, we need to classify exactly when the upstream machine of the sub-line is down and not blocked. There are three mutually exclusive, collectively exhaustive possibilities:

- M_i is down, and not blocked or starved for type one,
- M_i is down, and starved, but not blocked for type one, or
- M_i is up, and starved, but not blocked for type one.

In other words,

$$\begin{aligned} \{\alpha_u(i,1) = 0 \cap n_{i,1} < N_{i,1}\} &\iff \\ \{\alpha_i(t) = 0 \cap n_{i-1,1}(t-1) > 0 \cap n_{i,1}(t-1) < N_{i,1}\} & \\ \cup\, \{\alpha_i(t) = 0 \cap n_{i-1,1}(t-1) = 0 \cap n_{i,1}(t-1) < N_{i,1}\} & \\ \cup\, \{\alpha_i(t) = 1 \cap n_{i-1,1}(t-1) = 0 \cap n_{i,1}(t-1) < N_{i,1}\} &\iff \\ \{\alpha_i(t) = 0 \cap n_{i,1}(t-1) < N_{i,1}\} & \\ \cup\, \{\alpha_i(t) = 1 \cap n_{i-1,1}(t-1) = 0 \cap n_{i,1}(t-1) < N_{i,1}\} & \end{aligned} \tag{7.8}$$

Since these are disjoint events, it follows from Equations (7.4) and (7.5) that

$$Pr\,[X] = Pr\,[\alpha_u(i,1) = 0 \cap n_{i,1} < N_{i,1}] - \mathbf{p}(i-1,1;000) - p_s(i-1,1)$$

Now, from the repair-frequency-equals-failure-frequency property of the upstream machine for the two machine line (Equation (6.25)), this can be written as

$$\begin{aligned} Pr\,[X] &= \frac{p_u(i,1)}{r_u(i,1)} Pr\,[\alpha_u(i,1) = 1 \cap n_{i,1} < N_{i,1}] + \frac{q_u(i,1)}{r_u(i,1)} p_b(i,1) \\ &\quad -\mathbf{p}(i,1;N_{i,1},00) - \mathbf{p}(i-1,1;000) - p_s(i-1,1) \end{aligned}$$

Set

$$I_u(i,1) = \frac{p_u(i,1)}{r_u(i,1)} \tag{7.9}$$

Then using Equation (6.28), we have the final expression for $Pr\,[X]$.

$$\begin{aligned} Pr\,[X] &= I_u(i,1)\,[E(i,1) - q_u(i,1)p_b(i,1) + r_u(i,1)\mathbf{p}(i,1;N_{i,1},00)] \\ &\quad + \frac{q_u(i,1)}{r_u(i,1)} p_b(i,1) - \mathbf{p}(i,1;N_{i,1},00) \\ &\quad -\mathbf{p}(i-1,1;000) - p_s(i-1,1) \end{aligned} \tag{7.10}$$

In a similar manner, it is possible to calculate $Pr\,[Y]$.

$$\begin{aligned} Pr\,[Y] &= Pr\,[\alpha_i(t) = 0 \cap n_{i-1,1}(t-1) = 0 \cap n_{i,1}(t-1) < N_{i,1}] \\ &= Pr\,[n_{i-1,1}(t-1) = 0 \cap n_{i,1}(t-1) < N_{i,1}] \\ &\quad -Pr\,[\alpha_i(t) = 1 \cap n_{i-1,1}(t-1) = 0 \cap n_{i,1}(t-1) < N_{i,1}] \\ &= Pr\,[n_{i-1,1}(t-1) = 0] - p_s(i-1,1) \\ &= p_s(i-1,1) + \mathbf{p}(i-1,1;000) - p_s(i-1,1) \\ &= \mathbf{p}(i-1,1;000) \end{aligned} \tag{7.11}$$

where the third equality follows from Assumption 1. The calculation of $Pr\,[Z]$ is identical.

$$\begin{aligned} Pr\,[Z] &= Pr\,[\alpha_i(t) = 1 \cap n_{i-1,1}(t-1) = 0 \cap n_{i,1}(t-1) < N_{i,1}] \\ &= Pr\,[n_{i-1,1}(t-1) = 0 \cap n_{i,1}(t-1) < N_{i,1}] \\ &\quad -Pr\,[\alpha_i(t) = 0 \cap n_{i-1,1}(t-1) = 0 \cap n_{i,1}(t-1) < N_{i,1}] \\ &= Pr\,[n_{i-1,1}(t-1) = 0] - \mathbf{p}(i-1,1;000) \\ &= p_s(i-1,1) + \mathbf{p}(i-1,1;000) - \mathbf{p}(i-1,1;000) \\ &= p_s(i-1,1) \end{aligned}$$

(7.12)

Finally, it remains to calculate $Pr\left[X \cup Y \cup Z\right]$. There are two ways to do this. The first is to observe that X, Y, and Z are disjoint events, so the probability of their unions is the sum of their probabilities. The second is to derive it from first principles. We choose the latter as a check on our work. Observe that

$$\{X \cup Y \cup Z\} \iff \{\alpha_u(i,1) = 0 \cap n < N\}$$

Then

$$\begin{aligned}
Pr\left[X \cup Y \cup Z\right] &= Pr\left[\alpha_u(i,1) = 0 \cap n < N\right] \\
&= I_u(i,1) Pr\left[\alpha_u(i,1) = 1 \cap n < N\right] + \frac{q_u(i,1)}{r_u(i,1)} p_b(i,1) \\
&\quad - \mathbf{p}(i,1;N_{i,1},00) \\
&= I_u(i,1)[E(i,1) - q_u(i,1)p_b(i,1) + r_u(i,1)\mathbf{p}(i,1;N_{i,1},00)] \\
&\quad + \frac{q_u(i,1)}{r_u(i,1)} p_b(i,1) - \mathbf{p}(i,1;N_{i,1},00)
\end{aligned} \tag{7.13}$$

where the last equality follows from Equation (6.25). Observe that this quantity is precisely the sum of the probabilities of the events X, Y, and Z.

Finally, we arrive at the following expression for $r_u(i,1)$:

$$\begin{aligned}
r_u(i,1) &= \Bigg(r_i \Bigg[\frac{p_u(i,1)}{r_u(i,1)} \left\{ E(i-1,1) - q_u(i,1)p_b(i,1) + r_u(i,1)\mathbf{p}(i,1;N00) \right\} \\
&\quad + \frac{q_u(i,1)}{r_u(i,1)} p_b(i,1) - \mathbf{p}(i,1;N_{i,1},00) - \mathbf{p}(i-1,1;000) - p_s(i-1,1) \Bigg] \\
&\quad + r_i r_u(i-1,1) p_s(i-1,1) \\
&\quad + r_u(i-1,1)(1 - q_d(i-1,1))\mathbf{p}(i-1,1;000) \Bigg) \\
&\div \Bigg(\frac{p_u(i,1)}{r_u(i,1)} \left\{ E(i,1) - q_u(i,1)p_b(i,1) + r_u(i,1)\mathbf{p}(i,1;N_{i,1},00) \right\} \\
&\quad + \frac{q_u(i,1)}{r_u(i,1)} p_b(i,1) - \mathbf{p}(i,1;N_{i,1},00) \Bigg)
\end{aligned}$$

We observe that if we set $q_u(i,1) = q_d(i-1,1) = 0$, then $\mathbf{p}(i,1;N00)$ and $\mathbf{p}(i-1,1;000)$ both become zero. Then $r_u(i,1)$ becomes

$$\begin{aligned} r_u(i,1) &= r_i\left(\frac{p_u(i,1)}{r_u(i,1)}\{E(i-1,1) - p_s(i-1,1)\} + r_u(i-1,1)p_s(i-1,1)\right) \\ &\div \frac{p_u(i,1)}{r_u(i,1)}E(i,1). \end{aligned}$$

Observing that Conservation of Flow gives $E(i-1,1) = E(i,1)$, this equation is reduced to the resumption of flow equation for the one-part transfer line decomposition given in [16]. That is, if we no longer allow the two-machine lines to experience idleness failures, we are reduced to the one-part type case. This is consistent with the priority scheduling policy for determination of which part type to produce. That is, the machines will always produce a Type 1 part if they are able. Therefore, the only way in which the existence of a Type 2 part makes an appearance in the production of Type 1 parts is if a machine that was either blocked or starved for a Type 1 part (a necessary condition for it to be producing a Type 2 part) failed while making a Type 2 part.

Finally, we can derive an expression for $r_d(i,1)$. This calculation involves only minor technical differences from the derivation of $r_u(i,1)$, and the details are relegated to Section B.2 of Appendix B. The equation for $r_d(i,1)$ is given as:

$$\begin{aligned} r_d(i,1) &= \Bigg(r_{i+1}\left[\frac{p_d(i,1)}{r_d(i,1)}\{E(i+1,1) - q_d(i,1)p_s(i,1) + r_d(i,1)\mathbf{p}(i,1;000)\}\right. \\ &\quad \left. + \frac{q_d(i,1)}{r_d(i,1)}p_s(i,1) - \mathbf{p}(i,1;N00) - \mathbf{p}(i,1;000) - p_b(i,1)\right] \\ &\quad + r_{i+1}r_d(i+1,1)p_b(i+1,1) \\ &\quad + r_d(i+1,1)(1-q_u(i+1,1))\mathbf{p}(i+1,1;N00)\Bigg) \\ &\div \left(\frac{p_d(i,1)}{r_d(i,1)}\{E(i+1,1) - q_d(i,1)p_s(i,1) + r_d(i,1)\mathbf{p}(i,1;000)\}\right. \\ &\quad \left. + \frac{q_d(i,1)}{r_d(i,1)}p_s(i,1) - \mathbf{p}(i,1;000)\right). \end{aligned}$$

We note that this equation also reduces to the one-part type resumption of flow equation for the downstream machine if the idleness failure probabilities are set to zero.

7.6 Resumption of Flow Equations, Type Two

Having derived the resumption of flow equations for Type 1 parts, we next do the same for Type 2 parts. This derivation is much more complicated than the one done for Type 1 parts for the following reason. While the existence of Type 2 parts was made apparent to Type 1 parts only through the existence of idleness failures — which required only a minor modification of the resumption of flow equations — the existence of a Type 1 part has a major influence on the production of Type 2 parts. In particular, we now have to account for the possibility that an observer in a Type 2 buffer will see flow into his buffer cease because the upstream machine switched from making Type 2 to Type 1 parts. Furthermore, we still have to account for idleness failures, which can occur if, for example, a Type 2 buffer fills up, and the machine begins to produce Type 1 parts while the buffer is still full. We begin our analysis by defining some quantities that will be useful in the derivation.

7.6.1 Some useful identities

Before we begin with the derivation of the resumption of flow equations for the second part type, we will find it useful to derive some identities, which will be important for the derivation of certain quantities used later. To simplify the notation in what follows, we define the following four mutually-exclusive events:

$$\begin{aligned}
A &= \{n_{i-1,1} > 0 \cap n_{i,1} < N_{i,1}\} \\
A^c &= \{n_{i-1,1} = 0 \cup n_{i,1} = N_{i,1}\} \\
B &= \{n_{i-1,2} > 0 \cap n_{i,2} < N_{i,2}\} \\
B^c &= \{n_{i-1,2} = 0 \cup n_{i,2} = N_{i,2}\}
\end{aligned}$$

Our task now is to relate states of the real line with states in the corresponding two-machine lines of the decomposition. We begin by consider the event $(0,0,1)$ for line $L(i-1,2)$. This is the event that the buffer is empty, the upstream machine is down, and the downstream machine is up. We first observe that a necessary condition for the upstream machine to be up is that machine M_i is up, and either blocked or starved for Type 1 parts. Since the buffer $B_{i-1,2}$ is empty, our assumption that machines cannot be blocked or starved simultaneously leads immediately to the knowledge that buffer $B_{i,2}$ cannot be full. Finally, since the event $(0,1,0)$ in the two-machine line is transient, we observe that we do not have to take the state of machine M_{i-1} into account. We are then justified in setting

$$p_s(i-1,2) = Pr\,[\alpha_i(t) = 1 \cap A^c \cap n_{i-1,2} = 0 \cap n_{i,2} < N_{i,2}]. \tag{7.14}$$

Following a similar argument, we can set

$$p_b(i,2) = Pr\,[\alpha_i(t) = 1 \cap A^c \cap n_{i-1,2} > 0 \cap n_{i,2} = N_{i,2}], \tag{7.15}$$

and

$$\mathbf{p}(i-1,2;000) = Pr\,[A \cap n_{i-1,2} = 0 \cap n_{i,2} < N_{i,2}] \tag{7.16}$$
$$\mathbf{p}(i,2;N00) = Pr\,[A \cap n_{i-1,2} > 0 \cap n_{i,2} = N_{i,2}] \tag{7.17}$$

Now, observe that it follows from Equations (7.14) and (7.15)

$$Pr\,[\alpha_i(t) = 1 \cap A^c \cap B^c] = p_s(i-1,2) + p_b(i,2)$$

However, by our assumption that the machines in the real line have operation dependent failures, it follows immediately that

$$Pr\,[\alpha_i(t) = 0 \cap A^c \cap B^c] = 0,$$

because in this combination of states, the real machine is down and either blocked or starved for both part type one and part type two. Since this is impossible, as it would imply that the machine failed while idle, the probability of this combination of states is zero. Therefore we have

$$Pr\,[A^c \cap B^c] = p_s(i-1,2) + p_b(i,2).$$

Next, observe that Equations (7.16) and (7.17) imply that

$$Pr\,[A \cap B^c] = \mathbf{p}(i-1,2;000) + \mathbf{p}(i,2;N00).$$

By taking the union of $\{A^c \cap B^c\}$ and $\{A \cap B^c\}$, we immediately get that

$$Pr\,[B^c] = p_s(i-1,2) + p_b(i,2) + \mathbf{p}(i-1,2;000) + \mathbf{p}(i,2;N00).$$

It follows, of course, that

$$Pr\,[B] = 1 - p_s(i-1,2) - p_b(i,2) - \mathbf{p}(i-1,2;000) - \mathbf{p}(i,2;N00).$$

Furthermore, by exactly the same reasoning, we have

$$Pr\,[A^c] = p_s(i-1,1) + p_b(i,1) + \mathbf{p}(i-1,1;000) + \mathbf{p}(i,1;N00),$$

and

$$Pr\,[A] = 1 - p_s(i-1,1) - p_b(i,1) - \mathbf{p}(i-1,1;000) - \mathbf{p}(i,1;N00). \tag{7.18}$$

Now, since the events are disjoint, we have that

$$Pr\,[A] = Pr\,[A \cap B] + Pr\,[A \cap B^c].$$

Using this, we arrive at

$$\begin{aligned} Pr\,[A \cap B] &= 1 - p_s(i-1,1) - p_b(i,1) - \mathbf{p}(i-1,1;000) - \mathbf{p}(i,1;N00) \\ &\quad -\mathbf{p}(i-1,2;000) - \mathbf{p}(i,2;N00). \end{aligned}$$

Finally, taking the identity

$$Pr\,[A^c \cap B] = Pr\,[B] - Pr\,[A \cap B],$$

We find that

$$\begin{aligned} Pr\,[A^c \cap B] &= p_s(i-1,1) + p_b(i,1) + \mathbf{p}(i-1,1;000) + \mathbf{p}(i,1;N00) \\ &\quad -p_s(i-1,2) - p_b(i,2) \end{aligned} \tag{7.19}$$

To summarize, we have the following identities.

$$\begin{aligned} Pr\,[A \cap B] &= 1 - p_s(i-1,1) - p_b(i,1) - \mathbf{p}(i-1,1;000) \\ &\quad -\mathbf{p}(i,1;N00) - \mathbf{p}(i-1,2;000) - \mathbf{p}(i,2;N00) \end{aligned} \tag{7.20}$$

$$\begin{aligned} Pr\,[A^c \cap B] &= p_s(i-1,1) + p_b(i,1) + \mathbf{p}(i-1,1;000) + \mathbf{p}(i,1;N00) \\ &\quad -p_s(i-1,2) - p_b(i,2) \end{aligned} \tag{7.21}$$

$$Pr\,[A \cap B^c] = \mathbf{p}(i-1,2;000) + \mathbf{p}(i,2;N00) \tag{7.22}$$

$$Pr\,[A^c \cap B^c] = p_s(i-1,2) + p_b(i,2) \tag{7.23}$$

Next, we consider the following argument. We know that $p_s(i-1,2)$ is the probability that machine M_i is up, either blocked or starved for part type one, and starved for part type two. The probability that M_i is starved for both type one and type two is the probability that it is either starved or blocked for type one, and starved for type two, times the probability that it is starved for type one, given that it is either blocked or starved for type two.

This argument leads to the following approximation.

$$Pr\{\alpha_i(t) = 1 \cap n_{i-1,1}(t-1) = 0 \cap n_{i,1}(t-1) < N_{i,1} \tag{7.24}$$

$$\cap\ n_{i-1,2}(t-1) = 0 \cap n_{i,2}(t-1) < N_{i,2}\}$$
$$\approx p_s(i-1,2)\frac{p_s(i-1,1)}{p_s(i-1,1)+p_b(i,1)}$$

In a similar manner, we can derive

$$\begin{aligned} Pr\{\alpha_i(t) = 1 \cap n_{i-1,1}(t-1) > 0 \cap n_{i,1}(t-1) = N_{i,1} \\ \cap\ n_{i-1,2}(t-1) = 0 \cap n_{i,2}(t-1) < N_{i,2}\} \\ \approx p_s(i-1,2)\frac{p_b(i,1)}{p_s(i-1,1)+p_b(i,1)}, \end{aligned} \tag{7.25}$$

$$\begin{aligned} Pr\{\alpha_i(t) = 1 \cap n_{i-1,1}(t-1) = 0 \cap n_{i,1}(t-1) < N_{i,1} \\ \cap\ n_{i-1,2}(t-1) > 0 \cap n_{i,2}(t-1) = N_{i,2}\} \\ \approx p_b(i,2)\frac{p_s(i-1,1)}{p_s(i-1,1)+p_b(i,1)}, \end{aligned} \tag{7.26}$$

and

$$\begin{aligned} Pr\{\alpha_i(t) = 1 \cap n_{i-1,1}(t-1) > 0 \cap n_{i,1}(t-1) = N_{i,1} \\ \cap\ n_{i-1,2}(t-1) > 0 \cap n_{i,2}(t-1) = N_{i,2}\} \\ \approx p_b(i,2)\frac{p_b(i,1)}{p_s(i-1,1)+p_b(i,1)} \end{aligned} \tag{7.27}$$

With these identities in mind, we proceed with the derivation of the resumption of flow equations for part type two.

7.6.2 Type Two, Upstream

The parameter $r_u(i,2)$ is the repair probability of the upstream machine of the two-machine sub-line $L(i,2)$. That is, it is the probability that $M_u(i,2)$ makes a part in time step $t+1$, given it did not make one, and was not blocked, in time step t. This is approximately the same as the probability that M_i makes a type two in step $t+1$, given it did not make type two, and was not blocked for type two in step t. There are several states in which M_i does not produce a type two part in time step t, given that it was not blocked for type two at time $t-1$, but can produce a type two in time step $t+1$. These states and can be broken into the following three categories:

(a) M_i is up at time t, starved or blocked for type one, and starved for type two at time $t-1$.

(b) M_i is up at time t, and producing a type one, but is about to become either blocked or starved for type one, after which it will produce a type two.

(c) M_i is down at time t, and will produce a type two when it comes up.

Category (a) corresponds to the machine M_i being idle. We make the assumption that M_i is not blocked for type two. Then for the machine to be idle, we need it to be starved for type two, and either blocked or starved for type one. But we are interested only in states where the machine can make a type two in the next time step. In particular, we are interested in the following event:

$$I \;=\; \{\alpha_i(t) = 1 \cap A^c \cap n_{i-1,2}(t-1) = 0 \cap n_{i,2}(t-1) < N_{i,2}\}$$

Next we consider the the states that represent M_i making a type one part in step t, and a type two in step $t+1$, as described in category (b). This can happen only if the machine is about to become blocked or starved for type one. These events are denoted as follows.

$$\begin{aligned} V_1 &= \{\alpha_i(t) = 1 \cap (n_{i-1,1}(t-1) = 1 \cup n_{i,1}(t-1) = N_{i,1} - 1) \\ &\qquad \cap\, n_{i-1,2}(t-1) > 0 \cap n_{i,2}(t-1) < N_{i,2}\} \\ V_2 &= \{\alpha_i(t) = 1 \cap (n_{i-1,1}(t-1) = 1 \cup n_{i,1}(t-1) = N_{i,1} - 1) \\ &\qquad \cap\, n_{i-1,2}(t-1) = 0 \cap n_{i,2}(t-1) < N_{i,2}\} \end{aligned}$$

Last, we consider the two sets of states that represent M_i being down in step t, and making a type two in step $t+1$, as described in category (c). This happens if the machine is either blocked or starved for type one, and not blocked or starved for type two, and down. Since we assume that the machine cannot be simultaneously blocked and starved for either part type, these events are represented as follows.

$$\begin{aligned} D &= \{\alpha_i(t) = 0 \cap (n_{i-1,1}(t-1) = 0 \cup n_{i,1}(t-1) = N_{i,1}) \\ &\qquad \cap\, n_{i-1,2}(t-1) > 0 \cap n_{i,2}(t-1) < N_{i,2}\} \end{aligned}$$

With these events defined, we are now ready to begin our analysis of the probability $r_u(i,2)$. We start by observing that the six events defined above are disjoint. Define the event, called U, that M_i makes a type two in time step $t+1$. Then

$$U \;=\; \{\alpha_i(t+1) = 1 \cap (n_{i-1,1}(t) = 0 \cup n_{i,1}(t) = N_{i,1})$$

$$\cap \; n_{i-1,2}(t) > 0 \cap n_{i,2}(t) < N_{i,2}\}$$

That is, U is the event that the machine is up at the end of time step $t+1$, was blocked or starved for type one in step t, and was not blocked or starved for type two in step t. Under these conditions, M_i makes a part type two. Then, since the six conditioning events are disjoint, we can proceed in exactly the same manner as we did for the resumption of flow equations for part type one. That is, we can write the following:

$$\begin{aligned} r_u(i,2) &= Pr\,[U|I \cup V_1 \cup V_2 \cup D] \\ &= \frac{Pr\,[U|I]Pr\,[I] + Pr\,[U|V_1]Pr\,[V_1] + Pr\,[U|V_2]Pr\,[V_2] + Pr\,[U|D]Pr\,[D].}{Pr\,[I \cup V_1 \cup V_2 \cup D]} \end{aligned}$$

It remains for us to calculate the individual probabilities. We begin with the conditional probabilities.

We begin with $Pr\,[U|I]$. This is the conditional probability that M_i makes a type two in time step $t+1$, given it did not make either a type one or a type two in time step t because it was either blocked or starved for type one, and starved for type two. In order for this transition to occur, two events must happen simultaneously. First, the Type 2 upstream buffer for M_i has a part placed in it. That is, M_{i-1} makes a part at the end of time step t. Since we do not know the reason that M_{i-1} did not make a part at the end of time step $t-1$ — it could have been because M_{i-1} was down, or starved for Type 2 parts — we set this transition probability to be $r_u(i-1,2)$. That is, it is the probability that an observer in the buffer of line $L(i-1,2)$ sees flow resume. The second event that must occur is that M_i, which was either blocked or starved at the end of time t, is still either blocked or starved at the end of time $t+1$. That is, if M_i was starved for Type 1 parts at the end of time step t, it is still starved for them at the end of time step $t+1$m, and if it was blocked for Type 1 parts at the end of time t, it is still blocked at the end of time $t+1$. We deal with the first transition. We express the probability that M_i is still starved for Type 1 parts at the end of time $t+1$, given it was either blocked or starved for Type 1 parts at the end of time t as the probability that the observer in buffer $B_{i-1,1}$ did not see flow resume, given the buffer was empty. We express this as

$$(1 - r_u(i-1,1))\frac{p_s(i-1,1)}{p_s(i-1,1) + p_b(i,1)}.$$

Similarly, we can express the probability that an observer in $B_{i,1}$ did not see outflow resume at time $t+1$ as

$$(1 - r_d(i+1,1))\frac{p_b(i,1)}{p_s(i-1,1) + p_b(i,1)}.$$

We can then express the probability that M_i was still blocked or starved at the end of time $t+1$, given that it was blocked or starved at the end of time t as the probability that inflow did not resume into $B_{i-1,1}$ times the probability that M_i was starved over the probability that it was blocked or starved, plus the probability that outflow did not resume from $B_{i,1}$ times the probability that M_i was blocked over the probability that M_i was blocked or starved. Since the probability that M_i is simultaneously blocked and starved is assumed to be zero, this transition probability is approximated as

$$\frac{(1 - r_u(i-1,1))p_s(i-1,1) + (1 - r_d(i,1))p_b(i,1)}{p_s(i-1,1) + p_b(i,1)}.$$

We now express the probability that both events — that inflow did resume into $B_{i-1,2}$, and M_i remained blocked or starved for Type 1 parts — as the product of the two transition probabilities. This is

$$Pr\left[U|I\right] \approx r_u(i-1,2)\left(\frac{(1 - r_u(i-1,1))p_s(i-1,1) + (1 - r_d(i,1))p_b(i,1)}{p_s(i-1,1) + p_b(i,1)}\right). \quad (7.28)$$

Next, we deal with $Pr\left[U|D\right]$. The derivation is done in a manner similar to that done for $Pr\left[U|I\right]$. We note that for M_i to make a Type 2 part in step $t{+}1$, given that it did not make one in step t because it was down, several things must be true. First, M_i must undergo a repair. Second, M_i must be either blocked or starved for Type 1 parts before the repair, or it will come up making a Type 1 part. The probability of the first transition is given as r_i. The probability of the second transition is calculated by breaking down the event that M_i was blocked or starved into the two separate cases, as was done when calculating the probability $Pr\left[U|D\right]$. The transition probability is given as

$$\frac{(1 - r_u(i-1,1))\mathbf{p}(i-1,1;000) + (1 - r_d(i,1))\mathbf{p}(i,1;N00)}{\mathbf{p}(i-1,1;000) + \mathbf{p}(i,1;N00)}.$$

That is, the probability that M_i is still blocked or starved for Type 1 parts at the end of time step $t+1$, given that M_i was down, and blocked or starved at the end of time step t is the probability that inflow did not resume in buffer $B_{i-1,1}$, times the probability that M_i was down and starved, plus the probability that outflow did not resume in buffer $B_{i,1}$, times the probability that M_i was down and blocked. We then take the transition probability for the two events together to be

$$Pr\left[U|D\right] \approx r_i \left(\frac{(1 - r_u(i-1,1))\mathbf{p}(i-1,1;000) + (1 - r_d(i,1))\mathbf{p}(i,1;N00)}{\mathbf{p}(i-1,1;000) + \mathbf{p}(i,1;N00)} \right).$$

Next, we need to calculate the transition probabilities that relate to the case where M_i makes a Type 2 part because it has just become either blocked or starved for a Type 1 part. This can be further broken down into two cases: the probability that M_i makes a Type 2 part in time step $t+1$, given it was about to become blocked or starved for Type 1, and was not blocked or starved for Type 2 in time step t, and the probability that M_i makes a Type 2 part in time step $t+1$, given it was about to become blocked or starved for Type 1, and was starved for Type 2 in time step t. We deal with the former first.

We first deal with the state transition "M_i is about to become blocked or starved for Type 1". This is the event that either $B_{i-1,1}$ had one part in it, or $B_{i,1}$ had only one space for a part, and if M_i is about to become starved, M_{i-1} does not produce a Type 1 part, but if M_i is about to become blocked, M_{i+1} does not produce a Type 1 part to free up space. We begin with the event that M_i is about to become starved for a Type 1 part. We can represent the probability that M_i is about to become starved in terms transitions from the two two-machine states $\mathbf{p}(i-1,1;101)$ and $\mathbf{p}(i-1,1;111)$. If M_{i-1} is down, or starved for Type 1 parts, then we consider a transition from state $\mathbf{p}(i-1,1;101)$ in which $M_u(i-1,1)$ is not repaired; this occurs with probability $1 - r_u(i-1,1)$. If, on the other hand, M_{i-1} is up and not starved, then we consider a transition from the state $\mathbf{p}(i-1,1;111)$ in which $M_u(i-1,1)$ fails; this occurs with probability $p_u(i-1,1)$. We take

$$\frac{(1 - r_u(i-1,1))\mathbf{p}(i-1,1;101) + p_u(i-1,1)\mathbf{p}(i-1,1;111)}{\mathbf{p}(i-1,1;101) + \mathbf{p}(i-1,1;111)}$$

to be the probability of the transition from the state where M_i is about to become starved for a Type 1 part to the state where M_i is starved for a Type 1 part. In an similar manner, we can derive

$$\frac{(1 - r_d(i,1))\mathbf{p}(i,1;N-1,10) + p_d(i,1))\mathbf{p}(i,1;N-1,11)}{\mathbf{p}(i,1;N-1,10) + \mathbf{p}(i,1;N-1,11)}$$

to be the probability of transition from the state where M_i is about to become blocked for a Type 1 part to the state where M_i is blocked for a Type 1 part. From this, we can express $Pr\left[U|V_1\right]$ as

$$Pr\left[U|V_1\right] \approx \frac{(1 - r_u(i-1,1))\mathbf{p}(i-1,1;101) + p_u(i-1,1)\mathbf{p}(i-1,1;111)}{\mathbf{p}(i-1,1;101) + \mathbf{p}(i-1,1;111)}$$

$$+ \quad \frac{(1-r_d(i,1))\mathbf{p}(i,1;N-1,10)+p_d(i,1))\mathbf{p}(i,1;N-1,11)}{\mathbf{p}(i,1;N-1,10)+\mathbf{p}(i,1;N-1,11)}$$

Next, we wish to derive an expression for $Pr\,[U|V_2]$. This is done in exactly the same manner as was done for $Pr\,[U|V_1]$, with the addition that we also require the buffer $B_{i-1,2}$ to see inflow resume, which happens with probability $r_u(i-1,2)$. Thus, we set

$$\begin{aligned} Pr\,[U|V_2] \;\approx\; & r_u(i-1,2) \\ & \times \left[\frac{(1-r_u(i-1,1))\mathbf{p}(i-1,1;101)+p_u(i-1,1)\mathbf{p}(i-1,1;111)}{\mathbf{p}(i-1,1;101)+\mathbf{p}(i-1,1;111)}\right. \\ & \quad \left. +\frac{(1-r_d(i,1))\mathbf{p}(i,1;N-1,10)+p_d(i,1))\mathbf{p}(i,1;N-1,11)}{\mathbf{p}(i,1;N-1,10)+\mathbf{p}(i,1;N-1,11)}\right] \end{aligned}$$

Now we wish to calculate the probabilities of the unconditioned events, these being D, I, V_1, and V_2. We begin with event D, as it is the most complicated. In order to calculate this, it will be useful to make the following observation.

$$\begin{aligned} & \{\alpha_u(i,2)=0 \cap n_{i,2} < N_{i,2}\} \iff \\ & \{\alpha_i(t)=1 \cap n_{i-1,1}(t-1)>0 \cap n_{i,1}(t-1)<N_{i,1} \\ & \qquad \cap\, n_{i-1,2}(t-1)>0 \cap n_{i,2}(t-1)<N_{i,2}\} \,\cup \\ & \{\alpha_i(t)=1 \cap n_{i-1,1}(t-1)>0 \cap n_{i,1}(t-1)<N_{i,1} \\ & \qquad \cap\, n_{i-1,2}(t-1)=0 \cap n_{i,2}(t-1)<N_{i,2}\} \,\cup \\ & \{\alpha_i(t)=1 \cap (n_{i-1,1}(t-1)=0 \cup n_{i,1}(t-1)=N_{i,1}) \\ & \qquad \cap\, n_{i-1,2}(t-1)=0 \cap n_{i,2}(t-1)<N_{i,2}\} \,\cup \\ & \{\alpha_i(t)=0 \cap n_{i-1,1}(t-1)>0 \cap n_{i,1}(t-1)<N_{i,1} \\ & \qquad \cap\, n_{i-1,2}(t-1)>0 \cap n_{i,2}(t-1)<N_{i,2}\} \,\cup \\ & \{\alpha_i(t)=0 \cap (n_{i-1,1}(t-1)=0 \cup n_{i,1}(t-1)=N_{i,1}) \\ & \qquad \cap\, n_{i-1,2}(t-1)>0 \cap n_{i,2}(t-1)<N_{i,2}\} \,\cup \\ & \{\alpha_i(t)=0 \cap n_{i-1,1}(t-1)>0 \cap n_{i,1}(t-1)<N_{i,1} \\ & \qquad \cap\, n_{i-1,2}(t-1)=0 \cap n_{i,2}(t-1)<N_{i,2}\} \end{aligned}$$

We can simplify this by combining the first and fourth events and the second and last events. Using the notation defined in the previous section, this becomes.

$$\begin{aligned} & \{\alpha_u(i,2)=0 \cap n_{i,2} < N_{i,2}\} \iff \{A \cap B\} \\ & \cup\, \{A^c \cap n_{i-1,2}(t-1)=0 \cap n_{i,2}(t-1)<N_{i,2}\} \end{aligned}$$

$$\cup \{\alpha_i(t) = 1 \cap A^c \cap n_{i-1,2}(t-1) = 0 \cap n_{i,2}(t-1) < N_{i,2}\}$$
$$\cup \{\alpha_i(t) = 0 \cap A^c \cap B\}$$

Since these events are disjoint, the probability of the unions is the sum of the probabilities. It follows from the previous section that the probability of the second event is given as $\mathbf{p}(i-1,2;000)$, and the probability of the third event is just $p_s(i-1,2)$. Furthermore, observe that the last event corresponds precisely to the event D, whose probability we are trying to determine. Therefore, the appropriate substitutions and manipulations give

$$\begin{aligned} Pr\,[D] &= Pr\,[\alpha_u(i,2) = 0 \cap n_{i,2} < N_{i,2}] - p_s(i-1,2) - \mathbf{p}(i-1,2;000) \\ &\quad -1 + p_s(i-1,1) + \mathbf{p}(i-1,1;000) + p_b(i,1) + \mathbf{p}(i,1;N00) \end{aligned}$$

Now, as was done for the part type one upstream resumption of flow equation, we can use Equation (6.25) to rewrite the first term of the above equation as

$$I_u(i,2)\left(E(i,2) - q_u(i,2)p_b(i,2) + r_u(i,2)\mathbf{p}(i,2;N00)\right) + \frac{q_u(i,2)}{r_u(i,2)}p_b(i,2) - \mathbf{p}(i,2;N00),$$

where

$$I_u(i,2) = \frac{p_u(i,2)}{r_u(i,2)} \tag{7.29}$$

Putting everything together, we finally arrive at

$$\begin{aligned} Pr\,[D] &= I_u(i,2)\left(E(i,2) - q_u(i,2)p_b(i,2) + r_u(i,2)\mathbf{p}(i,2;N00)\right) \\ &\quad + \frac{q_u(i,2)}{r_u(i,2)}p_b(i,2) - \mathbf{p}(i,2;N00) - p_s(i-1,2) - \mathbf{p}(i-1,2;000) \\ &\quad -1 + p_s(i-1,1) + \mathbf{p}(i-1,1;000) + p_b(i,1) + \mathbf{p}(i,1;N00) \end{aligned} \tag{7.30}$$

Next, we wish to calculate the probability of the event I. It follows immediately from the previous section that

$$Pr\,[I] = p_s(i-1,2) \tag{7.31}$$

It remains to calculate the probabilities of the states V_1 and V_2. Unfortunately, exact analysis of these is not possible, so we have to make some approximations. We begin with $Pr\,[V_1]$.

We will express the probability of the event V_1 in terms of two-machine line $L(i-1,2)$ states. Since M_i is up, and not blocked or starved for Type 1 parts, we must have that $M_d(i-1,2)$ is down. We cannot, however, say anything about $M_u(i-1,2)$. Therefore, we must consider both the case where $M_u(i-1,2)$ is up, and the case when it is down. We start with the case when it is up. We propose to approximate this by

$$\sum_{n=1}^{N} \mathbf{p}(i-1,2;n10).$$

We know that this is not exact for a number of reasons. First, the states of the Type 1 parts are absorbed into the event that $M_d(i-1,2)$ is down. That is, we have absorbed the fact that M_i is up, and $n_{i-1,1}=1$ or $n_{i,1}=N_{i,1}-1$ into the simple statement that $M_d(i-1,2)$ is down. We know that, in fact, $M_d(i-1,2)$ will be down more often than in the case we have addressed. For example, it will be down if M_i is up, and $B_{i,2}$ is full, or in the case where M_i is down. However, expressing it as the sum given above makes the analysis much easier, and the resulting algorithm more stable. In a similar manner, we can approximate the case where $M_u(i-1,2)$ is down by the sum

$$\sum_{n=1}^{N} \mathbf{p}(i-1,2;n00).$$

Together, we have

$$Pr\left[V_1\right] \approx \sum_{n=1}^{N} \left(\mathbf{p}(i-1,2;n10) + \mathbf{p}(i-1,2;n00)\right). \tag{7.32}$$

By following a similar line of reasoning, we can write the probability of the event V_2 as

$$\mathbf{p}(i-1,2;000) + \mathbf{p}(i-1,2;010).$$

But since $\mathbf{p}(i-1,2;010)$, we write this as

$$Pr\left[V_2\right] = \mathbf{p}(i-1,2;000). \tag{7.33}$$

We can make no claims to the optimality of our approximation, or even that it is a good one. However, experimentation with several approximate expressions for the probabilities of events V_1 and V_2 have led us to conclude that the algorithm converges most often, and is most accurate with the above approximations. No doubt the algorithm could be improved if better approximations were found. We leave this as an open problem that merits further investigation.

Finally, we have the expression for the Type 2 resumption of flow probability for the upstream machine. It is given as:

$$
\begin{aligned}
r_u(i,2) \quad = \quad & \Bigg(r_i \left[\frac{(1 - r_u(i-1,1))\mathbf{p}(i-1,1;000) + (1 - r_d(i,1))\mathbf{p}(i,1;N00)}{\mathbf{p}(i-1,1;000) + \mathbf{p}(i,1;N00)} \right] \\
& \quad \times \left[\frac{p_u(i,2)}{r_u(i,2)} \left\{ E(i-1,2) - q_u(i,2)p_b(i,2) + r_u(i,2)\mathbf{p}(i,2;N00) \right\} \right. \\
& \quad + \frac{q_u(i,2)}{r_u(i,2)} p_b(i,2) - \mathbf{p}(i,2;N00) - p_s(i-1,2) - \mathbf{p}(i-1,2;000) - 1 \\
& \quad \left. + p_s(i-1,1) + \mathbf{p}(i-1,1;000) + p_b(i,1) + \mathbf{p}(i,1;N00) \right] \\
& + r_u(i-1,2)(1 - q_d(i-1,2))p_s(i-1,2) \\
& + \left[\frac{(1 - r_u(i-1,1))\mathbf{p}(i-1,1;101) + p_u(i-1,1)\mathbf{p}(i-1,1;111)}{\mathbf{p}(i-1,1;101) + \mathbf{p}(i-1,1;111)} \right] \\
& \quad \times \left[\sum_{n=1}^{N} \left\{ \mathbf{p}(i-1,2;n10) + \mathbf{p}(i-1,2;n00) \right\} \right. \\
& \quad \left. + r_u(i-1,2)\mathbf{p}(i-1,2;000) \right] (1 - p_u(i-1,1)) \\
& + \left[\frac{(1 - r_d(i,1))\mathbf{p}(i,1;N-1,10) + p_d(i,1))\mathbf{p}(i,1;N-1,11)}{\mathbf{p}(i,1;N-1,10) + \mathbf{p}(i,1;N-1,11)} \right] \\
& \quad \times \left[\sum_{n=1}^{N} \left\{ \mathbf{p}(i-1,2;n10) + \mathbf{p}(i-1,2;n00) \right\} \right. \\
& \quad \left. + r_u(i-1,2)\mathbf{p}(i-1,2;000) \right] (1 - p_u(i-1,1)) \Bigg) \\
\div \quad & \Bigg(\frac{p_u(i,2)}{r_u(i,2)} \left\{ E(i-1,2) - q_u(i,2)p_b(i,2) + r_u(i,2)\mathbf{p}(i,2;N00) \right\} \\
& \quad + \frac{q_u(i,2)}{r_u(i,2)} p_b(i,2) - \mathbf{p}(i,2;N00) - 1 \\
& \quad + p_s(i-1,1) + \mathbf{p}(i-1,1;000) + p_b(i,1) + \mathbf{p}(i,1;N00) \\
& \quad + \sum_{n=1}^{N} \left\{ \mathbf{p}(i-1,2;n10) + \mathbf{p}(i-1,2;n00) \right\} \Bigg)
\end{aligned}
$$

Following an analogous procedure, we can derive the expression for the Type 2 resumption of flow probability for the downstream machine. This is done in Appendix B, and we refer to reader there for details. The expression is

$$
\begin{aligned}
r_d(i,2) \quad = \quad & \Bigg(r_{i+1} \left[\frac{(1-r_u(i,1))\mathbf{p}(i,1;000) + (1-r_d(i+1,1))\mathbf{p}(i+1,1;N00)}{\mathbf{p}(i,1;000)+\mathbf{p}(i+1,1;N00)} \right] \\
& \quad \times \Bigg[\frac{p_d(i,2)}{r_d(i,2)} \left\{ E(i+1,2) - q_d(i,2)p_s(i,2) + r_d(i,2)\mathbf{p}(i,2;000) \right\} \\
& \quad + \frac{q_d(i,2)}{r_d(i,2)} p_s(i,2) - \mathbf{p}(i,2;000) - p_b(i+1,2) - \mathbf{p}(i+1,2;N00) - 1 \\
& \quad + p_s(i,1) + \mathbf{p}(i,1;000) + p_b(i+1,1) + \mathbf{p}(i+1,1;N00) \Bigg] \\
& + r_d(i+1,2)(1-q_u(i,2))p_b(i+1,2) \\
& + \left[\frac{(1-r_u(i,1))\mathbf{p}(i,1;101) + p_u(i,1)\mathbf{p}(i,1;111)}{\mathbf{p}(i,1;101)+\mathbf{p}(i,1;111)} \right] \\
& \quad \times \Bigg[\sum_{n=1}^{N} \Big\{ \mathbf{p}(i-1,2;n10) + \mathbf{p}(i-1,2;n00) \Big\} \\
& \quad + r_u(i-1,2)\mathbf{p}(i-1,2;000) \Bigg] (1-p_d(i+1,1)) \\
& + \Bigg[\frac{(1-r_d(i+1,1))\mathbf{p}(i+1,1;N-1,10)}{\mathbf{p}(i+1,1;N-1,10)+\mathbf{p}(i+1,1;N-1,11)} \\
& \quad + \frac{p_d(i+1,1))\mathbf{p}(i+1,1;N-1,11)}{\mathbf{p}(i+1,1;N-1,10)+\mathbf{p}(i+1,1;N-1,11)} \Bigg] \\
& \quad \times \Bigg[\sum_{n=0}^{N-1} \Big\{ \mathbf{p}(i+1,2;n01) + \mathbf{p}(i+1,2;n00) \Big\} \\
& \quad + r_d(i+1,2)\mathbf{p}(i+1,2;N00) \Bigg] (1-p_d(i+1,1)) \Bigg) \\
\div \quad & \Bigg(\frac{p_d(i,2)}{r_d(i,2)} \left\{ E(i+1,2) - q_d(i,2)p_s(i,2) + r_d(i,2)\mathbf{p}(i,2;000) \right\} \\
& \quad + \frac{q_d(i,2)}{r_d(i,2)} p_s(i,2) - \mathbf{p}(i,2;000) - 1 \\
& \quad + p_s(i,1) + \mathbf{p}(i,1;000) + p_b(i+1,1) + \mathbf{p}(i+1,1;N00) \\
& \quad + \sum_{n=0}^{N-1} \Big\{ \mathbf{p}(i+1,2;n01) + \mathbf{p}(i+1,2;n00) \Big\} \Bigg)
\end{aligned}
$$

7.7 Interruption of Flow Equations

The interruption of flow equations are used to calculate the values of $p_u(i,j)$ for $j = 1,2$ and $i = 1,\ldots,K$, giving $2K$ equations, and $p_d(i,j)$ for $j = 1,2$ and $i =$

$0, \ldots, K-1$, giving another $2K$ equations, for a total of $4K$ equations.

The role played by the interruption of flow equations is complementary to those played by the resumption of flow equations. While not used explicitly in the decomposition for the single part transfer line, it is possible to show, using the flow rate-idle time relationship, that when there is only one part type,

$$p_u(i) = p_i + \frac{r_u(i-1)p_s(i-1)}{E(i)},$$

and

$$p_d(i-1) = p_{i+1} + \frac{r_d(i)p_b(i)}{E(i+1)}.$$

Notice that while the resumption of flow equations were a weighted average, that approximated up- and downstream behavior, the interruption of flow parameters will always be greater than the corresponding real machine failure parameter. This is because the failure parameter of the machine in the two-machine line will not only have to take into account the failure of the corresponding processing line machine, but also the failure of flow due to blockage or starvation.

7.7.1 Type One

The parameter $p_u(i,1)$ is the failure probability of the upstream machine of the two-machine sub-line $L(i,1)$. That is, it is the probability that $M_u(i,1)$ does not make a part in time step $t+1$, given it did make one, and was not blocked, in time step t. This is expressed as

$$p_u(i,1) \quad = \quad Pr\left[M_u(i,1) \text{ down at time} t+1 | M_u(i,1) \text{ up, } n_{i,1} < N_{i,1} \text{ at time } t\right]$$

There are two ways that M_i could have not produced a type one part in time step $t+1$, given that it did produce one, and was not blocked for type one at time t. They are

- M_i was down at the end to time step $t+1$,
- or $n_{i-1,1} = 0$ and $M_u(i-1,1)$ is down at the end of time step $t+1$.

We can then write

$$\begin{aligned} p_u(i,1) \quad = \quad & Pr\left[\alpha_i(t+1) = 0 | M_u(i,1) \text{ up, } n_{i,1} < N_{i,1} \text{ at time } t\right] \\ + \quad & Pr\left[n_{i-1,1} = 0 \text{ and } M_u(i-1,1) \text{ down at time step } t+1 | \right. \\ & \qquad \left. M_u(i,1) \text{ up, } n_{i,1} < N_{i,1} \text{ at time } t\right] \end{aligned}$$

The first term evaluates to p_i. The second can be further split into

$$\begin{aligned}
&Pr\left[n_{i-1,1}=0, M_u(i-1,1) \text{ down }, M_d(i-1,1) \text{ up at time step } t+1\right| \\
&\qquad\qquad\qquad M_u(i,1) \text{ up}, n_{i,1}<N_{i,1} \text{ at time } t] \\
&+Pr\left[n_{i-1,1}=0, M_u(i-1,1) \text{ down }, M_d(i-1,1) \text{ down at time step } t+1\right| \\
&\qquad\qquad\qquad M_u(i,1) \text{ up}, n_{i,1}<N_{i,1} \text{ at time } t]
\end{aligned} \tag{7.34}$$

We deal first with the first term in (7.34). If $n_{i-1,1}=0$, $M_u(i-1,1)$ down, and $M_d(i-1,1)$ up at the end of time step $t+1$, then this can happen only if

- $n_{i-1,1}=0$,$M_u(i-1,1)$ down, and $M_d(i-1,1)$ down at the end of time step t, $M_u(i-1,1)$ is not repaired with probability $1-r_u(i-1,1)$, and $M_d(i-1,1)$ is repaired with probability $r_d(i-1,1)$.
- $n_{i-1,1}=1$,$M_u(i-1,1)$ up, and $M_d(i-1,1)$ down at the end of time step t, $M_u(i-1,1)$ fails with probability $p_u(i-1,1)$, and $M_d(i-1,1)$ is repaired with probability $r_d(i-1,1)$.
- $n_{i-1,1}=0$,$M_u(i-1,1)$ down, and $M_d(i-1,1)$ up at the end of time step t, $M_u(i-1,1)$ is not repaired with probability $1-r_u(i-1,1)$, and $M_d(i-1,1)$ does not fail with probability $1-p_d(i-1,1)$.
- $n_{i-1,1}=1$,$M_u(i-1,1)$ down, and $M_d(i-1,1)$ down at the end of time step t, $M_u(i-1,1)$ is not repaired, with probability $1-r_u(i-1,1)$, and $M_d(i-1,1)$ is not repaired, with probability $1-r_d(i-1,1)$.
- $n_{i-1,1}=1$,$M_u(i-1,1)$ down, and $M_d(i-1,1)$ up at the end of time step t, $M_u(i-1,1)$ is not repaired, with probability $1-r_u(i-1,1)$, and $M_d(i-1,1)$ does not fail, with probability $1-p_d(i-1,1)$.
- $n_{i-1,1}=1$,$M_u(i-1,1)$ up, and $M_d(i-1,1)$ up at the end of time step t, $M_u(i-1,1)$ fails, with probability $p_u(i-1,1)$, and $M_d(i-1,1)$ does not fail, with probability $1-p_d(i-1,1)$.

Thus, the first term of (7.34) is equivalent to

$$\begin{aligned}
&Pr\left[n_{i-1,1}=0, M_u(i-1,1) \text{ down }, M_d(i-1,1) \text{ down at time step } t\right| \\
&\qquad\qquad M_u(i,1) \text{ up}, n_{i,1}<N_{i,1} \text{ at time } t]\,(1-r_u(i-1,1))r_d(i-1,1) \\
&+Pr\left[n_{i-1,1}=1, M_u(i-1,1) \text{ up }, M_d(i-1,1) \text{ down at time step } t\right| \\
&\qquad\qquad M_u(i,1) \text{ up}, n_{i,1}<N_{i,1} \text{ at time } t]\,r_u(i-1,1)r_d(i-1,1) \\
&+Pr\left[n_{i-1,1}=0, M_u(i-1,1) \text{ down }, M_d(i-1,1) \text{ up at time step } t\right| \\
&\qquad\qquad M_u(i,1) \text{ up}, n_{i,1}<N_{i,1} \text{ at time } t]\,(1-r_u(i-1,1))(1-q_d(i-1,1))
\end{aligned}$$

$$
\begin{aligned}
&+Pr\,[n_{i-1,1} = 1, M_u(i-1,1) \text{ down }, M_d(i-1,1) \text{ down at time step } t| \\
&\qquad M_u(i,1) \text{ up, } n_{i,1} < N_{i,1} \text{ at time } t]\,(1-r_u(i-1,1))r_d(i-1,1) \\
&+Pr\,[n_{i-1,1} = 1, M_u(i-1,1) \text{ down }, M_d(i-1,1) \text{ up at time step } t| \\
&\qquad M_u(i,1) \text{ up, } n_{i,1} < N_{i,1} \text{ at time } t]\,(1-r_u(i-1,1))(1-p_d(i-1,1)) \\
&+Pr\,[n_{i-1,1} = 1, M_u(i-1,1) \text{ up }, M_d(i-1,1) \text{ up at time step } t| \\
&\qquad M_u(i,1) \text{ up, } n_{i,1} < N_{i,1} \text{ at time } t]\,p_u(i-1,1)(1-p_d(i-1,1))
\end{aligned} \tag{7.35}
$$

By using Bayes' formula, we can rewrite the first term of (7.35) as

$$
\frac{Pr\left[\begin{array}{c} n_{i-1,1} = 0, M_u(i-1,1) \text{ down }, M_d(i-1,1) \text{ down }, \\ M_u(i,1) \text{ up, } n_{i,1} < N_{i,1} \text{ at time } t \end{array}\right]}{Pr\,[M_u(i,1) \text{ up, } n_{i,1} < N_{i,1} \text{ at time } t]}
$$

We now argue that this term is zero. Recall that $M_u(i,1)$ is up at the end of time step t if, and only if, M_i is up at the end of time step t, and $n_{i-1,1}(t-1) > 0$. Then the event $n_{i-1,1} = 0$ and $M_u(i,1)$ up are mutually exclusive. From a similar argument, it follows that the third term of (7.35) is also zero.

The main part of the second term of (7.35) may be written as

$$
\frac{Pr\left[\begin{array}{c} n_{i-1,1} = 1, M_u(i-1,1) \text{ up }, M_d(i-1,1) \text{ down }, \\ M_u(i,1) \text{ up, } n_{i,1} < N_{i,1} \text{ at time } t \end{array}\right]}{Pr\,[M_u(i,1) \text{ up, } n_{i,1} < N_{i,1} \text{ at time } t]}
$$

This is zero. We argue as follows. In order for $M_u(i,1)$ to be up, we must have that M_i is up, and $n_{i-1,1} > 0$. Likewise, $M_d(i-1,1)$ is up at the end of time step t if, and only if, M_i is up at the end of time step t, and $n_{i,1}(t-1) < N_{i,1}$. Therefore, we have that

$$
\{M_u(i,1) \text{ up, and } n_{i,1} < N_{i,1}\} \iff \{M_d(i-1,1) \text{ up, and } n_{i-1,1} > 0\} \tag{7.36}
$$

It follows that it is impossible for $M_u(i,1)$ to be up, $n_{i,1} < N_{i,1}$, $n_{i-1,1} > 0$, and $M_d(i-1,1)$ to be down, simultaneously. A similar argument shows that the fourth term of (7.35) must also be zero. This leaves only the fifth and sixth terms.

The main part of the fifth term of (7.35) can be written

$$
\frac{Pr\left[\begin{array}{c} n_{i-1,1} = 1, M_u(i-1,1) \text{ down }, M_d(i-1,1) \text{ up }, \\ M_u(i,1) \text{ up, } n_{i,1} < N_{i,1} \text{ at time } t \end{array}\right]}{Pr\,[M_u(i,1) \text{ up, } n_{i,1} < N_{i,1} \text{ at time } t]}
$$

From (7.36), this is equivalent to

$$\frac{Pr\left[n_{i-1,1}=1, M_u(i-1,1) \text{ down }, M_d(i-1,1) \text{ up at time } t\right]}{Pr\left[M_u(i,1) \text{ up}, n_{i,1} < N_{i,1} \text{ at time } t\right]}$$

Similarly, the sixth term is equivalent to

$$\frac{Pr\left[n_{i-1,1}=1, M_u(i-1,1) \text{ up }, M_d(i-1,1) \text{ up at time } t\right]}{Pr\left[M_u(i,1) \text{ up}, n_{i,1} < N_{i,1} \text{ at time } t\right]}$$

Together, we have finally that the second term in (7.34) is equal to

$$\begin{aligned} & \frac{\mathbf{p}(i-1,1;101)(1-r_u(i-1,1))(1-p_d(i-1,1))}{Pr\left[M_u(i,1) \text{ up}, n_{i,1} < N_{i,1} \text{ at time } t\right]} \\ + & \frac{\mathbf{p}(i-1,1;111)p_u(i-1,1)(1-p_d(i-1,1))}{Pr\left[M_u(i,1) \text{ up}, n_{i,1} < N_{i,1} \text{ at time } t\right]} \end{aligned}$$

Next, we address the second term of (7.34). If $n_{i-1,1}=0$, $M_u(i-1,1)$ down, and $M_d(i-1,1)$ down at the end of time step $t+1$, then this can happen only if

- $n_{i-1,1}=0$,$M_u(i-1,1)$ down, and $M_d(i-1,1)$ down at the end of time step t, $M_u(i-1,1)$ is not repaired with probability $1-r_u(i-1,1)$, and $M_d(i-1,1)$ is not repaired with probability $1-r_d(i-1,1)$.
- $n_{i-1,1}=0$,$M_u(i-1,1)$ down, and $M_d(i-1,1)$ up at the end of time step t, $M_u(i-1,1)$ is not repaired with probability $1-r_u(i-1,1)$, and $M_d(i-1,1)$ is repaired with probability $r_d(i-1,1)$.

Thus, the first term of (7.34) is equivalent to

$$\begin{aligned} & Pr\left[n_{i-1,1}=0, M_u(i-1,1) \text{ down }, M_d(i-1,1) \text{ down at time step } t \right| \\ & \qquad M_u(i,1) \text{ up}, n_{i,1} < N_{i,1} \text{ at time } t\left.\right] (1-r_u(i-1,1))(1-r_d(i-1,1)) \\ & + Pr\left[n_{i-1,1}=0, M_u(i-1,1) \text{ down }, M_d(i-1,1) \text{ up at time step } t \right| \\ & \qquad M_u(i,1) \text{ up}, n_{i,1} < N_{i,1} \text{ at time } t\left.\right] (1-r_u(i-1,1))r_d(i-1,1) \end{aligned}$$

The first term of the expression above is zero according to (7.36). The main part of the second term is easily seen to be

$$\frac{Pr\left[n_{i-1,1}=0, M_u(i-1,1) \text{ down }, M_d(i-1,1) \text{ up at time } t\right]}{Pr\left[M_u(i,1) \text{ up}, n_{i,1} < N_{i,1} \text{ at time } t\right]}$$

Putting the derived quantities from the first and second terms of (7.34) together then gives

$$\frac{\begin{array}{c} p_s(i-1,1)(1-r_u(i-1,1))q_d(i-1,1) \\ +\mathbf{p}(i-1,1;101)(1-r_u(i-1,1))(1-p_d(i-1,1)) \\ +\mathbf{p}(i-1,1;111)p_u(i-1,1)(1-p_d(i-1,1)) \end{array}}{Pr\left[M_u(i,1) \text{ up, } n_{i,1} < N_{i,1} \text{ at time } t\right]} \tag{7.37}$$

It follows from the flow balance equations derived in Chapter 6 that the numerator is equal to

$$\begin{array}{ll} & (r_u(i-1,1)+(1-r_u(i-1,1))q_d(i-1,1))p_s(i-1,1) \\ + & (1-r_u(i-1,1))r_d(i-1,1)\mathbf{p}(i-1,1;000) \\ + & p_u(i-1,1)r_d(i-1,1)\mathbf{p}(i-1,1;110). \end{array}$$

From (6.28), the denominator is equal to

$$E(i-1,1) - q_u(i-1,1)\mathbf{p}(i-1,1;N10) + r_u(i-1,1)\mathbf{p}(i-1,1;N00).$$

Putting everything together, we have derived the interruption of flow equation:

$$\begin{aligned} p_u(i,1) &= p_i \\ &+ \frac{(r_u(i-1,1)+(1-r_u(i-1,1))q_d(i-1,1))p_s(i-1,1)}{E(i,1)-q_u(i-1,1)p_b(i-1,1)+r_u(i-1,1)\mathbf{p}(i-1,1;N00)} \\ &+ \frac{(1-r_u(i-1,1))r_d(i-1,1)\mathbf{p}(i-1,1;000)}{E(i,1)-q_u(i-1,1)p_b(i-1,1)+r_u(i-1,1)\mathbf{p}(i-1,1;N00)} \\ &+ \frac{p_u(i-1,1)r_d(i-1,1)\mathbf{p}(i-1,1;110)}{E(i,1)-q_u(i-1,1)p_b(i-1,1)+r_u(i-1,1)\mathbf{p}(i-1,1;N00)} \end{aligned}$$

In a similar manner, we can derive an expression for the downstream interruption of flow parameter. It is given as:

$$\begin{aligned} p_d(i-1,1) &= p_i \\ &+ \frac{(r_d(i,1)+q_u(i,1)(1-r_d(i,1))p_b(i,1)}{E(i,1)-q_d(i,1)p_s(i,1)+r_d(i,1)\mathbf{p}(i,1;000)} \\ &+ \frac{r_u(i,1)(1-r_d(i,1))\mathbf{p}(i,1;N00)}{E(i,1)-q_d(i,1)p_s(i,1)+r_d(i,1)\mathbf{p}(i,1;000)} \\ &+ \frac{r_u(i,1)p_d(i,1)\mathbf{p}(i,1;N-1,10)}{E(i,1)-q_d(i,1)p_s(i,1)+r_d(i,1)\mathbf{p}(i,1;000)} \end{aligned}$$

7.7.2 Type Two

While it is possible to derive interruption of flow equations for part type two along the lines done for type one, it is extremely messy. Therefore, we prefer to use the approximate flow rate-idle time relationship described in Chapter 5. When this work is extended to more than two part types, it will probably be necessary to abandon this tack. However, it suits our purposes now, and as the results will show in Chapter 8, the resulting decomposition is quite accurate.

From Equation (5.8) and Equation (7.21), we have the approximate relationship

$$\begin{aligned} E(i,2) \approx\ & e_i \left(p_s(i-1,1) + p_b(i,1) + \mathbf{p}(i-1,1;000) + \mathbf{p}(i,1;N00)\right. \\ & \left. - p_s(i-1,2) - p_b(i,2)\right) \end{aligned} \tag{7.38}$$

Observe that equations (6.38) and (6.39) can be rewritten to yield the following two expressions.

$$p_s(i,j) = \frac{\left(1 - \frac{E(i,j)}{e_d(i,j)}\right) r_d(i,j)R(i,j)}{r_d(i,j)R(i,j) + q_d(i,j)(R(i,j) - r_u(i,j)r_d(i,j) - r_u(i,j)p_d(i,j))} \tag{7.39}$$

$$p_b(i,j) = \frac{\left(1 - \frac{E(i,j)}{e_u(i,j)}\right) r_u(i,j)R(i,j)}{r_u(i,j)R(i,j) + q_u(i,j)(R(i,j) - r_u(i,j)r_d(i,j) - r_d(i,j)p_u(i,j))} \tag{7.40}$$

where

$$R(i,j) = r_u(i,j) + r_d(i,j) - r_u(i,j)r_d(i,j) \tag{7.41}$$

Define the constants

$$C = p_s(i-1,1) - \mathbf{p}(i-1,1;000) - p_b(i,1) - \mathbf{p}(i,1;N00) \tag{7.42}$$

and

$$\begin{aligned} P_d(i,2) &= \frac{r_d(i,2)R(i,2)}{r_d(i,2)R(i,2) + q_d(R(i,2) - r_u(i,2)r_d(i,2) - r_u(i,2)p_d(i,2))} \\ P_u(i,2) &= \frac{r_u(i,2)R(i,2)}{r_u(i,2)R(i,2) + q_u(i,2)(R(i,2) - r_u(i,2)r_d(i,2) - r_d(i,2)p_u(i,2))} \end{aligned}$$

Then we can write the approximate flow rate-idle time equation as

$$E(i,2) \approx e_i(C - p_s(i-1,2) - p_b(i,2)), \tag{7.43}$$

By dividing through by e_i, and some simply algebra, we arrive at

$$P_d(i-1,2)\frac{E(i-1,2)}{e_d(i-1,2)} + P_u(i,2)\frac{E(i,2)}{e_u(i,2)} = \frac{E(i,2)}{e_i} + P_d(i-1,2) + P_u(i,2) - 1 \tag{7.44}$$

Now, because of conservation of flow, we know that $E(i-1,2) = E(i,2)$, so we make this substitution in Equation (7.44). Furthermore, we observe that

$$\begin{aligned}
\frac{1}{e_d(i-1,2)} &= 1 + \frac{p_d(i-1,2)}{r_d(i-1,2)} \\
\frac{1}{e_u(i-1,2)} &= 1 + \frac{p_u(i,2)}{r_u(i,2)}
\end{aligned}$$

By substituting these equations into Equation (7.44), and with some simple manipulation, we arrive at the approximate flow rate idle time relationships for part type two.

$$\begin{aligned}
\frac{p_u(i,2)}{r_u(i,2)} &= \frac{1}{e_iP_u(i,2)} + \frac{P_d(i-1,2)+P_u(i,2)-1}{E(i,2)P_u(i,2)} - \frac{P_d(i-1,2)}{P_u(i,2)} \qquad (7.45) \\
&\quad -1 - \frac{p_d(i-1,2)}{r_d(i-1,2)}\frac{P_d(i-1,2)}{P_u(i,2)} \\
\frac{p_d(i-1,2)}{r_d(i-1,2)} &= \frac{1}{e_iP_d(i-1,2)} + \frac{P_d(i-1,2)+P_u(i,2)-1}{E(i,2)P_d(i-1,2)} \qquad (7.46) \\
&\quad -\frac{P_u(i,2)}{P_d(i-1,2)} - 1 - \frac{p_u(i,2)}{r_u(i,2)}\frac{P_u(i,2)}{P_d(i-1,2)}
\end{aligned}$$

Observe that if $q_u(i,2)$ and $q_d(i-1,2)$ are both zero — i.e., if there is no part type one — then both $P_d(i-1,2)$ and $P_u(i,2)$ are unity, and the above equations reduce to the approximate flow rate-idle time equations for the deterministic processing time flow line found in Gershwin [16].

The final step in the derivation is to multiply (7.45) by $r_u(i,2)$, and (7.46) by $r_d(i-1,2)$ to get

$$\begin{aligned}
p_u(i,2) &= r_u(i,2)\Bigg(\frac{1}{e_iP_u(i,2)} + \frac{P_d(i-1,2)+P_u(i,2)-1}{E(i,2)P_u(i,2)} \\
&\quad -\frac{P_d(i-1,2)}{P_u(i,2)} - 1 - \frac{p_d(i-1,2)}{r_d(i-1,2)}\frac{P_d(i-1,2)}{P_u(i,2)}\Bigg)
\end{aligned}$$

and

$$\begin{aligned} p_d(i-1,2) &= r_d(i-1,2)\left(\frac{1}{e_i P_d(i-1,2)} + \frac{P_d(i-1,2)+P_u(i,2)-1}{E(i,2)P_d(i-1,2)}\right. \\ &\left. -\frac{P_u(i,2)}{P_d(i-1,2)} - 1 - \frac{p_u(i,2)}{r_u(i,2)}\frac{P_u(i,2)}{P_d(i-1,2)}\right) \end{aligned}$$

7.8 Idleness Failure Equations

Next, we derive expressions for the idleness failure probabilities of the two-machine lines. That is, we need expressions for $q_u(i,j)$, where $j = 1,2$ and $i = 1,2,\ldots,K$ and $q_d(i,j)$, where $j = 1,2$, and $i = 0,1,\ldots,K-1$. This will give us $4K$ additional equations.

7.8.1 Type One

The probability $q_u(i,1)$ represents the probability that pseudo-machine $M_u(i,1)$ is down in step $t+1$, given that is was up and blocked in step t. Since we assume operation dependent failures, the only way that this is possible is if the processing machine M_i failed while making a part type two in step t. Therefore, this can be expressed as

$$\begin{aligned} q_u(i,1) &= Pr\left[\alpha_i(t+1)=0|\alpha_i(t)=1 \cap n_{i-1,1}(t-1)=0 \cap n_{i,1}(t-1) < N_{i,1}\right. \\ &\quad \left.\cap\, n_{i-1,2}(t-1) > 0 \cap n_{i,2}(t-1) < N_{i,1}\right] \\ &= p_i Pr\left[\alpha_i(t)=1 \cap n_{i-1,1}(t-1)=0 \cap n_{i,1}(t-1) < N_{i,1}\right. \\ &\quad \left.\cap\, n_{i-1,2}(t-1) > 0 \cap n_{i,2}(t-1) < N_{i,1}\right] \end{aligned}$$

It remains to calculate the expression in the second line above. From first principals, we have

$$Pr[A] = Pr[A \cap B] + Pr[A \cap B^c]$$

Thus,

$$\begin{aligned} & Pr\left[\alpha_i(t)=1 \cap n_{i-1,1}(t-1)=0 \cap n_{i,1}(t-1) < N_{i,1}\right. \\ &\quad \left.\cap\, n_{i-1,2}(t-1) > 0 \cap n_{i,2}(t-1) < N_{i,1}\right] \\ = \;& Pr\left[\alpha_i(t)=1 \cap n_{i-1,1}(t-1)=0 \cap n_{i,1}(t-1) < N_{i,1}\right] \\ - \;& Pr\left[\alpha_i(t)=1 \cap n_{i-1,1}(t-1)=0 \cap n_{i,1}(t-1) < N_{i,1}\right. \end{aligned}$$

$$\cap\ (n_{i-1,2}(t-1) = 0 \cup n_{i,2}(t-1) = N_{i,1})]$$

The first term is defined to be $p_s(i-1,1)$. We argue that the second term can be approximated by

$$(p_s(i-1,2) + p_b(i,2))\,\frac{p_s(i-1,1)}{p_s(i-1,1) + p_b(i,1)}.$$

This follows because, from (7.23), we know that the probability that machine M_i is either blocked or starved for type one, and either blocked or starved for type two is given by

$$p_s(i-1,2) + p_b(i,2).$$

It follows that the probability that M_i is starved for type one, and either blocked or starved for type two would be the probability that it is either blocked or starved for type one, and either blocked or starved for type two, times the weighted probability that the machine is starved for type one, given it is blocked or starved for type one. The result is our idleness failure equation for the upstream machine of line $L(i,1)$.

$$q_u(i,1) = p_i p_s(i-1,1)\left(1 - \frac{p_s(i-1,2) + p_b(i,2)}{p_s(i-1,1) + p_b(i,1)}\right).$$

Similarly, we can derive

$$q_d(i-1,1) = p_i p_b(i,1)\left(1 - \frac{p_s(i-1,2) + p_b(i,2)}{p_s(i-1,1) + p_b(i,1)}\right).$$

7.8.2 Type Two

Next, we derive the expression for $q_u(i,2)$, the probability that the upstream machine in $L(i,2)$ undergoes a failure while it is blocked. This can only occur if M_i begins to process type one parts because the only time $M_u(i,2)$ will be up when blocked is if M_i is either blocked or starved for type one parts, and blocked for type two parts. This is given by

$$\begin{aligned} q_u(i,2) \;=\;& Pr\,[\alpha_i(t+1) = 0 \cap n_{i-1,1} > 0 \cap n_{i,1} < N_{i,1} \text{ at time } t+1 | \\ & \alpha_i(t) = 1 \cap (n_{i-1,1}(t-1) = 0 \cup n_{i,1}(t-1) = N_{i,1}) \\ & \cap\ n_{i-1,2}(t-1) > 0 \cap n_{i,2}(t-1) < N_{i,1}] \end{aligned}$$

If we define the following events

$$
\begin{aligned}
U &= \{\alpha_i(t+1) = 1 \cap n_{i-1,1}(t) > 0 \cap n_{i,1}(t) < N_{i,1}\} \\
V &= \{\alpha_i(t) = 1 \cap n_{i-1,1}(t-1) = 0 \cap n_{i,1}(t-1) < N_{i,1} \\
&\qquad\qquad \cap\, n_{i-1,2}(t-1) > 0 \cap n_{i,2}(t-1) < N_{i,1}\} \\
W &= \{\alpha_i(t) = 1 \cap n_{i-1,1}(t-1) > 0 \cap n_{i,1}(t-1) = N_{i,1} \\
&\qquad\qquad \cap\, n_{i-1,2}(t-1) > 0 \cap n_{i,2}(t-1) < N_{i,1}\}
\end{aligned}
$$

then $q_u(i,2)$ can be written as

$$q_u(i,2) = \frac{Pr\,[U|V]Pr\,[V] + Pr\,[U|W]Pr\,[W]}{Pr\,[V \cup W]}$$

It is straightforward to see that

$$Pr\,[U|V] = r_u(i-1,1),$$

and

$$Pr\,[U|W] = r_d(i,1).$$

Furthermore,

$$Pr\,[V \cup W] = p_b(i,2),$$

by definition. It remains to identify the quantities $Pr\,[V]$ and $Pr\,[W]$. But these are just (7.26) and (7.27), respectively. Making these substitutions, and performing some simple algebra gives

$$q_u(i,2) = \frac{r_u(i-1,1)p_s(i-1,1) + r_d(i,1)p_b(i,1)}{p_s(i-1,1) + p_b(i,1)}$$

In a similar manner, we can derive

$$q_d(i-1,2) = \frac{r_u(i-1,1)p_s(i-1,1) + r_d(i,1)p_b(i,1)}{p_s(i-1,1) + p_b(i,1)}$$

7.9 Boundary Conditions

The boundary conditions give the remaining number of equations for the parameters of the two machine lines. This gives us the remaining 12 equations that we need for the decomposition.

They are as follows.

$$
\begin{aligned}
r_u(0,1) &= r_{1,1} \\
p_u(0,1) &= p_{1,1} \\
q_u(0,1) &= 0 \\
r_u(0,2) &= r_{1,2} \\
p_u(0,2) &= p_{1,2} \\
q_u(0,2) &= 0 \\
r_d(K,1) &= r_{K+1,1} \\
p_d(K,1) &= p_{K+1,1} \\
q_d(K,1) &= 0 \\
r_d(K,2) &= r_{K+1,2} \\
p_d(K,2) &= p_{K+1,2} \\
q_d(K,2) &= 0
\end{aligned}
\tag{7.47}
$$

7.10 Summary of the decomposition equations

$$
q_u(i,1) = p_i p_s(i-1,1)\left(1 - \frac{p_s(i-1,2)+p_b(i,2)}{p_s(i-1,1)+p_b(i,1)}\right) \tag{7.48}
$$

$$
\begin{aligned}
p_u(i,1) &= p_i + \Big((r_u(i-1,1) + (1-r_u(i-1,1))q_d(i-1,1))p_s(i-1,1) \qquad (7.49) \\
&\quad (1-r_u(i-1,1))r_d(i-1,1)\mathbf{p}(i-1,1;000) \\
&\quad p_u(i-1,1)r_d(i-1,1)\mathbf{p}(i-1,1;110) \Big) \\
&\div \Big(E(i,1) - q_u(i-1,1)p_b(i-1,1) + r_u(i-1,1)\mathbf{p}(i-1,1;N00) \Big)
\end{aligned}
$$

$$
\begin{aligned}
r_u(i,1) &= \Bigg(r_i \left[\frac{p_u(i,1)}{r_u(i,1)} \{E(i-1,1) - q_u(i,1)p_b(i,1) + r_u(i,1)\mathbf{p}(i,1;N00)\} \right. \qquad (7.50) \\
&\quad \left. + \frac{q_u(i,1)}{r_u(i,1)} p_b(i,1) - \mathbf{p}(i,1;N_{i,1},00) - \mathbf{p}(i-1,1;000) - p_s(i-1,1) \right] \\
&\quad + r_i r_u(i-1,1)p_s(i-1,1) \\
&\quad + r_u(i-1,1)(1-q_d(i-1,1))\mathbf{p}(i-1,1;000) \Bigg) \\
&\div \Bigg(\frac{p_u(i,1)}{r_u(i,1)} \{E(i,1) - q_u(i,1)p_b(i,1) + r_u(i,1)\mathbf{p}(i,1;N_{i,1},00)\}
\end{aligned}
$$

$$
\begin{aligned}
& +\frac{q_u(i,1)}{r_u(i,1)}p_b(i,1) - \mathbf{p}(i,1;N_{i,1},00)\Bigg) \\
q_d(i,1) &= p_{i+1}p_b(i+1,1)\left(1 - \frac{p_s(i,2)+p_b(i+1,2)}{p_s(i,1)+p_b(i+1,1)}\right) && (7.51)\\
p_d(i,1) &= p_{i+1} + \Bigg((r_d(i+1,1) + q_u(i+1,1)(1-r_d(i+1,1))p_b(i+1,1) && (7.52)\\
& r_u(i+1,1)(1-r_d(i+1,1))\mathbf{p}(i+1,1;N00) \\
& r_u(i+1,1)p_d(i+1,1)\mathbf{p}(i+1,1;N-1,10)\Bigg) \\
& \div \Bigg(E(i,1) - q_d(i,1)p_s(i,1) + r_d(i,1)\mathbf{p}(i,1;000)\Bigg) \\
r_d(i,1) &= \Bigg(r_{i+1}\left[\frac{p_d(i,1)}{r_d(i,1)}\{E(i+1,1) - q_d(i,1)p_s(i,1) + r_d(i,1)\mathbf{p}(i,1;000)\}\right. && (7.53)\\
& \left.+\frac{q_d(i,1)}{r_d(i,1)}p_s(i,1) - \mathbf{p}(i,1;N00) - \mathbf{p}(i,1;000) - p_b(i,1)\right] \\
& +r_{i+1}r_d(i+1,1)p_b(i+1,1) \\
& + r_d(i+1,1)(1-q_u(i+1,1))\mathbf{p}(i+1,1;N00)\Bigg) \\
& \div \Bigg(\frac{p_d(i,1)}{r_d(i,1)}\{E(i+1,1) - q_d(i,1)p_s(i,1) + r_d(i,1)\mathbf{p}(i,1;000)\} \\
& +\frac{q_d(i,1)}{r_d(i,1)}p_s(i,1) - \mathbf{p}(i,1;000)\Bigg)
\end{aligned}
$$

$$
\begin{aligned}
q_u(i,2) &= \frac{r_u(i-1,1)p_s(i-1,1) + r_d(i,1)p_b(i,1)}{p_s(i-1,1)+p_b(i,1)} && (7.54)\\
p_u(i,2) &= r_u(i,2)\Bigg(\frac{1}{e_iP_u(i,2)} + \frac{P_d(i-1,2)+P_u(i,2)-1}{E(i,2)P_u(i,2)} && (7.55)\\
& -\frac{P_d(i-1,2)}{P_u(i,2)} - 1 - \frac{p_d(i-1,2)}{r_d(i-1,2)}\frac{P_d(i-1,2)}{P_u(i,2)}\Bigg) \\
r_u(i,2) &= \Bigg(r_i\left[\frac{(1-r_u(i-1,1))\mathbf{p}(i-1,1;000) + (1-r_d(i,1))\mathbf{p}(i,1;N00)}{\mathbf{p}(i-1,1;000)+\mathbf{p}(i,1;N00)}\right] && (7.56)\\
& \times\left[\frac{p_u(i,2)}{r_u(i,2)}\{E(i-1,2) - q_u(i,2)p_b(i,2) + r_u(i,2)\mathbf{p}(i,2;N00)\}\right. \\
& +\frac{q_u(i,2)}{r_u(i,2)}p_b(i,2) - \mathbf{p}(i,2;N00) - p_s(i-1,2) - \mathbf{p}(i-1,2;000) - 1
\end{aligned}
$$

$$
\begin{aligned}
& \quad +p_s(i-1,1)+\mathbf{p}(i-1,1;000)+p_b(i,1)+\mathbf{p}(i,1;N00)\Big] \\
& +r_u(i-1,2)(1-q_d(i-1,2))p_s(i-1,2) \\
& +\left[\frac{(1-r_u(i-1,1))\mathbf{p}(i-1,1;101)+p_u(i-1,1)\mathbf{p}(i-1,1;111)}{\mathbf{p}(i-1,1;101)+\mathbf{p}(i-1,1;111)}\right] \\
& \quad \times\left[\sum_{n=1}^{N}\Big\{\mathbf{p}(i-1,2;n10)+\mathbf{p}(i-1,2;n00)\Big\}\right. \\
& \quad \left.+r_u(i-1,2)\mathbf{p}(i-1,2;000)\right](1-p_u(i-1,1)) \\
& +\left[\frac{(1-r_d(i,1))\mathbf{p}(i,1;N-1,10)+p_d(i,1))\mathbf{p}(i,1;N-1,11)}{\mathbf{p}(i,1;N-1,10)+\mathbf{p}(i,1;N-1,11)}\right] \\
& \quad \times\left[\sum_{n=1}^{N}\Big\{\mathbf{p}(i-1,2;n10)+\mathbf{p}(i-1,2;n00)\Big\}\right. \\
& \quad \left.+r_u(i-1,2)\mathbf{p}(i-1,2;000)\right](1-p_u(i-1,1))\Big) \\
\div & \left(\frac{p_u(i,2)}{r_u(i,2)}\{E(i-1,2)-q_u(i,2)p_b(i,2)+r_u(i,2)\mathbf{p}(i,2;N00)\}\right. \\
& \quad +\frac{q_u(i,2)}{r_u(i,2)}p_b(i,2)-\mathbf{p}(i,2;N00)-1 \\
& \quad +p_s(i-1,1)+\mathbf{p}(i-1,1;000)+p_b(i,1)+\mathbf{p}(i,1;N00) \\
& \quad \left.+\sum_{n=1}^{N}\Big\{\mathbf{p}(i-1,2;n10)+\mathbf{p}(i-1,2;n00)\Big\}\right)
\end{aligned}
$$

$$
q_d(i,2) = \frac{r_u(i,1)p_s(i,1)+r_d(i+1,1)p_b(i+1,1)}{p_s(i,1)+p_b(i+1,1)} \tag{7.57}
$$

$$
\begin{aligned}
p_d(i,2) = r_d(i,2)&\left(\frac{1}{e_iP_d(i,2)}+\frac{P_d(i,2)+P_u(i+1,2)-1}{E(i+1,2)P_d(i,2)}\right. \\
&\left.-\frac{P_u(i+1,2)}{P_d(i,2)}-1-\frac{p_u(i+1,2)}{r_u(i+1,2)}\frac{P_u(i+1,2)}{P_d(i,2)}\right)
\end{aligned} \tag{7.58}
$$

$$
\begin{aligned}
r_d(i,2) = & \left(r_{i+1}\left[\frac{(1-r_u(i,1))\mathbf{p}(i,1;000)+(1-r_d(i+1,1))\mathbf{p}(i+1,1;N00)}{\mathbf{p}(i,1;000)+\mathbf{p}(i+1,1;N00)}\right]\right. \\
& \quad \times\left[\frac{p_d(i,2)}{r_d(i,2)}\{E(i+1,2)-q_d(i,2)p_s(i,2)+r_d(i,2)\mathbf{p}(i,2;000)\}\right. \\
& \quad +\frac{q_d(i,2)}{r_d(i,2)}p_s(i,2)-\mathbf{p}(i,2;000)-p_b(i+1,2)-\mathbf{p}(i+1,2;N00)-1 \\
& \quad \left.+p_s(i,1)+\mathbf{p}(i,1;000)+p_b(i+1,1)+\mathbf{p}(i+1,1;N00)\right] \\
& +r_d(i+1,2)(1-q_u(i,2))p_b(i+1,2)
\end{aligned} \tag{7.59}
$$

$$
\begin{aligned}
&+\left[\frac{(1-r_u(i,1))\mathbf{p}(i,1;101)+p_u(i,1)\mathbf{p}(i,1;111)}{\mathbf{p}(i,1;101)+\mathbf{p}(i,1;111)}\right] \\
&\quad\times\left[\sum_{n=1}^{N}\left\{\mathbf{p}(i-1,2;n10)+\mathbf{p}(i-1,2;n00)\right\}\right. \\
&\quad\left.+r_u(i-1,2)\mathbf{p}(i-1,2;000)\right](1-p_d(i+1,1)) \\
&+\left[\frac{(1-r_d(i+1,1))\mathbf{p}(i+1,1;N-1,10)}{\mathbf{p}(i+1,1;N-1,10)+\mathbf{p}(i+1,1;N-1,11)}\right. \\
&\quad\left.+\frac{p_d(i+1,1))\mathbf{p}(i+1,1;N-1,11)}{\mathbf{p}(i+1,1;N-1,10)+\mathbf{p}(i+1,1;N-1,11)}\right] \\
&\quad\times\left[\sum_{n=0}^{N-1}\left\{\mathbf{p}(i+1,2;n01)+\mathbf{p}(i+1,2;n00)\right\}\right. \\
&\quad\left.+r_d(i+1,2)\mathbf{p}(i+1,2;N00)\right](1-p_d(i+1,1))\Bigg) \\
\div\ &\left(\frac{p_d(i,2)}{r_d(i,2)}\left\{E(i+1,2)-q_d(i,2)p_s(i,2)+r_d(i,2)\mathbf{p}(i,2;000)\right\}\right. \\
&\quad+\frac{q_d(i,2)}{r_d(i,2)}p_s(i,2)-\mathbf{p}(i,2;000)-1 \\
&\quad+p_s(i,1)+\mathbf{p}(i,1;000)+p_b(i+1,1)+\mathbf{p}(i+1,1;N00) \\
&\quad\left.+\sum_{n=0}^{N-1}\left\{\mathbf{p}(i+1,2;n01)+\mathbf{p}(i+1,2;n00)\right\}\right)
\end{aligned}
$$

where

$$
\begin{aligned}
P_d(i,2) &= \frac{r_d(i,2)R(i,2)}{r_d(i,2)R(i,2)+q_d(R(i,2)-r_u(i,2)r_d(i,2)-r_u(i,2)p_d(i,2))} \\
P_u(i,2) &= \frac{r_u(i,2)R(i,2)}{r_u(i,2)R(i,2)+q_u(i,2)(R(i,2)-r_u(i,2)r_d(i,2)-r_d(i,2)p_u(i,2))}
\end{aligned}
$$

Chapter 8

Numerical Results

8.1 Introduction

The derivation of the decomposition equations for the multiclass processing line — done in Chapter 7 — constitutes only the first half of the decomposition of the original network. It is now necessary to develop an algorithm that will solve the equations. This chapter presents an algorithm, classifies its effectiveness, and provides numerical data for determining its accuracy.

We make no claims as to the convergence reliability of the algorithm, other than to state that we find that it converges well for lines with five or fewer processing machines, and that when it converges, it always converges to the same values, regardless of the starting point. Furthermore, the results to which the algorithm converges compare well with discrete event simulation in general, and the cases where the results are not as accurate can be classified, and predicted in advance. We are unable to offer a proof of convergence, and we provide no bounds on the errors of the performance measures that the algorithm calculates. Furthermore, we are unable to assert that the algorithm, is the fastest or most reliable. As is often the case with initial numerical implementations, we expect that the algorithm will be improved over the course of the next several years. Our main purpose is to test the validity of the decomposition methodology, which we see as the main contribution of this thesis.

8.2 The MDDX Algorithm

In this section we present the MDDX algorithm for solving the decomposition equations derived in Chapter 7. The basic idea of the algorithm is a generalization of that for the single part case of the DDX algorithm. In this case, we first sweep down the line, calculating the upstream two-machine parameters for type one, using the parameters of the previous two machine line, and then sweep up the line to calculate the downstream two-machine line parameters. We then repeat the process for each successive part type.

Step 0: Initialization

initialize upstream parameters

for i = 1 to NumMachines **do**;
  for j = 1 to NumParts**do** ;
    $p_u[i][j] = p[i][j];$
    $r_u[i][j] = r[i][j];$
    $I_u[i][j] = p_u[i][j]/r_u[i][j];$
  end;
end;
initialize downstream parameters

for i = 0 to NumMachines-1 **do** ;
  for j = 1 to NumParts **do**;
    $p_d[i][j] = p[i+1][j];$
    $r_d[i][j] = r[i+1][j];$
    $I_d[i][j] = p_d[i][j]/r_d[i][j];$
  end;
end;
initialize boundary conditions

for j = 1 to NumParts **do**;
  $p_u[0][j] = p[0][j];$
  $r_u[0][j] = r[0][j];$
  $q_u[0][j] = 0;$
  $p_d[NumMachines+1][j] = p[NumMachines+1][j];$
  $r_d[NumMachines+1][j] = r[NumMachines+1][j];$
  $q_d[NumMachines+1][j] = 0;$
end;

While the termination criterion of step 5 is not met, do steps one through four, in order, until the termination criterion of step 5 is met, or until the algorithm iterates for a preset number of iterations.

Step 1: Upstream Sweep for Type One
for i = 1 to NumMachines **do**;
  Evaluate Two Machine Line $L(i-1,1)$;
  Calculate $q_u[i][1]$ using Equation (7.48)
  Calculate $p_u[i][1]$ using Equation (7.49)

Calculate $r_u[i][1]$ using Equation (7.50)
end;

Step 2: Downstream Sweep for Type One
for i = NumMachines-1 to 0 **do**;
Evaluate Two Machine Line $L(i+1,1)$;
Calculate $q_d[i][1]$ using Equation (7.51)
Calculate $p_d[i][1]$ using Equation (7.52)
Calculate $r_d[i][1]$ using Equation (7.53)
end;

Step 3: Upstream Sweep for Type Two
for i = 1 to NumMachines **do**;
Evaluate Two Machine Line $L(i-1,2)$;
Calculate $q_u[i][2]$ using Equation (7.54)
Calculate $p_u[i][2]$ using Equation (7.55)
Calculate $r_u[i][2]$ using Equation (7.56)
end;

Step 4: Downstream Sweep for Type Two
for i = NumMachines-1 to 0 **do**;
Evaluate Two Machine Line $L(i+1,2)$;
Calculate $q_d[i][2]$ using Equation (7.57)
Calculate $p_d[i][2]$ using Equation (7.58)
Calculate $r_d[i][2]$ using Equation (7.59)
end;

Step 5: Evaluate Stopping Criterion

Terminate the algorithm when the maximum value of

$$\|E(i,j) - E(0,j)\|$$

for $i = 1, \ldots, NumMachines$ is less than some pre-specified ϵ *for each* part type j.

The MDDX algorithm is similar to earlier implementations in that it begins by sweeping down the line, using parameter values from previous iterations to calculate parameters at the current iterate, and then sweeping up the line in the reverse direction. The termination criteria was considered met when Conservation of Flow held among all two-machine lines to within a certain tolerance. This method exploits the recursive nature of the interruption and resumption of flow equations, and has proved to be the most effective algorithm in the one-part case. Our extension is to the case with multiple parts, where we extended the methodology by taking a full iteration to be one where we sweep down and up the line using the Type 1 equations, and then sweep down and up the line using the Type 2 equations, and then repeat until Conservation of Flow holds for both part types to within a certain preset tolerance.

8.3 Randomly Generated Cases

Since we are unable to offer a proof of convergence, and cannot give bounds on the accuracy of the performance measures that the algorithm calculates, we follow the procedure of Burman [7], and test the algorithm on multiple randomly-generate cases, where the parameters of the random systems are within certain pre-set tolerances. We then present comparisons with discrete-event simulations, and report on the accuracy of the algorithm with respect to the simulation results for production rate and average buffer levels.

We have found that the algorithm seems to converge most reliably when the mean time to fail (MTTF) and mean time to repair (MTTR) of all the machines in the network — supply, processing, and demand — are of the same order of magnitude as each other, and the MTTR one order of magnitude less than the MTTF. Thus, a typical five-machine, two part-type line might consist of one where the MTTR for the supply,processing, and demand machines were 10, the MTTF for the supply and processing machine were 100, and the MTTF for the demand machines were 20.

We have found that the algorithm is does not converge reliably when the MTTF and MTTR are radically different from one machine to the next, and if the algorithm does converge in these cases, it is usually not accurate. In physical terms, this corresponds to cases where adjacent machines fail and are repaired at radically different rates. That the algorithm would not perform well in these circumstances is understandable for the following reason. Recall that the idea behind the decomposition is to break the network up into tractable two-machine lines, and then set the parameters of the upstream and downstream machines of the sub-line to capture the behavior both up and downstream of the intermediate buffer such that an observer in the intermediate buffer of the two-machine line would be unable to tell if he was in a buffer in the real line, or not. Flow can stop entering the intermediate buffer for a number of reasons. In the case of a type one buffer, it can be because the processing machine immediately upstream of the buffer has failed, or because the processing machine immediately upstream of the buffer has become starved due a disruption further upstream. Our aim is to capture the behavior of both types of events using a single pseudo-machine that has geometrically distributed up and down times. Suppose that the processing machine immediately upstream of the buffer — call it M_i — has a repair probability of 0.1, but the processing machine immediate upstream of it — call it M_{i-1} — has a repair probability of 0.001. When a failure occurs of the sub-line pseudo-machine — that is, when the observer in the buffer sees that flow has stopped, but that the intermediate buffer is not full — he begins to count time steps until flow resumes. If flow resumes within ten to fifteen time steps, he would be unable to tell if the flow stoppage was due to machine M_i or M_{i-1} failing. However, if it takes two-hundred time steps for flow to resume, then the observer can be very sure that the flow stoppage was *not* the result of machine M_i failing. That is, he is getting more information than we expect him to have. In this case, the upstream sub-line pseudo machine is no longer acting as the aggregate behavior of the upstream line.

In light of this explanation, the random cases which we generate have machines that have similar, though not identical, characteristics. We allow for the machines to have different isolated efficiency rates, but we do not generate lines where one machine is an extreme bottleneck. In general, the isolated efficiencies of the supply and processing machines vary from 0.85 and 0.95, with occasional machines having isolated efficiencies between 0.8 and 0.85, and 0.95 and one. For the demand machines, we typically generate repair probabilities that are of the same order of magnitude as those generated for the supply and processing machines, but the failure probabilities of the demand machines are higher; rather than being an order of magnitude smaller than the repair probabilities, they are of the same order of magnitude. This ensures that the demand rates for part types one and two are individually below the capacity of the line. This is because a system in which the type-one demand machine has an isolated efficiency similar to that of the other machines in the line tends to be uninteresting, as the line spends all of its time producing type one products, and none producing type two products. In this case, a two-part processing line with five processing machines behaves like a single-part transfer line with seven machines (where the additional machines are the supply machine upstream, and the demand machine downstream). This case has been thoroughly investigated, as discussed in Chapter 1.

8.4 Random Case Generation Procedure

We generated five sets of 300 random lines — where the first set consisted of a line with two processing machines, the second of three processing machines, and so on — for a total of 1,500 randomly generated processing lines. Each set of 300 random lines was further split into three subsets. For a given line configuration, the first 100 cases generated (cases 1 to 100) were of lines where the demand for part types one and two were roughly the same. The second 100 cases generated (cases 101 to 200) were of lines where the demand for part type one exceeded that of part type two by up to roughly 30%. The remaining 100 cases (cases 201 to 300) were of lines where the demand for part type two exceeded that of part type one by up to roughly 30%. In all three cases, we ensured that the demand rate for types one and two were such that, combined, the processing line would *not* have the capacity to meet demand for both type one and Type 2 parts. The reason for this is because if the line does have the capacity to meet demand for both part types, then the estimation process is trivial; production rate will equal demand, and all intermediate buffers will be nearly full.

8.4.1 Machine Parameter Generation

As discussed in Section 8.3, we consider systems where the supply, processing, and demand machines have similar repair rates, but where the failure rates of the demand machines are about one order of magnitude higher than those of the supply and processing machines. For the repair probabilities, we sought random values that would be at most +/- 10% of the mean, and failure probabilities that would at at most +/- 50% of the mean. For the machines modeling the supply process, and the internal processing machines, the repair probabilities were generated from a tight triangular distribution (see [42], pages 341-342) with mean 0.1 and SCV 0.01, giving a range of values from 0.08 to 0.12. The failure probabilities (the p parameters) were generated from a tight triangular distribution with mean 0.01 and SCV 0.2, giving a range of values from 0.05 to 0.015. Using these values gives an average isolated efficiency of approximately 0.91, taking values ranging from 0.8 to 1.0. The repair and failure probabilities for the demand machines were generated from similar triangular distributions, and are summarized in Table 8.4.1. There, r refers to the repair probability of *both* demand machines, p_1 is the failure probability of the type-one demand machine, p_2 is the failure rate of the type-two demand machines, and $\overline{E}_1$ and $\overline{E}_2$ are the average demand rates for part types one and two, respectively.

The parameters given in Table 8.4.1 will generate isolated efficiencies for the demand machines that cover a relatively wide range of values.

Case	Parameter	Mean	Expected Demand Rate
1-100	r	0.1	
	p_1	0.08	$\overline{E}_1 = 0.55$
	p_2	0.08	$\overline{E}_2 = 0.55$
101-200	r	0.1	
	p_1	0.08	$\overline{E}_1 = 0.55$
	p_2	0.15	$\overline{E}_2 = 0.40$
201-300	r	0.1	
	p_1	0.15	$\overline{E}_1 = 0.40$
	p_2	0.08	$\overline{E}_2 = 0.55$

Table I: Failure and repair parameters generated for the demand machines in random cases.

For Cases 1-100, the average demand rate for types one and two will be around 0.55, but can be as low as 0.5, and as high as 0.6. This ensures that the line can never have the capacity to meet demand for type two, because the total demand rate for types one and two will, on average, exceed 1.0. Even if all the machines in the line were perfectly reliable, this production rate could not be achieved. In general, unless the type one buffers are very small, the line will have the capacity to meet demand for Type 1 parts. Because of this, the problem of approximating type one production rates is essentially trivial; the production rate will be that of demand, less a small amount due to blockages and starvations. Determining the production rate for Type 2 parts is non-trivial, as they are always less than demand. A typical type two production rate for cases 1-100 was about 0.3. This declines as the line gets longer, or as the type-two buffer sizes get smaller. It increases as the number of processing machines decreases, or the type-two buffer sizes increase. We regard the accuracy of the decomposition to predict this production rate as the main result of this work.

For Cases 101-200, the average demand rate for types one and two differ; on average, the demand for Type 1 parts remains high (between 0.5 and 0.6), while that of Type 2 parts is lowered, taking values between 0.35 and 0.45, with an average demand rate of 0.4. As in the previous case, these numbers were chosen so that the line would not, in general, have the capacity to meet demand for part type two.

For Cases 201-300, we reversed the demand rates for types one and two set in Cases 101-200; demand for type one is now lower, taking an average value of 0.4 on a range from 0.35 to 0.45, while demand for type two is higher, taking an average value of 0.55 on a range from 0.5 to 0.6. As with the first 200 cases, the line will, on average, not have the capacity to meet demand for part type two.

8.4.2 Buffer Size Generation

After generating the machine parameters using the procedure described in Section 8.4.1, we used the parameters to generate the buffer sizes.

For the supply buffers ($B_{0,1}$ and $B_{0,2}$), we set the buffer sizes to be the inverse of the repair probabilities, rounded up to the nearest integer. The usually gave buffer sizes of ten or eleven. We made this simplification for several reasons. First, we found that both the convergence reliability of the algorithm, as well as the accuracy were only minimally affected by changing the size of the supply machine buffers. Second, we noticed that regardless of the size of the supply machine output buffers, the buffers were almost always full. This is because although the supply machines have tended to have isolated efficiencies that were close to those of the processing machines in the line, the actual production rates of the supply machines for their individual part types are much higher than those of the processing machines. Put another way, since the supply machines are assumed to never be starved, and have the same repair and failure probabilities as the processing machines, but do not have to share capacity between two part types as the processing machines do, their effective capacities to produce their given part types are much higher than the respective capacities of the processing machines. The choice of the buffer size being the inverse of the repair probability is not entirely arbitrary, however. Since we use the supply machines to model randomness in supply, we expect that there will be starvations occasionally. By setting the buffer size to be slightly larger than the MTTR of the upstream machine, we will generally have enough material in the supply buffer to ride out an interruption due to a failure, but not always.

To compute the internal buffer sizes — the downstream buffers for the processing machines — we used a method that is a variation on the procedure used by Burman [7]. First, we generated K random numbers $y_1, \ldots, y_K$, where y_i was calculated by first getting two uniform $0-1$ random variables U_1 and U_2, and setting

$$y_i = 1 + \lfloor 3 * U_1 \rfloor + \lfloor 3 * U_2 \rfloor. \tag{8.1}$$

Then we generated an additional $2K$ uniform $0-1$ random variables, denoted by $U_{i,1}$ and $U_{i,2}$, for $i = 1, \ldots, K$. Finally, for each $i = 1, \ldots, K$, we calculated the buffer sizes for types one and two by

$$N_{i,1} = \max\left\{\frac{1.3 * U_{i,1}}{r_i}, \frac{1.3 * U_{i,1}}{r_{i+1}}, \frac{y_i}{r_i}, \frac{y_{i+1}}{r_{i+1}}\right\} \tag{8.2}$$

$$N_{i,2} = \max\left\{\frac{2 * U_{i,1}}{r_i}, \frac{2 * U_{i,1}}{r_{i+1}}, \frac{y_i}{r_i}, \frac{y_{i+1}}{r_{i+1}}\right\} \tag{8.3}$$

By choosing the buffer sizes in this way, we generate systems where a given buffer size is randomly determined, taking into account the repair probabilities of the machines up- and downstream of it. For Type 1 parts, it chooses buffer sizes that are between one-half and four times average number of Type 1 parts produced during an adjacent machine's downtime, if the machine only works on type one. For Type 2 parts, the procedure generates buffer sizes that are between one and four times the average number of Type 2 parts produced during a downtime, if the machine had spent all that time working on Type 2 parts. We have set the buffer sizes for Type 2 parts to be higher, on average, because we expect there to be much more variability in their production. Therefore, when a processing machine is able to produce Type 2 parts, we want it to produce then with a minimal chance of being blocked, as such production time is probably relatively rare. Finally, by allowing the buffer size calculation to rely on the repair probabilities, we ensure that the downstream buffers after the K^{th} processing machine are calculated in the same manner as those upstream of it. We do so because we set the isolated efficiencies of the demand machines to be lower than those of the processing machines in the line by setting the failure probabilities of the demand machines to be higher. Since this is an artificial construction, we do not wish it to influence the systems we generate randomly.

8.4.3 Computation of Approximation Error

With the above parameters, the cases were randomly generated using a program written in C. A script then simulated each case, and recorded the results, using a discrete-event simulator written in C. Each simulation consisted of 30 independent simulations runs of 1,500,000 time periods each, where the first 500,000 time periods were discarded to ensure data was only collected on a system in steady state. The length of the simulation runs was chosen so that we could be sure that the transient period did not affect the results and that the period of data collection was long enough to give confidence intervals that gave at least three significant digits of accuracy to the right of the decimal place. To do so, we followed a simple procedure of picking a simulation length and transient period at random, and the doubling it until the results from one iteration to the next were essentially identical. The performance measures simulated were the production rates of part types one and two, and the average inventory levels for both part types in all buffers. Approximate 95% confidence intervals (see [2, 42]) were calculated on all performance measures, and typically production rate point estimates were accurate to three digits to the right of the decimal point, and average buffer levels accurate to two digits to the right of the decimal point. We then ran the algorithm to estimate the same performance measures, and calculated approximation error using the the following two metrics.

For production rates, we calculated the percent error of the approximated production rate from the simulated production rate in the following manner.

$$\% \text{ Error} = 100 \times \frac{E_{decomp} - E_{sim}}{E_{sim}}. \tag{8.4}$$

This metric is standard in the literature, and provides easy recognition of whether the approximation is under- or over-estimating the simulated production rate.

For average buffer levels, we calculated the percent error of the approximated average buffer level from that of the simulated buffer level using a metric that took the difference between the approximated average buffer level ($\overline{N}_{decomp}$) and the simulated average buffer level ($\overline{N}_{sim}$), divided the difference by the buffer size (N), and multiplied the quotient by 100. In other words,

$$\% \text{ Error} = 100 \times \frac{\overline{N}_{decomp} - \overline{N}_{sim}}{N}. \tag{8.5}$$

This measurement is standard in the literature cited, and justified with the following argument. In the case of a two-machine line, one can show that if the machine parameters are reversed (that is, if the upstream and downstream machines swap failure and repair probabilities), then the average buffer level will also invert. For example, if the buffer size is 10, and the average buffer level is 4.2, then swapping the parameters will yield an average buffer level of 5.8. This symmetry is also exhibited in one-part long lines; if the line is reversed, the buffer levels invert. This phenomenon is a result of having finite buffers. Because we wish the percent error of the buffer level approximations to be from simulation to be independent of anything other than the performance of the line, we use the approximation given in (8.5). Under this metric, both the forward and backward lines give the same percent error.

8.5 Non-Homogeneous Line Tests

Using the generation procedure described in Section 8.4, this section presents the numerical results for lines with two, three, four, five, and six processing machines, respectively. In general, we find the decomposition algorithm to be accurate in estimation of both production rates and average inventory levels.

8.5.1 Production Lines with Two Processing Machines

The first system consisted of a production line with two internal processing machines, two machines capturing upstream supply uncertainty, and two machines capturing the behavior of downstream demand uncertainty. Following the procedure described in Section 8.4, 300 random cases were generated and simulated. The algorithm was then run, and the results were compared to simulation. For the case of a production line with two processing machines, the algorithm converged 100% of the time, usually in less than 20 iterations. On a 450Mhz Pentium II, the results appeared instantaneously.

Production Rates

The percent errors calculated for all 300 cases are summarized in Figure 8-1. In addition, The average absolute percent error for all 300 cases — that is, the sum of the absolute value of the percent errors, divided by the number of cases — was calculated to be 0.15% for Type 1 parts, and 2.25% for Type 2 parts. Broken down into the three sub-cases, the average absolute percent errors for production rates are given in Table III. The impulse plots in Figure 8-1 show the distribution of percent errors for production rate of type two parts. We see several things immediately. First, the behavior of the algorithm varies depending on the demand rates for types one and two. In the case where the demand rates for both are high — cases 1 - 100 in Figure 8-1— the algorithm tends to underestimate the production rate. In the case where demand is high for type one, and lower for type two — cases 101 - 200 in Figure 8-1 — the algorithm tends to overestimate the production rate of Type 2 parts. Last, in the case where demand for type one is low, and type two is high — cases 201 - 300 in Figure 8-1 — the algorithm is at its most accurate, tending to slightly overestimate the type two production rate. We propose that the difference in demand rates for types one and two affects the line in the following ways.

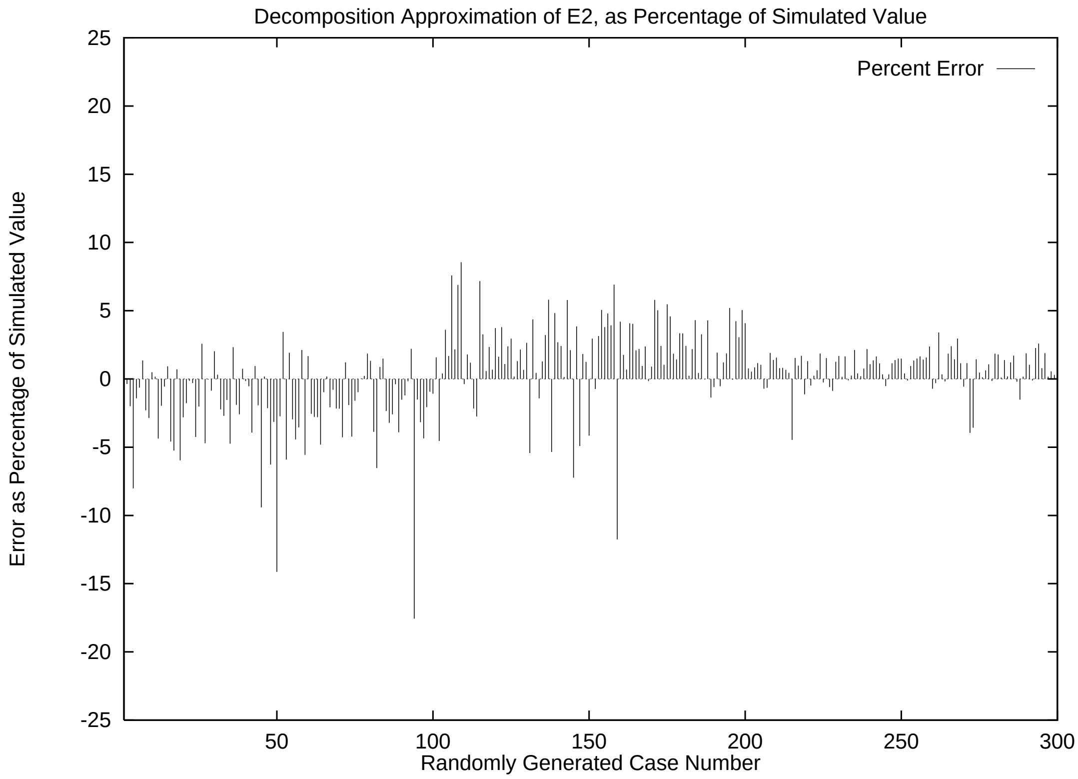

Figure 8-1: The errors in the decomposition approximation for Type 2 production rates in a production line with two processing machines.

When the demand rates for both part types are high (cases 1-100) the line will spend much of the time working on Type 1 parts. Only in cases when the a processing machine is blocked or starved for Type 1 parts will it work on Type 2 parts. Since the demand rate for Type 1 parts is relatively high, we expect that a blockage propagating back from a lack of demand will not occur too frequently. Because of this, we expect that the production of part type two is highly variable. That is, the line spends long periods making type one parts in between short periods making Type 2 parts. As explained in Section 8.3, we do not expect the algorithm to perform well under situations where machines go up and down at very different rates. In this case, an observer in the part type two finished goods inventory will see exactly such behavior.

Another feature of the approximations for the case where the demand rates for both part types are similar, and high, is that the algorithm underestimates the production rate three times as often as it overestimates it. We attribute this again to the variability in the production of Type 2 parts due to the high demand for Type 1 parts. This translates into a processing machine switching from type two production to type one production more often than in a system with lower demand for type one. This, in turn, has the effect of artificially raising the repair probabilities of the pseudo-machines in the two-machine lines, which in turn lowers the production rate estimates of those lines.

When the demand rate for Type 1 parts is high, but that of Type 2 parts is low (cases 101-200), we find that the approximation tends to overestimate the production rate of Type 2 parts much more often than it underestimates it, by a ratio of almost ten to one. Again, we attribute this error to a large amount of variability in the production of Type 2 parts.

When the demand for type one is low, but demand for type two is high, the algorithm gives the best approximations. We find that it tends to overestimate the production rate for type two parts much more often than it underestimates it, but does so by a small amount. We attribute this to a reduction in the variability discussed in the previous two cases that occurs for the following reason. Since the demand rate for type one parts is relatively low, there will be more blockage of Type 1 part production, leading to longer periods in which Type 2 parts can be produced. This will have the effect of smoothing out the disturbances in type two production, leading the simulated system to appear more like the system that the decomposition procedure is supposed to be estimating.

The second observation is that the algorithm performs badly in some cases. Specifically, there are three cases where the algorithm underestimates the production rate by 10%, or more. These are cases 50, 94, and 159. In all three cases, the random production line generated had buffer size distributions that contained large differences. These are summarized in Table II. In all three cases, the last buffer in the line is much larger than the internal buffer. In cases 50 and 159, the finished goods inventory size is about twice that of the intermediate buffer, while in case 94 it is approximately three times as high. The algorithm performs worst for case 94, underestimating the simulated production rate by almost 17% (0.3031 vs. 0.3677). For case 50, it underestimates the simulated production rate by 14% (0.2721 vs. 0.3169), and for case 159 it underestimates by 11% (0.2841 vs. 0.3220).

Case No.	N_{11}	N_{12}	N_{21}	N_{22}
50	14	12	21	21
94	11	11	31	31
159	16	12	24	41

Table II: Buffer sizes of the systems for which the algorithm performs badly in estimating type-two production rates, for the two-processing- machine line

Cases 1-100

Production Rate Percent Errors			
E_1		E_2	
0.213777		2.629042	
Part Type One		Part Type Two	
Buffer	% Error	Buffer	% Error
$B_{0,1}$	6.18	$B_{0,2}$	3.94
$B_{1,1}$	4.23	$B_{1,2}$	4.72
$B_{2,1}$	1.27	$B_{2,2}$	7.06

Cases 101-200

Production Rate Percent Errors			
E_1		E_2	
0.193178		2.977209	
Part Type One		Part Type Two	
Buffer	% Error	Buffer	% Error
$B_{0,1}$	6.11	$B_{0,2}$	3.56
$B_{1,1}$	4.17	$B_{1,2}$	7.23
$B_{2,1}$	1.17	$B_{2,2}$	11.11

Cases 201-300

Production Rate Percent Errors			
E_1		E_2	
0.041258		1.150070	
Part Type One		Part Type Two	
Buffer	% Error	Buffer	% Error
$B_{0,1}$	4.24	$B_{0,2}$	5.22
$B_{1,1}$	2.18	$B_{1,2}$	5.22
$B_{2,1}$	0.35	$B_{2,2}$	8.69

Table III: Aggregate absolute percent errors for production rates and average inventory levels for the two-processing-machine line.

Average Buffer Levels

The estimates for average buffer levels for the two-processing machine line are given in Table III. Estimates for Type 1 production rates are within 0.25% Estimates for production rates for Type 2 and Type 2 are within 3% of simulation. The worst estimates are for the finished goods inventory buffers ($B_{3,2}$). This is due to two factors. First, the numbers being estimated are quite small. A typical average is inventory level is between 0.5 and 2. This small number is due to the fact that demand far outstrips capacity; most of the time, when a part enters the finished goods inventory, it is taken immediately by demand. That is, we expect the buffer to be empty most of the time. However, there will be some epochs when he demand machine will fail, and the buffer will fill up. This will be a temporary event, but it leads to much greater variability in the finished goods inventory. This variability is the second reason that we believe the estimates are worse than the others.

8.5.2 Production Lines With Three Processing Machines

The second system we generated consisted of a production line with three internal processing machines, two machines capturing upstream supply uncertainty, and two machines capturing the behavior of downstream demand uncertainty. Following the procedure described in Section 8.4, 300 random cases were generated and simulated. The algorithm was then run, and the results were compared to simulation. For the case of a production line with two processing machines, the algorithm converged 100% of the time, usually taking between 20 and 40 iterations. On a 450Mhz Pentium II, the results again appeared instantaneously.

Production Rates

The percent errors calculated for all 300 cases are summarized in Figure 8-2. The absolute percent error for all 300 cases — that is, the sum of the absolute value of the percent errors, divided by the number of cases — was calculated to be 0.12% for Type 1 parts, and 2.48% for Type 2 parts. Broken down into the three sub-cases, the average absolute percent errors for production rates are given in Table V.

We observe that the classification of estimates of production rate made in the cases with two processing machines still holds. That is, for the first 100 cases, the algorithm tends to underestimate the simulated production rate much more often than it overestimates it. In the second 100 cases it tends to overestimate the production rate much more often than it underestimates it, and in the last 100 cases, the algorithm is at its most accurate, but still tends to overestimate the production rate slightly.

As we did with the two-processing machine line, we would like to be able to classify the systems in which the approximation gives very bad estimates for type two production rate (i.e. worse than 10% error). The cases where the algorithm

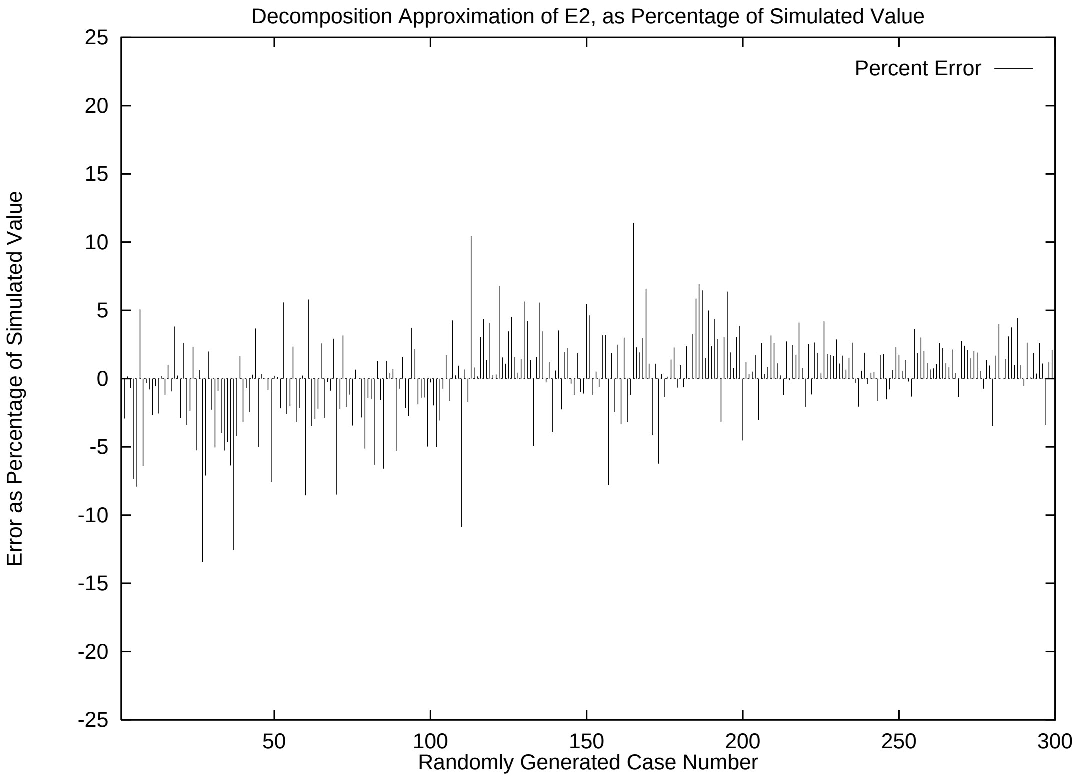

Figure 8-2: The errors in the decomposition approximation for Type 2 production rates in a production line with three processing machines.

performed this badly for the three-processing machine line are given in Table 8.5.2. We found nothing particular about the isolated efficiencies of the supply, demand, or processing machines in these cases, and must conclude that the error is a result of the buffer sizes. In each of the cases, we note that there is one buffer that is much smaller than the others. We observed the same phenomenon in the cases with two processing machines that resulted in bad underestimates. As before, we are led to conclude that the presence of a buffer size that is much smaller than those adjacent to it leads to an increase in the variability of the flow of material into the finished goods buffer, leading to the error in estimating the production rate.

In the cases where the algorithm severely overestimated the production rate (cases 113 and 165), we were unable to identify an aspect of the system that was the probable cause of the error.

Average Buffer Levels

For the three-processing machine system, the estimates for average inventory levels tend to be quite good. At their worst, a twelve percent error was recorded for the finished goods inventory buffer for Type 2 parts (buffer $B_{3,2}$), which we attribute to the variation observed in this buffer.

8.5.3 Production Lines With Four Processing Machines

The third system we generated consisted of a production line with four internal processing machines, two machines capturing upstream supply uncertainty, and two machines capturing the behavior of downstream demand uncertainty. Following the procedure described in Section 8.4, 300 random cases were generated and simulated. The algorithm was then run, and the results were compared to simulation. For the case of a production line with four processing machines, the algorithm converged 100% of the time, usually taking between 20 and 40 iterations, but sometimes taking as many as 300 iterations. On a 450Mhz Pentium II, the results appeared instantaneously in most cases, and with a slight but noticeable delay in the worst cases.

Production Rates

The average absolute percent error for all 300 cases — that is, the sum of the absolute value of the percent errors, divided by the number of cases — was calculated to be 0.13% for Type 1 parts, and 3.21% for Type 2 parts. Broken down into the three sub-cases, the average absolute percent errors for production rates are given in Table VII.

When it comes to classifying the over- or underestimation tendencies of the algorithm, the observations made for the two- and three-processing machine lines begins to break down. Specifically, we note that the algorithm begins to overestimate the

Case No.	N_{11}	N_{12}	N_{21}	N_{22}	N_{31}	N_{32}
27	10	10	30	39	34	34
37	26	30	11	11	31	31
110	35	35	12	12	28	36

Table IV: Buffer sizes of the systems for which the algorithm performs badly in estimating type-two production rates, for the three-processing- machine line

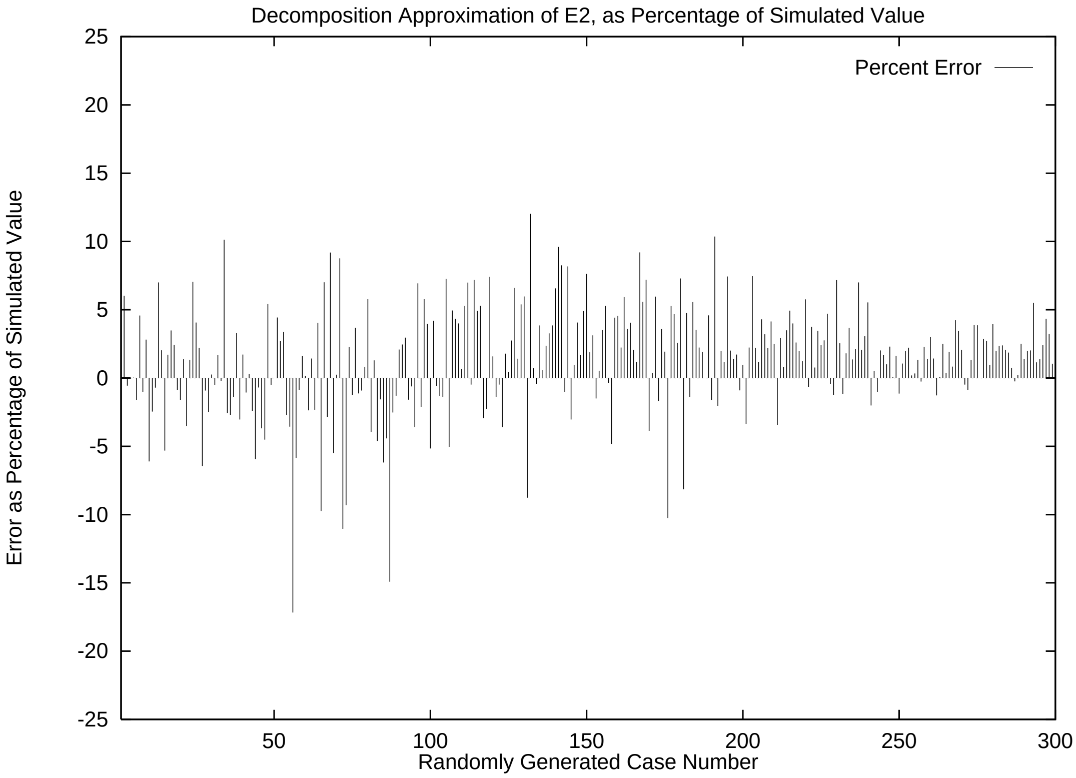

Figure 8-3: The errors in the decomposition approximation for Type 2 production rates in a production line with four processing machines.

Cases 1-100

Production Rate Percent Errors			
E_1		E_2	
0.157256		2.925625	
Part Type One		Part Type Two	
Buffer	% Error	Buffer	% Error
$B_{0,1}$	6.24	$B_{0,2}$	3.93
$B_{1,1}$	4.15	$B_{1,2}$	4.23
$B_{2,1}$	4.10	$B_{2,2}$	4.13
$B_{3,1}$	1.08	$B_{3,2}$	6.75

Cases 101-200

Production Rate Percent Errors			
E_1		E_2	
0.160494		2.843376	
Part Type One		Part Type Two	
Buffer	% Error	Buffer	% Error
$B_{0,1}$	6.27	$B_{0,2}$	3.70
$B_{1,1}$	4.14	$B_{1,2}$	6.06
$B_{2,1}$	3.95	$B_{2,2}$	8.39
$B_{3,1}$	1.00	$B_{3,2}$	11.86

Cases 201-300

Production Rate Percent Errors			
E_1		E_2	
0.039362		1.662800	
Part Type One		Part Type Two	
Buffer	% Error	Buffer	% Error
$B_{0,1}$	4.29	$B_{0,2}$	5.24
$B_{1,1}$	2.16	$B_{1,2}$	4.16
$B_{2,1}$	2.13	$B_{2,2}$	5.77
$B_{3,1}$	0.30	$B_{3,2}$	9.00

Table V: Aggregate absolute percent errors for production rates and average inventory levels for the three-processing-machine line.

production rate in the case where the demand rates for the parts are the same almost as much as it underestimates it. We believe that the lengthening of the line leads to this overestimation; due to upstream variability, the algorithm is led to believe that production is met less often that it really is.

We found in systems with two and three processing machines that cases of severe underestimation of type-two production rates could be correlated with cases where there was one buffer whose size was considerably smaller than that of its neighbors. There were four systems with four processing machines for which the production rate was underestimated by 10%, or more. These were cases 56,72,87, and 176. As in the other cases where the algorithm produced such a bad underestimate, we found nothing particular about the isolated efficiencies or the failure and repair parameters of the systems generated. We are therefore led to conclude that the source of the underestimate comes from the buffer sizes. Table 8.5.3 tabulates these buffer sizes. We note that in every case, there is one buffer that is considerably smaller than the rest. In case 56, it is $B_{1,2}$, and in cases 72 and 176 it is $B_{2,2}$, while in case 87 it is $B_{3,2}$.

Average Buffer Levels

We again find, despite the lengthening of the production line, that the estimates for average inventory levels are quite accurate. As before, the estimate for the part type two finished goods inventory is worst, but we note that it is actually a small improvement over the estimates in the system with three processing machines. This runs contrary to expectation. We expect that as the line gets longer, the variability of flow of material into the type two finished goods inventory buffer would increase, leading to worse estimates.

8.5.4 Production Lines With Five Processing Machines

The fourth set of systems we generated consisted of a production line with five internal processing machines, two machines capturing upstream supply uncertainty, and two machines capturing the behavior of downstream demand uncertainty. Following the procedure described in Section 8.4, 300 random cases were generated and simulated. The algorithm was then run, and the results were compared to simulation. For the case of a production line with two processing machines, the algorithm converged 90% of the time (269 out of 300 cases), usually taking between 200 and 60 iterations, but sometimes taking as many as 1000 iterations when it converged. On a 450Mhz Pentium II, the results still appeared almost instantaneously in most cases. In the worst cases, it seemed to take up to two seconds.

We are unable to determine why the algorithm did not converge when it did not. Investigation of the cases in which convergence failed has not revealed any insights. There are no features of the lines — such as overly-high or low isolated efficiencies, unbalances in the supply machines, or strangely distributed buffer sizes — that would lead us to be able to classify cases of non-convergence beforehand. However, we expect that the algorithm's behavior is largely determined by the structure of the line. If there is a great deal of inherent variability in the line (i.e. if there are very different buffer sizes, or if the machines in the line go up and down at different rates), we expect that the algorithm will be less likely to converge, and when it does converge, it will be less accurate.

Production Rates

The average absolute percent error for the 269 cases where the algorithm converged was 0.13% for Type 1 parts, and 3.30% for Type 2 parts. Broken down into the three sub-cases, the average absolute percent errors for production rates are given in Table IX.

In the production line with five machines, we note that the distinction in over- and underestimation of the type two production rate has almost completely disappeared. In most instances when the algorithm converged, it overestimated the type two production rate.

For the systems generated with five processing machines, there were only two cases that produced an underestimate of more than 10%. These were cases 91 and 200. As before, we found nothing in the isolated efficiencies or machine parameters that could be blamed for this underestimate, so we concentrate on the buffer sizes. For these two cases, these are given in Table VIII. As with the other instances of sharp underestimation, both the cases exhibit buffer sizes where one buffer is sized significantly smaller than the others. As explained previously, we believe that this leads to greater variability in the flow of material into the type two finished goods inventory, and thus to a misestimation of the production rate.

Average Buffer Levels

For the systems generated with five processing machines, we find that the estimates for average inventory levels are good. As with the previous cases, they are worst for the type two finished goods inventories. As noted in the four-processing machine case, the results actually seem to be improving as the line gets longer. We attribute this to the fact that the algorithm has not converged one-hundred percent of the time. Since we expect non-convergence in cases where the underlying system sees a lot of variability — precisely the cases where the decomposition approximation is expected to produce bad results — the removal of these cases leads to an improvement in the average inventory level estimates.

Case No.	N_{11}	N_{12}	N_{21}	N_{22}	N_{31}	N_{32}	N_{41}	N_{42}
56	11	11	33	33	30	30	24	39
72	19	19	11	11	21	21	24	38
87	12	11	11	42	11	11	10	25
176	11	26	11	11	11	16	31	31

Table VI: Buffer sizes of the systems for which the algorithm performs badly in estimating type-two production rates, for the four-processing- machine line

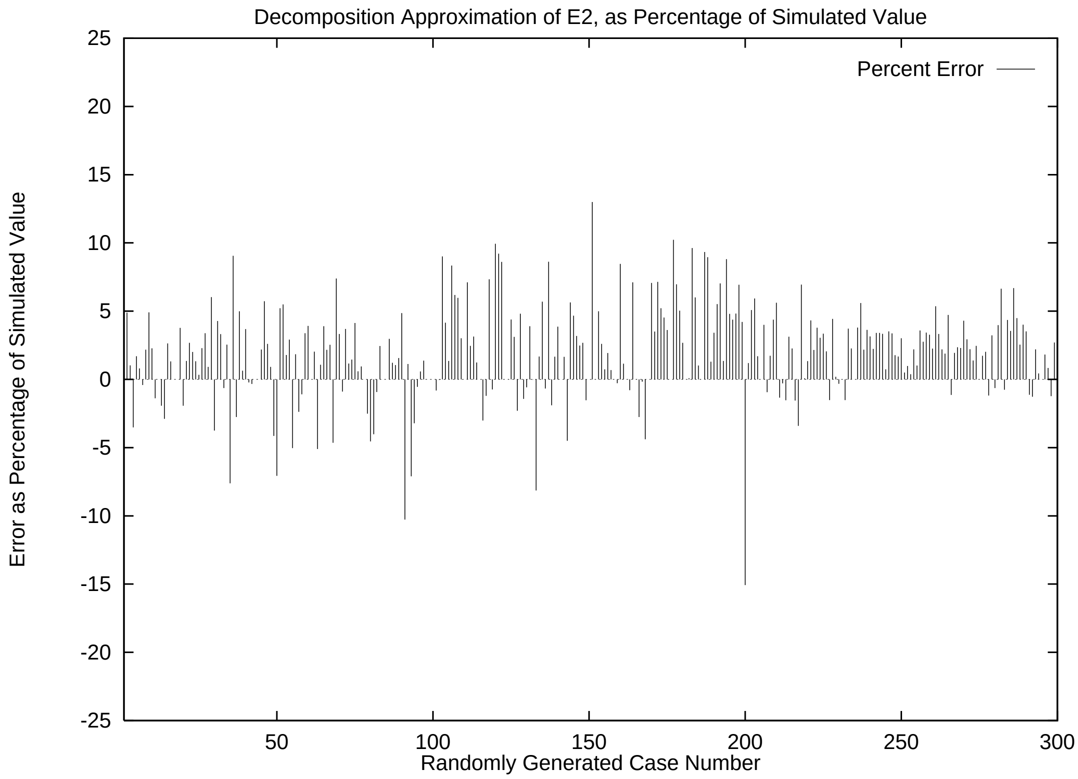

Figure 8-4: The errors in the decomposition approximation for Type 2 production rates in a production line with five processing machines.

Cases 1-100

Production Rate Percent Errors			
E_1		E_2	
0.165274		3.485540	
Part Type One		Part Type Two	
Buffer	% Error	Buffer	% Error
$B_{0,1}$	6.36	$B_{0,2}$	3.77
$B_{1,1}$	4.20	$B_{1,2}$	4.71
$B_{2,1}$	4.31	$B_{2,2}$	4.14
$B_{3,1}$	4.11	$B_{3,2}$	3.81
$B_{4,1}$	1.08	$B_{4,2}$	6.22

Cases 101-200

Production Rate Percent Errors			
E_1		E_2	
0.172897		3.819639	
Part Type One		Part Type Two	
Buffer	% Error	Buffer	% Error
$B_{0,1}$	6.38	$B_{0,2}$	3.42
$B_{1,1}$	4.33	$B_{1,2}$	5.80
$B_{2,1}$	4.02	$B_{2,2}$	6.42
$B_{3,1}$	4.05	$B_{3,2}$	7.04
$B_{4,1}$	1.02	$B_{4,2}$	11.50

Cases 201-300

Production Rate Percent Errors			
E_1		E_2	
0.042899		2.313152	
Part Type One		Part Type Two	
Buffer	% Error	Buffer	% Error
$B_{0,1}$	4.25	$B_{0,2}$	5.07
$B_{1,1}$	2.34	$B_{1,2}$	4.51
$B_{2,1}$	2.37	$B_{2,2}$	5.15
$B_{3,1}$	2.18	$B_{3,2}$	5.56
$B_{4,1}$	0.27	$B_{4,2}$	8.58

Table VII: Aggregate absolute percent errors for production rates and average inventory levels for the four-processing-machine line.

8.5.5 Production Lines with Six Processing Machines

The last set of systems we generated consisted of production lines with six internal processing machines, two machines capturing upstream supply uncertainty, and two machines capturing the behavior of downstream demand uncertainty. Following the procedure described in Section 8.4, 300 random cases were generated and simulated. The algorithm was then run, and the results were compared to simulation. For the case of a production line with two processing machines, the algorithm converged 75% of the time (226 out of 300 cases), usually taking between 20 and 500 iterations, but sometimes taking as many as 2000 iterations when it converged. On a 450Mhz Pentium II, the results still appeared almost instantaneously in most cases, but in the worst cases, it seemed to take up four seconds.

We are unable to determine why the algorithm did not converge. Investigation of the cases in which convergence failed has not revealed any insights. There are no features of the lines — such as overly-high or low isolated efficiencies, unbalances in the supply machines, or strangely distributed buffer sizes — that would lead us to be able to classify cases of non-convergence beforehand.

Production Rates

The average absolute percent error for the 226 cases where the algorithm converged was calculated to be 0.10% for Type 1 parts, and 3.90% for Type 2 parts. Broken down into the three sub-cases, the average absolute percent errors for production rates are given in Table X.

We note that the tendency of the algorithm to underestimate the Type 2 part production rate has almost completely disappeared in the line with six processing machines. Out of the 226 cases where the algorithm converged, it resulted in an underestimate in only 31 cases.

Average Buffer Levels

The estimates for average inventory levels in the six processing machine line are mostly within 5% of simulation, with again the worst estimates being for the Type 2 part finished goods inventories. There is a tendency for the percent errors to be an improvement over lines with four or five processing machines. As before, we attribute this to the fact that the algorithm did not converge 100% of the time, and the cases where the algorithm did not converge are expected to be cases where there is a lot of underlying variability, and were therefore expected to be least accurate.

Case No.	N_{11}	N_{12}	N_{21}	N_{22}	N_{31}	N_{32}	N_{41}	N_{42}	N_{51}	N_{52}
91	32	32	32	34	32	32	11	11	19	19
200	10	10	29	33	28	35	28	28	34	34

Table VIII: Buffer sizes of the systems for which the algorithm performs badly in estimating type-two production rates, for the five-processing- machine line

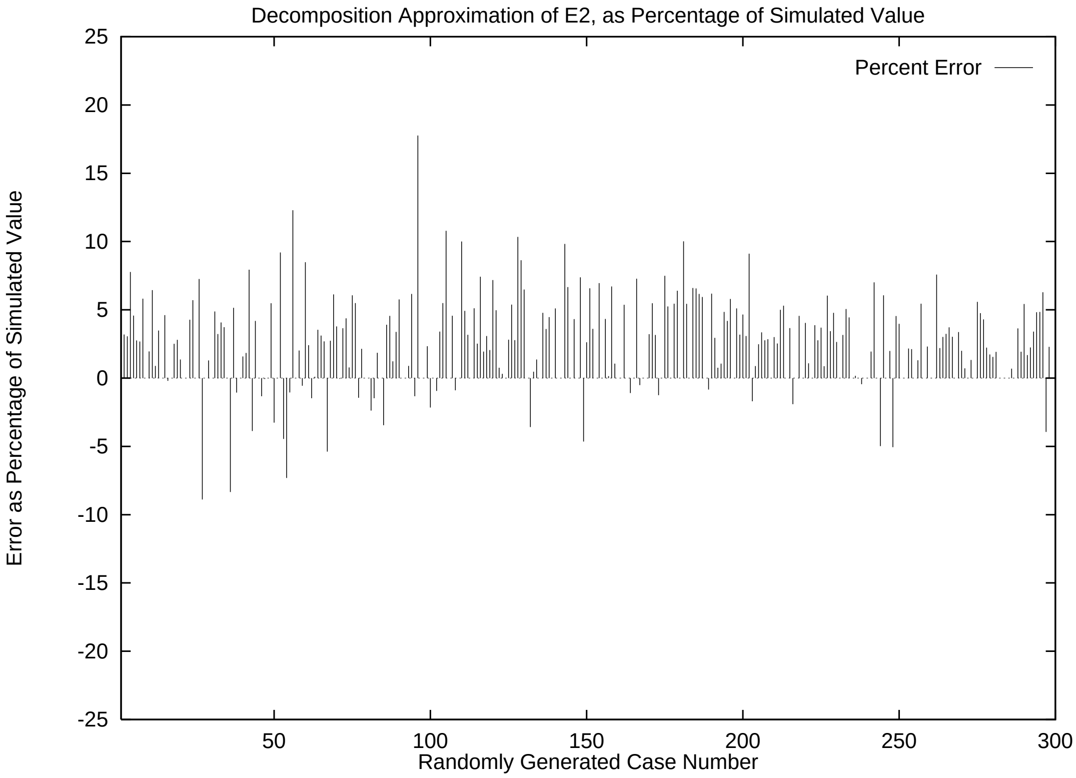

Figure 8-5: The errors in the decomposition approximation for Type 2 production rates in a production line with six processing machines.

Cases 1-100

Production Rate Percent Errors			
E_1 0.170956		E_2 2.918379	
Part Type One		Part Type Two	
Buffer	% Error	Buffer	% Error
$B_{0,1}$	6.43	$B_{0,2}$	3.54
$B_{1,1}$	4.33	$B_{1,2}$	4.39
$B_{2,1}$	4.28	$B_{2,2}$	5.15
$B_{3,1}$	4.38	$B_{3,2}$	4.42
$B_{4,1}$	4.19	$B_{4,2}$	3.28
$B_{5,1}$	1.06	$B_{5,2}$	5.96

Cases 101-200

Production Rate Percent Errors			
E_1 0.192086		E_2 4.415224	
Part Type One		Part Type Two	
Buffer	% Error	Buffer	% Error
$B_{0,1}$	6.39	$B_{0,2}$	3.33
$B_{1,1}$	4.27	$B_{1,2}$	5.93
$B_{2,1}$	4.20	$B_{2,2}$	6.27
$B_{3,1}$	4.15	$B_{3,2}$	6.08
$B_{4,1}$	3.94	$B_{4,2}$	6.37
$B_{5,1}$	1.00	$B_{5,2}$	10.88

Cases 201-300

Production Rate Percent Errors			
E_1 0.038850		E_2 2.646493	
Part Type One		Part Type Two	
Buffer	% Error	Buffer	% Error
$B_{0,1}$	4.36	$B_{0,2}$	5.06
$B_{1,1}$	2.50	$B_{1,2}$	5.16
$B_{2,1}$	2.35	$B_{2,2}$	5.58
$B_{3,1}$	2.19	$B_{3,2}$	5.44
$B_{4,1}$	2.18	$B_{4,2}$	4.69
$B_{5,1}$	0.26	$B_{5,2}$	8.55

Table IX: Aggregate absolute percent errors for production rates and average inventory levels for the five-processing-machine line.

Cases 1-100

Production Rate Percent Errors			
E_1		E_2	
0.171099		3.846623	
Part Type One		Part Type Two	
Buffer	% Error	Buffer	% Error
$B_{0,1}$	6.47	$B_{0,2}$	3.40
$B_{1,1}$	4.48	$B_{1,2}$	5.43
$B_{2,1}$	4.35	$B_{2,2}$	4.88
$B_{3,1}$	4.34	$B_{3,2}$	4.07
$B_{4,1}$	4.13	$B_{4,2}$	4.16
$B_{5,1}$	4.09	$B_{5,2}$	3.36
$B_{6,1}$	1.07	$B_{6,2}$	5.36

Cases 101-200

Production Rate Percent Errors			
E_1		E_2	
0.089991		4.546357	
Part Type One		Part Type Two	
Buffer	% Error	Buffer	% Error
$B_{0,1}$	5.76	$B_{0,2}$	3.78
$B_{1,1}$	3.25	$B_{1,2}$	5.87
$B_{2,1}$	3.42	$B_{2,2}$	6.32
$B_{3,1}$	3.39	$B_{3,2}$	6.18
$B_{4,1}$	3.36	$B_{4,2}$	7.60
$B_{5,1}$	3.13	$B_{5,2}$	8.99
$B_{6,1}$	0.53	$B_{6,2}$	10.64

Cases 201-300

Production Rate Percent Errors			
E_1		E_2	
0.042166		3.311010	
Part Type One		Part Type Two	
Buffer	% Error	Buffer	% Error
$B_{0,1}$	4.40	$B_{0,2}$	5.05
$B_{1,1}$	2.45	$B_{1,2}$	5.05
$B_{2,1}$	2.42	$B_{2,2}$	5.94
$B_{3,1}$	2.32	$B_{3,2}$	5.26
$B_{4,1}$	2.29	$B_{4,2}$	4.90
$B_{5,1}$	2.15	$B_{5,2}$	4.76
$B_{6,1}$	0.24	$B_{6,2}$	8.05

Table X: Aggregate absolute percent errors for the six-processing-machine line.

8.6 Conclusions

In testing the algorithm on randomly generated, non-homogeneous lines, we found several things. First, the algorithm converges very reliably for systems with between two and four processing machines, and less reliably for system with five or more processing machines. We noted that the estimates for Type 1 production rates and buffer levels are always within 0.5% and 5%, respectively, while those for Type 2 production rates decrease as the line gets longer. The estimates for type two inventory levels are normally within 5-8%, but worst for the finished goods inventory buffer, where the maximum error recorded was just under 12%. This we attributed to the variability in the buffer, and the low average levels seen in the buffer. We also were able to classify the cases in which the algorithm substantially underestimates the type two production rate. These were the cases in which one of the buffers was sized smaller than the others. We also noted that as the line got longer, the algorithm tended to overestimate the production rates for Type 2 parts. This suggests that improvements can be made to the decomposition to increase the accuracy.

Appendix A

Proofs

A.1 Introduction

In this Appendix, we prove Theorems 2 and 3 from Chapter 2.

A.2 Proof of Theorem 2

We consider a sequence of systems, indexed by n, that approach a limiting system as $n \to +\infty$. Let $\{\alpha_k^n\}_{n=0}^{+\infty}$ be a sequence of real numbers where $\alpha_k^n \to \alpha_k$ as $n \to +\infty$. The α_k^n is the arrival rate of class k product orders in the nth system. Define $\lambda_i^n \equiv \sum_{k=1}^{K} q_{ik}\alpha_k$. Then clearly $\lambda_i^n \to \lambda_i$ as $n \to +\infty$, where λ_i is given by (2.2). Similarly, let $\{m_i^n\}_{n=0}^{+\infty}$ be a sequence of real numbers where $m_i^n \to m_i$ as $n \to +\infty$. We then define $u_k(\ell)/\alpha_k^n$ to be the time between the $(\ell-1)$st and ℓth arrival of demand for product of class k in the nth system. Likewise, we define $m_i^n v_i(\ell)$ to be the service time of the ℓth type i item at the workstation in the nth system. Finally, we define $a_i^n(\ell)$ to be the time between the ℓth and $(\ell+1)$st demand for item i in the nth system. For each n, we make the stability assumption $m_i^n < \frac{1}{\lambda_i^n}$.

Define $D_k(t)$ to be the total number of arrivals of product orders from stream k, $(1 \le k \le K)$, in t units of busy time at the k-th virtual station. That is,

$$D_k(t) \equiv max\left\{\ell \,\middle|\, \sum_{i=1}^{\ell} \frac{u_k(i)}{\alpha_k} \le t\right\} \tag{A.1}$$

Define $T_i(t)$ to be the cumulative amount of time the workstation has devoted to producing type i components, $(1 \le i \le c)$, in $[0,t]$. Define $B_j(t)$ to be cumulative amount of time station j,$(0 \le j \le K)$ is busy in $[0,t]$. That is, $B_j(t)$ is the cumulative amount of time that none of FG inventory buffers that station j draws from are completely backlogged. By definition, $B_0(t) = \sum_{i=1}^{c} T_i(t)$. Furthermore, by writing

$$I_k(t) = t - B_k(t) \tag{A.2}$$

$I_k(t)$ is defined to be the cumulative amount of time station k,$(0 \leq k \leq K)$ is idle in $[0,t]$. That is $I_k(t)$ is the cumulative total amount of time that *at least one* of the FG inventory buffers that station j draws from are empty. Clearly, in the full backorder case this does *not* occur, and $I_k(t) = 0$, $1 \leq k \leq K$.

We let $A_i(t)$ be the cumulative total number of orders for component of type i to arrive to the workstation in $[0,t]$,$(1 \leq i \leq c)$. In the full backorder case, this is defined as follows. For demand stream k, let $\{\chi_k(j), j = 1,2,\ldots\}$ be i.i.d. random vectors of dimension $L(k)$. Let e_ℓ be the $L(k)$-dimensional unit vector of zeros with a one in the ℓ^{th} position, and zeros elsewhere. The vector e_ℓ represents the arrival of product order type ℓ from stream k. That is, $Pr\left[\chi_k(j) = e_\ell\right] = p_{k\ell}$. Let X denote the following process:

$$X_k(t) = \sum_{j=1}^{\lfloor t \rfloor} \chi_k(j), \tag{A.3}$$

for $k = 1,\ldots,K$. Denote the $c \times L(k)$ product constituency matrix for stream k by Q_k, where $[Q_k]_{ij} = q_{ij}^k$. Then we can write the component demand process A in terms of X by

$$A(t) = Q_1 X_1(D_1(t)) + \ldots + Q_K X_K(D_K(t)). \tag{A.4}$$

$$A_i(t) = \sum_{k=1}^{K} q_{ik} D_k(t) \tag{A.5}$$

In the lost sales case it is immediate that

$$A_i(t) = \sum_{k=1}^{K} q_{ik} D_k(B_k(t)) \tag{A.6}$$

Equations (A.5) and (A.6) state that the cumulative number of type i component orders to arrive to the workstation in $[0,t]$ is the sum of cumulative number of demands from each product order that requires component i to arrive in $[0,t]$.

Define $V_i(\ell)$ to be the sum of the first ℓ processing times of component type i, $(1 \leq i \leq c)$, at the workstation, and $S_i(t)$ to be the total number of type i components, $(1 \leq i \leq c)$, produced at the workstation in t units of busy time. In other words,

$$S_i(t) \equiv max\{\ell | V_i(\ell) \leq t\} \tag{A.7}$$

Define $A_i(\ell)$ to be the cumulative number of type i, $(1 \leq i \leq c)$, components to complete service at the workstation and enter FG inventory in $[0, t]$.Then

$$A_i(t) = S_i(T_i(t)). \tag{A.8}$$

states that the cumulative number of type i component orders to complete service at the workstation and enter FG inventory in $[0, t]$ is just the total number of type i components produced by the workstation in $T_i(t)$ units of busy time.

Finally, define the following corrections factors that will be needed to make our network equations exact.

$$\delta_i(t) = \begin{cases} 1 & \text{if order in service at workstation at time } t \text{ is of type } i \\ 0 & \text{otherwise} \end{cases} \tag{A.9}$$

$$\epsilon_2(t) = \begin{cases} \text{remaining service time of order in service at} & \\ \text{the workstation} & \text{if } W(t) > 0 \\ & \\ 0 & \text{otherwise} \end{cases} \tag{A.10}$$

$$\epsilon_{1i}(t) = \begin{cases} \text{amount of service current order has} & \\ \text{received at the workstation} & \text{order type } i \\ & \\ 0 & \text{otherwise} \end{cases} \tag{A.11}$$

Equation (A.9) defines $\delta_i(t)$ to be 1 if the order currently in service is of type i at time t in the n-th system, and 0 otherwise. Equation (A.10) defines $\epsilon_2(t)$ to be the remaining service time of the the order in service at the workstation, or 0 if the workstation is idle, at time t in the n-th system. Finally, Equation (A.11) defines $\epsilon_{1i}(t)$ to be the amount of service the current order in service at the workstation has received at time t in the n-th system if the order in service is type i, and 0 otherwise.

A.2.1 The Network Equations

We can now describe the physics of the network. First, let $N_i(t)$ denote the number of orders for component i, $(1 \leq i \leq c)$, in the workstation queue at time t, including the order in service. Likewise, let $Z_i(t)$ be number the of demands to FG inventory i, $(1 \leq i \leq c)$, that could be accepted (either filled immediately, or backlogged) at time t.

In the full backorder case we have

$$N_i(t) = s_i - Z_i(t). \tag{A.12}$$

Then $0 \leq N_i \leq +\infty$, and $-\infty \leq Z_i \leq s_i$.

In the lost sales the system is closed. The total number of component i orders in the system is constant, and equal to the initial FG inventory base stock level s_i. In this case we again have

$$N_i(t) = s_i - Z_i(t), \tag{A.13}$$

but both N_i and Z_i take values between 0 and s_i.

We next define the *cumulative workload* L sent to the workstation in $[0,t]$ to be

$$L(t) = \sum_{i=1}^{c} V_i \left(\sum_{k=1}^{K} q_{ik} D_k(t) \right), \tag{A.14}$$

in the full backorder case, and

$$L(t) = \sum_{i=1}^{c} V_i (\sum_{k=1}^{K} q_{ik} D_k(B_k(t))), \tag{A.15}$$

in the lost sales case. Then L is the sum of the processing times of all the component orders sent to the workstation in $[0,t]$. Using this, define

$$W(t) = W(0) + L(t) - B_0(t), \tag{A.16}$$

— where $B_0(t)$ is the cumulative amount of time the workstation has been busy in $[0,t]$ — to be the *current workstation workload*. That is, the current amount of work at the workstation at time t is the amount of work at time 0 plus the difference between the total load sent to workstation and the total work produced by the workstation in $[0,t]$. This is our main process of interest.

We now write our queue-length processes in terms of our primitive processes. From the definition of Z_i, it follows that

$$Z_i(t) = Z_i(0) + A_i(t) - \textstyle\sum_{k=1}^{K} q_{ik} D_k(B_k(t)), \text{ for } 1 \leq i \leq c. \tag{A.17}$$

That is, the level of FG inventory i at time t is the initial starting level plus the difference of the total number of type i orders to complete processing at the workstation and the total number of accepted component i demands, in $[0,t]$. Next, let

$$\tau(t) = t - W(\tau(t)) + \epsilon_2(t). \tag{A.18}$$

Then τ denotes the arrival time of the order currently in service at the workstation, if any. If there is no order in service at time t, $\tau(t)$ is defined to be t. This equation captures the FCFS processing policy. We may then write

$$N_i(t) = A_i(t) - A_i(\tau(t)) + \delta_i(t), \text{ for } 1 \leq i \leq c. \tag{A.19}$$

That is, the number of orders for component i at the workstation at time t is the difference between the cumulative total that have arrived in $[0,t]$ minus the cumulative total that arrived between 0 and the time the job currently in service (if any) arrived.

Next, we write.

$$T_i(t) = V_i(A_i(\tau(t))) + \epsilon_{1i}(t), \quad \text{for } 1 \le i \le c. \tag{A.20}$$

That is, the cumulative amount of time the workstation has devoted to producing components of type i in $[0, t]$ is defined to be the sum of the service times of all orders for component i to have arrived between 0 and the time that the order currently in service arrived, plus the amount of work the order currently in service has received if it is of type i.

A.2.2 Centering the Processes

We first prove the heavy traffic limit theorem in the partial backorder case. Consider the following centered processes.

$$\begin{aligned}
\hat{A}_i^n(t) &= A_i^n(t) - \lambda_i^n t & \text{(A.21)}\\
\hat{D}_k^n(t) &= D_k^n(t) - \alpha_k^n t & \text{(A.22)}\\
\hat{S}_i^n(t) &= S_i^n(\lfloor t \rfloor) - m_i^n \lfloor t \rfloor & \text{(A.23)}\\
\hat{\tau}^n(t) &= \tau_k^n(t) - t & \text{(A.24)}\\
\hat{F}_i^n(t) &= F_i^n(t) - \frac{1}{m_i^n} t & \text{(A.25)}
\end{aligned}$$

From (A.2), (A.6), and (A.22)

$$\begin{aligned}
A_i^n(t) &= \sum_{k=1}^{K} q_{ik} D_k^n(B_k^n(t)) & \text{(A.26)}\\
&= \sum_{k=1}^{K} \left[q_{ik} \hat{D}_k^n(B_k^n(t)) + \alpha_k^n B_k^n(t) \right] \\
&= \sum_{k=1}^{K} \left[q_{ik} \hat{D}_k^n(B_k^n(t)) + \alpha_k^n t - \alpha_k^n I_k^n(t) \right] \\
&= \sum_{k=1}^{K} q_{ik} \hat{D}_k^n(B_k^n(t)) - \sum_{k=1}^{K} \alpha_k^n I_k^n(t) + \lambda_i^n t.
\end{aligned}$$

Then from (A.15) and (A.26)

$$\begin{aligned}
L^n(t) &= \sum_{i=1}^{c} S_i^n(A_i^n(t)) & \text{(A.27)}\\
&= \sum_{i=1}^{c} \left[\hat{S}_i^n(A_i^n(t)) + m_i^n A_i^n(t) \right]
\end{aligned}$$

$$= \sum_{i=1}^{c}\left(\hat{S}_i(A_i^n(t)) + m_i^n \sum_{k=1}^{K} q_{ik}\hat{D}_k^n(B_k^n(t))\right) + \rho^n t - \sum_{k=1}^{K}\left(\sum_{i=1}^{c} q_{ik}m_i\right) I_k^n(t).$$

Set

$$\xi^n(t) = W^n(0) + \sum_{i=1}^{c}\left(\hat{V}_i(A_i^n(t)) + m_i^n \sum_{k=1}^{K} q_{ik}\hat{D}_k^n(B_k^n(t))\right) + (\rho^n - 1)t. \tag{A.28}$$

This and (A.16) give

$$W^n(t) = \xi^n(t) + I_0^n(t) - \sum_{k=1}^{K}\left(\sum_{i=1}^{c} q_{ik}m_i\right) I_k^n(t). \tag{A.29}$$

Then from (A.17) and (A.23),

$$\begin{aligned}
Z_i^n(t) &= s_i + V_i^n(t) - A_i^n(t) \qquad\qquad (A.30)\\
&= s_i + F_i^n(T_i^n(t)) - A_i^n(t)\\
&= s_i + \hat{F}_i^n(T_i^n(t)) + \frac{1}{m_i^n}T_i^n(t) - A_i^n(t)\\
&= s_i + \hat{F}_i^n(T_i^n(t)) + \frac{1}{m_i^n}\left[S_i^n(A_i^n(\tau^n(t))) + \epsilon_{1i}^n(t)\right] - A_i^n(t)\\
&= s_i + \hat{F}_i^n(T_i^n(t)) + \frac{1}{m_i^n}\left[\hat{S}_i^n(A_i^n(\tau^n(t))) + m_i^n A_i^n(\tau^n(t))\right] - A_i^n(t) + \frac{1}{m_i^n}\epsilon_{1i}^n(t)\\
&= s_i + \hat{F}_i^n(T_i^n(t)) + \frac{1}{m_i^n}\hat{S}_i^n(A_i^n(\tau^n(t))) + \left[A_i^n(\tau^n(t)) - A_i^n(t)\right] + \frac{1}{m_i^n}\epsilon_{1i}^n(t).
\end{aligned}$$

Observe that from (A.26) and (A.29) the following holds

$$\begin{aligned}
A_i^n(\tau^n(t)) &= \sum_{k=1}^{K} q_{ik}\hat{D}_k^n(B_k^n(\tau^n(t))) + \sum_{k=1}^{K}\alpha_k^n q_{ik}\tau^n(t) - \sum_{k=1}^{K}\alpha_k^n q_{ik} I_k^n(\tau^n(t))\\
&= \sum_{k=1}^{K} q_{ik}\hat{D}_k^n(B_k^n(\tau^n(t))) - \sum_{k=1}^{K}\alpha_k^n q_{ik} I_k^n(\tau^n(t))\\
&\quad + \lambda_i^n t - \lambda_i W^n(\tau^n(t)) + \lambda_i^n \epsilon_2^n(t)
\end{aligned}$$

Set

$$\xi_i^n(t) = Z_i^n(0) - \lambda_i^n \xi^n(\tau^n(t)) + \hat{S}_i^n(T_i^n(t)) + \frac{1}{m_i^n}\hat{V}_i^n(A_i^n(\tau^n(t))) \tag{A.31}$$

$$+\sum_{k=1}^{K} q_{ik}\left(\hat{D}_k^n(B_k^n(\tau^n(t))) - \hat{D}_k^n(B_k^n(t))\right) + \lambda_i^n\epsilon_2^n(t) + \frac{1}{m_i^n}\epsilon_{1i}^n(t).$$

Then from (A.30), (A.31) and (A.31)

$$\begin{aligned} Z_i^n(t) &= \xi_i^n(t) - \lambda_i^n I_0^n(t) + \lambda_i^n \sum_{k=1}^{K}\left(\sum_{j=1}^{n} q_{jk}m_j\right)\alpha_k^n I_k^n(\tau^n(t)) \\ &\quad + \sum_{k=1}^{K} q_{ik}\alpha_k^n\left(I_k^n(t) - I_k^n(\tau^n(t))\right) \end{aligned} \tag{A.32}$$

Finally, from (A.19) we have

$$\begin{aligned} N_i^n(t) &= A_i^n(t) - A_i^n(\tau^n(t)) + \delta_i^n(t) \\ &= \hat{A}_i(t) + \lambda_i^n t - \hat{A}_i^n(\tau^n(t)) - \lambda_i^n\tau^n(t) + \delta_i^n(t) \\ &= \hat{A}_i(t) - \hat{A}_i^n(\tau^n(t)) + \lambda_i^n\left(t - \tau^n(t)\right) + \delta_i^n(t) \\ &= -\lambda_i^n\hat{\tau}(t) + \hat{A}_i(t) - \hat{A}_i^n(\tau^n(t)) + \delta_i^n(t) \end{aligned} \tag{A.33}$$

Let $X(t)$ be a stochastic process. Then we define the Brownian scaling, and two fluid scalings as follows.

$$\tilde{X}^n(t) \equiv \tfrac{1}{n}X^n(n^2t), \quad \overline{X}^n(t) \equiv \tfrac{1}{n}X^n(nt), \quad \overline{\overline{X}}^n(t) \equiv \tfrac{1}{n^2}X^n(n^2t). \tag{A.34}$$

Notice that in the first equation $\tilde{X}^n$ is scaled in a way that corresponds to a central limit theorem scaling, while in the second and third it is scaled in a way that makes it amenable to a strong law of large numbers argument.

Taking the Brownian scaling of $N_i^n(\cdot)$ yields

$$\tilde{N}_i^n(t) = -\lambda_i^n\tilde{\tau}^n(t) + \tilde{A}_i^n(t) - \tilde{A}_i^n(\overline{\overline{\tau}}^n(t)) + \tilde{\delta}_i^n(t). \tag{A.35}$$

In a similar manner, taking the Brownian scaling of $W^n(\cdot)$ yields.

$$\tilde{W}^n(t) = \tilde{\xi}^n(t) + \tilde{I}_0^n - \sum_{k=1}^{K}\left(\sum_{i=1}^{c} q_{ik}m_i\right)\alpha_k^n\tilde{I}_k^n(t). \tag{A.36}$$

Taking the Brownian scaling of $\xi^n(\cdot)$ yields

$$\tilde{\xi}^n(t) = \overline{W}^n(t) + \sum_{i=1}^{c}\left(\tilde{V}_i^n(\overline{\overline{A}}_i^n(t)) + m_i^n\sum_{k=1}^{K} q_{ik}\tilde{D}_k^n(\overline{\overline{B}}_k^n(t))\right) + n(\rho^n - 1)t \tag{A.37}$$

Taking the Brownian scaling of $Z_i^n(\cdot)$ yields

$$\begin{aligned}\tilde{Z}_i^n(t) &= \tilde{\xi}^n(t) + \lambda_i^n \tilde{I}_0^n(t) + \lambda_i^n \sum_{k=1}^{K} \left(\sum_{j=1}^{n} q_{jk} m_j \right) \alpha_k^n \tilde{I}_k^n(\overline{\overline{\tau}}^n(t)) \\ &\quad + \sum_{k=1}^{K} q_{jk} \alpha_k^n \left(\tilde{I}_k^n(t) - \tilde{I}_k^n(\overline{\overline{\tau}}^n(t)) \right)\end{aligned} \tag{A.38}$$

Last, taking the Brownian scaling of $\xi_i^n(\cdot)$ yields

$$\begin{aligned}\tilde{\xi}_i^n(t) &= \overline{Z}_i^n(0) - \lambda_i^n \tilde{\xi}^n(t) + \tilde{S}_i^n(\overline{\overline{T}}_i^n(t)) + \frac{1}{m_i^n} \tilde{V}_i^n(\overline{\overline{A}}_i^n(\overline{\overline{\tau}}^n(t))) \\ &\quad + \sum_{k=1}^{K} q_{ik} \left(\tilde{D}_k^n(\overline{\overline{B}}_k^n(\overline{\overline{\tau}}^n(t))) - \tilde{D}_k^n(\overline{\overline{B}}_k^n(t)) \right) + \lambda_i^n \tilde{\epsilon}_2^n(t) + \frac{1}{m_i^n} \tilde{\epsilon}_{1i}^n(t).\end{aligned} \tag{A.39}$$

To prove the theorem, we must first prove some preliminary lemmas.

Lemma 2 *As $n \to +\infty$,*

$$(\overline{V}^n, \overline{S}^n, \overline{D}^n) \to (\overline{V}^*, \overline{S}^*, \overline{D}^*) \tag{A.40}$$

where

$$\overline{V}_i^* = m_i t, \tag{A.41}$$

$$\overline{S}_i^* = m_i^{-1} t, \tag{A.42}$$

and

$$\overline{D}_k^* = \alpha_k t. \tag{A.43}$$

Proof: This follows from the fundamental strong law of large numbers due to the independence of the variables.

Lemma 3 *As $n \to +\infty$,*

$$(\tilde{D}^n, \tilde{V}^n) \to (D^*, V^*) \tag{A.44}$$

where $D_k^* \sim BM(0, \alpha_k c_{a_k}^2)$, $1 \leq k \leq K$, and $V_i^* \sim BM(0, m_i^2 c_i^2)$, $1 \leq k \leq K$, and all component processes are independent of one another.

Proof: This follows from the functional Central Limit Theorem for renewal processes and Donsker's theorem.

Lemma 4 *As $n \to +\infty$, then the following convergences hold almost surely*

$$\begin{array}{lll} \|\overline{\epsilon}_{1i}^n(\cdot)\|_t \to 0, & \|\tilde{\epsilon}_{1i}^n(\cdot)\|_t \to 0 & a.s.\ \text{for } 1 \leq i \leq c \\ \|\overline{\epsilon}_2^n(\cdot)\|_t \to 0, & \|\tilde{\epsilon}_2^n(\cdot)\|_t \to 0 & a.s. \\ \|\overline{\delta}_i^n(\cdot)\|_t \to 0, & \|\tilde{\delta}_i^n(\cdot)\|_t \to 0 & a.s.\ \text{for } 1 \leq i \leq c \end{array}$$

Proof: The proof for $\tilde{\delta}^n$ and $\overline{\delta}^n$ is immediate. By definition of $\epsilon_{1i}(\cdot)$ from Equation (A.11), it follows that

$$0 \leq \epsilon_{1i}(t) \leq \max_{1 \leq i \leq c} \max_{1 \leq \ell \leq A_i(t)} m_i v_i(\ell)$$

That is, the service time of the item order currently in service at the workstation is bounded above by maximum service time over all classes of item orders that have arrived to the workstation in $[0, t]$. Then an application of Lemma 3.3 from Ingelhart and Whitt [34] yields

$$\|\tfrac{1}{n}\epsilon^n_{1i}(n^2\cdot)\|_t \to 0 \quad \text{a.s as } n \to +\infty.$$

Likewise, by definition of $\epsilon_2(\cdot)$ from Equation (A.10), it follows that

$$0 \leq \epsilon_2(t) \leq \max_{1 \leq i \leq c} \max_{1 \leq \ell \leq S_i(t)} m_i v_i(\ell)$$

That is, the remaining service time of the order currently in service is bounded above by the maximum service time over all item classes that have *entered service* in $[0, t]$. Again, by Lemma 3.3 of Iglehart and Whitt (1970), we have

$$\|\tfrac{1}{n}\epsilon^n_2(n^2\cdot)\|_t \to 0 \quad \text{a.s as } n \to +\infty.$$

Lemma 5 *Under the heavy traffic assumption in Equation (2.20),*

$$(\overline{T}^n_0, \overline{B}^n, \overline{A}^n_0, \overline{\tau}^n, \overline{W}^n, \overline{Q}^n_0, \overline{Q}^n) \to (\overline{T}^*_0, \overline{B}^*, \overline{A}^*_0, \overline{\tau}^*, \overline{W}^*, \overline{Q}^*_0, \overline{Q}^*) \tag{A.45}$$

uniformly on compact subsets, where

$$\overline{T}^*_i(t) = \lambda_i m_i t \quad \textit{for } 1 \leq i \leq c \tag{A.46}$$

$$\overline{B}^*_k(t) = t \quad \textit{for } 1 \leq k \leq K \tag{A.47}$$

$$\overline{A}^*_i(t) = \lambda_i t \quad \textit{for } 1 \leq i \leq c \tag{A.48}$$

$$\overline{\tau}^*(t) = t \tag{A.49}$$

$$\overline{W}^*(t) = 0 \tag{A.50}$$

$$\overline{N}^*_i(t) = 0 \quad \textit{for } 1 \leq i \leq c \tag{A.51}$$

$$\overline{Z}^*_i(t) = \beta_i \quad \textit{for } 1 \leq i \leq c \tag{A.52}$$

Proof: Applying the fluid scaling to (2.64) yields $\overline{\tau}^n(t) = t - \overline{W}^n(\overline{\tau}^n(t)) + \overline{\epsilon}^n_2(t)$. By rearranging terms and using the triangle inequality(?) we get $\mid \overline{\tau}^n(t) - t \mid \leq \mid$

$\overline{W}^n(\overline{\tau}^n(t)) \mid + \mid \overline{\epsilon}_2^n(t) \mid$. Taking the supremum over $0 \le s \le t$ of both sides and using the fact that $\overline{\tau}^n(t) \le t$ gives $\|\tau^n(\cdot) - \cdot\|_t \le \|\overline{W}^n(\cdot)\|_t + \|\overline{\epsilon}_2^n(\cdot)\|_t$. Now, from Equation (A.16) and the assumption that $W^n(0) = 0$, we have $W^n(t) = L^n(t) - B_0^n(t)$. From Equation (A.2), this can be written

$$W^n(t) = L^n(t) - t + I_0^n(t). \tag{A.53}$$

By Proposition 2.2.3 of Harrison (1985), $I_0^n(t)$ can be expressed as

$$I_0^n(t) = \sup_{0\le s\le t} \{L^n(s) - s\}^- = \sup_{0\le s\le t} \{s - L^n(s)\}, \tag{A.54}$$

where the second equality follows due to the initial condition $W^n(0) = 0$. Rewriting Equation (A.53) using Equation (A.54) gives

$$W^n(t) = L^n(t) - t + \sup_{0\le s\le t} \{s - L^n(s)\}. \tag{A.55}$$

Taking the fluid scaling of Equation (A.55) gives

$$\overline{W}^n(t) = \overline{L}^n(t) - t + \sup_{0\le s\le t} \{s - \overline{L}^n(s)\}, \tag{A.56}$$

which is

$$\overline{W}^n(t) = \sum_{i=1}^{c} \overline{V}_i(\sum_{k=1}^{K} q_{ik}\overline{D}_k^n(\overline{B}_k^n(t)) - t + \sup_{0\le s\le t} \{s - \sum_{i=1}^{c} \overline{V}_i(\sum_{k=1}^{K} q_{ik}\overline{D}_k^n(\overline{B}_k^n(t))\} \tag{A.57}$$

Now, $\overline{B}_k^n(t) - B_k^n(s) \le t - s$ and $\overline{T}_i^n(t) - T_i^n(s) \le t - s$ a.s. for all $s \le t$. Then the Arzela-Ascoli theorem gives that for any subsequence n', there exists a further subsequence n'_k on which $\overline{B}_k^{n'_k}$ and $\overline{T}_i^{n'_k}$ converge almost surely to continuous limits and the convergence is uniform on compact subsets. Since $\overline{I}_k^n(t) = t - \overline{B}_k^n(t)$, it follows that $\overline{I}_k^n \to \overline{I}_k^*$, where $\overline{I}_k^*(t) = t - \overline{B}_k^*(t)$. Moreover, $\overline{B}_k^*(t) - \overline{B}_k^*(s) \le t - s$ almost surely when $s \le t$. On the convergent subsequence n_k, it follows from Lemmas 1 and 2 and the deterministic time change theorem that $\overline{W}^{n'_k} \to \overline{W}^*$ uniformly on compact subsets, where

$$\overline{W}^*(t) = \sum_{k=1}^{K} \left(\sum_{i=1}^{c} m_i q_{ik}\alpha_k\right) \overline{B}_k^*(t) - t + \sup_{0\le s\le t} \left\{s - \sum_{k=1}^{K} \left(\sum_{i=1}^{c} m_i q_{ik}\right) \alpha_k \overline{B}_k^*(s)\right\}, \tag{A.58}$$

which can be written

$$\overline{W}^*(t) = \sup_{0\le s\le t} \left\{\sum_{k=1}^{K} \left(\sum_{i=1}^{c} m_i q_{ik}\alpha_k\right) \overline{B}_k^*(t) - t + s - \sum_{k=1}^{K} \left(\sum_{i=1}^{c} m_i q_{ik}\right) \alpha_k \overline{B}_k^*(s)\right\}. \tag{A.59}$$

By assumption, $\rho \leq 1$. It then follows immediately from Equation (2.4) that

$$\sum_{i=1}^{c} \lambda_i m_i = \sum_{i=1}^{c} \left(\sum_{k=1}^{K} q_{ik} \alpha_k \right) m_i = \sum_{k=1}^{K} \left(\sum_{i=1}^{c} q_{ik} m_i \right) \alpha_k \leq 1. \tag{A.60}$$

Let

$$\overline{B}^*(t) = \sup_{1 \leq k \leq K} \left\{ \overline{B}_k^*(t) \right\}.$$

Then $\overline{B}^*(t) - \overline{B}^*(s) \leq t - s$ a.s. for $s \leq t$. Rewrite Equation (A.59) as

$$\begin{aligned}
\overline{W}^*(t) &= \sup_{0 \leq s \leq t} \left\{ \sum_{k=1}^{K} \left(\sum_{i=1}^{c} m_i q_{ik} \right) \alpha_k \overline{B}_k^*(t) + \sum_{k=1}^{K} \left(\sum_{k=1}^{K} m_i q_{ik} \right) \alpha_k \overline{B}_k^*(s) - (t - s) \right\} \\
&\leq \sup_{0 \leq s \leq t} \left\{ \left[\sum_{k=1}^{K} \left(\sum_{i=1}^{c} m_i q_{ik} \right) \alpha_k \right] \left(\overline{B}^*(t) - \overline{B}^*(s) \right) - (t - s) \right\} \\
&\leq \sup_{0 \leq s \leq t} \left\{ \overline{B}^*(t) - \overline{B}^*(s) - (t - s) \right\}
\end{aligned}$$

where the first inequality holds because all of the $\overline{B}_k^*$s are non-decreasing and the second follows from Equation (A.60). Since $\overline{B}^*(t) - \overline{B}^*(s) \leq t - s$ almost surely for $s \leq t$, the above quantity is zero. It then follows immediately that $\overline{W}^*(t) = 0$. Then it follows from Lemma 4 that

$$\limsup_{n_k \to +\infty} \| \overline{\tau}^{n_k}(\cdot) - \cdot \|_t \leq \limsup_{n_k \to +\infty} \| \overline{W}^{n_k}(\cdot) \|_t + \limsup_{n_k \to +\infty} \| \overline{\epsilon}_2^{n_k}(\cdot) \|_t = 0. \tag{A.61}$$

Now, taking the fluid scaling of Equation (A.6) yields

$$\overline{A}_i^n(t) = \sum_{k=1}^{K} q_{ik} \overline{D}_k^n(\overline{B}_k^n(t)). \tag{A.62}$$

Taking the limit of Equation (A.62) yields

$$\overline{A}_i^*(t) = \sum_{k=1}^{K} q_{ik} \overline{D}_k^*(\overline{B}_k^*(t)). \tag{A.63}$$

Taking the fluid scaling of Equation (A.19) yields

$$\overline{N}_i^n(t) = \overline{A}_i^n(t) - \overline{A}_i^n(\overline{\tau}^n(t)) + \overline{\delta}_i^n(t). \tag{A.64}$$

Using the deterministic time change theorem, Lemma 4, the fact that $\overline{\tau}^n \to t$, it follows that $\overline{N}_i^n \to \overline{N}_i^*$, and

$$\| \overline{N}_i^n(\cdot) \|_t \leq \| \overline{A}_i^n(\cdot) - \overline{A}_i^n(\overline{\tau}^n(\cdot)) \|_t + \| \overline{\delta}_i^n(\cdot) \|_t \to \| \overline{A}_i^*(\cdot) - \overline{A}_i^*(\cdot) \|_t + 0 = 0. \quad \text{(A.65)}$$

Therefore, $\overline{N}_i^*(t) = 0$, for $1 \leq i \leq c$. Since $N_i^n(t) = s_i + b_i - Z_i^n(t)$, it follows that $\overline{N}_i^n(t) = \beta_i - \overline{Z}_i^n(t)$, from which it follows that $\overline{Z}_i^*(t) = \beta_i$, for $1 \leq i \leq c$. Next, taking the fluid scaling of Equation (A.20) yields

$$\overline{T}_i^n(t) = \overline{V}_i^n(\overline{A}_i^n(\overline{\tau}^n(t))) + \overline{\epsilon}_{1i}^n(t). \quad \text{(A.66)}$$

By the deterministic time change theorem, Lemma 4, and the fact that $\overline{\tau}^n \to t$, it follows that $\overline{T}_i^n \to \overline{T}_i^*$, for $1 \leq i \leq c$, where

$$\overline{T}_i^*(t) = m_i \sum_{k=1}^{K} q_{ik} \alpha_k \overline{B}_k^*(t). \quad \text{(A.67)}$$

Now, by assumption the idleness processes $I_k^n(t)$ increase only for those t where at least one of the FG inventories that a class k product draws from is empty, and since $\overline{Z}_i^*(t) = \beta_i$, then there exists some finite n^* and $\epsilon > 0$ such that $\overline{Z}_i^{n_k}(t) \geq \epsilon$ almost surely for all $t \geq 0$ and $n_k \geq n^*$. Thus, since $I_k^n(t)$ can increase only for those t where at least one FG inventory is empty, it follows that $\overline{I}_k^{n_k}(t) = 0$ almost surely for $t \geq 0$ and $n_k \geq n^*$. Therefore, $\overline{I}_k^*(t) = 0$ and $\overline{B}_k^*(t) = t$ for $1 \leq k \leq K$. It then follows that $\overline{T}_i^*(t) = m_i \sum_{k=1}^{K} q_{ik}\alpha_k t = m_i \lambda_i t$ and $\overline{A}_i^*(t) = \sum_{k=1}^{K} q_{ik}\alpha_k t = \lambda_i t$, for $1 \leq i \leq c$.

We can now prove Statements (2.21), (2.23), (2.10), (2.28), and (2.27), of Theorem 2. Taking the Brownian scaling of Equation (A.24) yields $-\tilde{W}^n(\overline{\overline{\tau}}^n(t)) + \tilde{\epsilon}_2^n(t)$. By the deterministic time change theorem, Lemma 4, and the Continuous Mapping Theorem $\tilde{\tau}^n(t) \Rightarrow -W^*(t)$, as $n \to +\infty$. This, Lemma 4, Lemma 5, the assumption that $\lambda_i^n \to \lambda_i$, the Deterministic Time Change Theorem and the Continuous Mapping Theorem give

$$\tilde{N}_i^n(t) = -\lambda_i^n \tilde{\tau}^n(t) + \tilde{A}_i^n(t) - \tilde{A}_i^*(\overline{\overline{\tau}}^n(t)) \Rightarrow \lambda_i W^*(t) + A_i^*(t) - A_i^*(t) = \lambda_i W^*(t),$$

for $1 \leq i \leq c$, which proves Equation (2.28).

We turn our attention to Statements (2.23) and (2.10). It follows from the heavy traffic assumption, Lemma 3, Lemma 5, the deterministic time change theorem, and the continuous mapping theorem that $\tilde{\xi}^n(t) \Rightarrow \xi^*(t)$, where

$$\xi^*(t) = \sum_{i=1}^{c} \left(V_i^*(\lambda_i t) + m_i \sum_{k=1}^{K} q_{ik} D_k^*(t) \right) + \theta t. \quad \text{(A.68)}$$

From Lemma 3 and the fundamental properties of Brownian motion we have that $V_i^*(\lambda_i t) \sim BM(0, \lambda_i m_i^2 c_i^2)$ and $q_{ik} D_k^*(t) \sim BM(0, q_{ik}^2 \alpha_k c_{a_k}^2)$. Since the processes are assumed to be mutually independent, it follows that $V_i^*(\lambda_i t) + m_i \sum_{k=1}^{K} q_{ik} D_k^*(t) \sim$

$BM\left(0, m_i^2(\lambda_i c_i^2 + \sum_{k=1}^K q_{ik}^2 \alpha_k c_{a_k}^2)\right)$. Let $X_i = V_i^*(\lambda_i t) + m_i \sum_{k=1}^K q_{ik} D_k^*(t)$. Then we have $\sum_{i=1}^c X_i \sim BM(0, \sigma^2)$. It remains to calculate the quantity σ^2. Since the X_i's need not necessarily be independent, $\sigma^2 = Var[\sum_{i=1}^c X_i] + 2\sum_{i>j} Cov[X_i, X_j]$. First, $Var[\sum_{i=1}^c X_i] = \sum_{i=1}^c m_i^2(\lambda_i c_i^2 + \sum_{k=1}^K q_{ik}^2 \alpha_k c_{a_k}^2)$. It follows from the definition of X_i and Lemma 3 that $E[X_i] = 0$ for $1 \le i \le c$. Therefore, $Cov[X_i, X_j] = E[X_i X_j]$. Substituting $V_i^*(\lambda_i t) + m_i \sum_{k=1}^K q_{ik} D_k^*(t)$ for X_i, $V_{0j}^*(\lambda_j t) + m_j \sum_{k=1}^K q_{jk} D_k^*(t)$ for X_j, expanding out the product and taking expectations yields $E[X_i X_j] = m_i m_j \sum_{k=1}^K q_{ik} q_{jk} c_{a_k}^2$. Equation (2.10) then follows. Since $\xi^*(t)$ is written as the sum of a zero-drift Brownian term and θt, it follows that $\xi^*(t) \sim BM(\theta, \sigma^2)$, and Statement (2.23) follows.

We now show that $\tilde{\xi}_i^n(\cdot) \Rightarrow \xi_i^*(\cdot)$, where

$$\xi_i^*(t) = \beta_i - \lambda_i \xi^*(t), \tag{A.69}$$

for $1 \le i \le c$. It follows from the assumption that $\lambda_i^n \to \lambda_i$, Lemma 3, Lemma 5, the deterministic time change theorem, the continuous mapping theorem and Equation (A.39) that $\tilde{\xi}_i^n(\cdot)$ converges in distribution to

$$\xi_o^*(t) = \beta_i - \lambda_i \xi^*(t) + S_i^*(\lambda_i m_i t) + \frac{1}{m_i} V_i^*(t) + \sum_{k=1}^K q_{ik}\left[D_k^*(t) - D_k^*(t)\right].$$

The last term disappears, and the second and third terms cancel because $S_i^*(\cdot)$ and $V_i^*(\cdot)$ are inverse processes of each other, and Equation (A.69) follows.

To show the convergence of the following two processes, we invoke the continuous mapping theorem, where the continuity of the processes in question is demonstrated in the Appendix. It follows that $\tilde{Z}_i^n(\cdot) \Rightarrow Z_i^*(\cdot)$, where

$$Z_i^*(t) = \xi_i^*(t) - \lambda_i I_0^*(t) + \lambda_i \sum_{k=1}^K \left(\sum_{j=1}^n q_{jk} m_j\right) \alpha_k I_k^*(t), \tag{A.70}$$

for $1 \le i \le c$, from the deterministic time change theorem, Lemma 5, and the continuous mapping theorem it follows that Equation (A.39) converges weakly to

$$Z_i^*(t) = \xi_i^*(t) - \lambda_i I_0^*(t) + \lambda_i \sum_{k=1}^K \left(\sum_{j=1}^n q_{jk} m_j\right) \alpha_k I_k^*(t) + \sum_{k=1}^K q_{ik} \alpha_k \left[I_k^*(t) - I_k^*(t)\right].$$

The last term vanishes, and Equation (2.27) follows. From the continuous mapping theorem it follows that Equation (A.36) converges weakly to

$$W^*(t) = \xi^*(t) + I_0^*(t) - \sum_{k=1}^K \left(\sum_{i=1}^c q_{ik} m_i\right) \alpha_k I_k^*(t). \tag{A.71}$$

Writing $Y_0^*(t) = \sum_{k=1}^K \left(\sum_{i=1}^c q_{ik} m_i\right) \alpha_k I_k^*(t)$ give Statements (2.21) and (2.22). Next, Statement (2.28) follows directly from Equations (2.21) and (A.69).

A.3 Proof of Theorem 3

We now prove Theorem 3 of Section 3. That is, under the scaling conventions and assumptions of Section 2.1, the scaled response time for the next type i order to arrive after time t, denoted $R_i^n(t)$, converges in distribution to the limit $R_i^*(t)$, where $R_i^*(t)$ is given in the statement of Theorem 3.

Let $D_i(t)$ denote the cumulative total number of product i orders to arrive in $[0, t]$. Let α_i denote the long-run average arrival rate of product i. Note that D_i is not, in general, a renewal process. Let $q_{ik}(r)$ denote the component k batch size of the r^{th} product i order. Set

$$\eta_k(t) \quad = \quad \sup_{u \geq 0}\{ \sum_{i \in \mathcal{P}(k)} \sum_{r=D_i(u)}^{D_i(t)} q_{ik}(r) \geq s_k \}. \tag{A.72}$$

Equation (A.73) represents the time when the replenishment orders that will be used to fill the product order arriving after time t were sent to the workstation. This batch is called the *designated batch.*

Next, set

$$\beta_k(t) \quad = \quad \sum_{i \in \mathcal{P}(k)} \sum_{r=D_i(u)}^{D_i(t)} q_{ik}(r) - s_k. \tag{A.73}$$

Equation (A.74) represents the remaining number of components to be produced from the batch that arrived at time $\eta_k(t)$.

Let B_j denote the production time at the workstation of the j^{th} component of the next order to arrive after time t. Set

$$L_k(t) = \left(W(\eta_k(t)) + \sum_{j=1}^{\beta_k(t)} B_j - (t - \eta_k(t)) \right)^+ \tag{A.74}$$

Then Equation (A.74) denotes the response time of the next component k batch to arrive after time t.

To begin our analysis of Equation (A.74), define

$$F_i(n) = \sum_{r=1}^{n} q_{ik}(r) \tag{A.75}$$

Then suppressing the subscript k to simplify the notation, and letting q_i denote the batch size of demand D_i, we have

$$\begin{aligned} \eta(t) &= \sup_{u\geq 0}\{\sum_{i\in\mathcal{P}(k)} [F_i(D_i(t)) - F_i(D_i(u))] \geq s\} \qquad \text{(A.76)} \\ &= \sup_{u\geq 0}\{\sum_{i\in\mathcal{P}(k)} \Big[\hat{F}_i(D_i(t)) - \hat{F}_i(D_i(u)) + q_i\hat{D}_i(t) - q_i\hat{D}_i(u)\Big] - \lambda t + \lambda u \geq s\}. \end{aligned}$$

Define the following scaling notion, as applied to a generic process $X(t)$:

$$\overline{X}^n(t) \equiv \frac{1}{n^2}X(n^2 t). \qquad \text{(A.77)}$$

Observe that this is similar to, but not the same as, the scaling defined in Section 2.3. There, time was scaled by a factor of n^2, but the special scaling was only on the order of n. Here, both the temporal and spatial scalings are on the order of n^2. We refer to the scaling defined in Equation (A.77) as the *fluid scaling*. Observe that in this case $\overline{X}^n$ is scaled in a manner consistent with law of large number theorems.

Scaling Equation (A.76) using the fluid scaling gives

$$\begin{aligned} \overline{\eta}^n(t) &= \frac{1}{n^2}\sup_{u\geq 0}\{\sum_{i\in\mathcal{P}(k)} \Big[\hat{F}_i(D_i(n^2t)) - \hat{F}_i(D_i(u)) + q_i\hat{D}_i(n^2t) - q_i\hat{D}_i(u)\Big] \qquad \text{(A.78)} \\ &\quad -\lambda n^2 t + \lambda u \geq s\} \\ &= \sup_{u'\geq 0}\{\sum_{i\in\mathcal{P}(k)} \Big[\hat{F}_i(D_i(n^2t)) - \hat{F}_i(D_i(n^2u')) + q_i\hat{D}_i(n^2t) - q_i\hat{D}_i(n^2u')\Big] \\ &\quad -\lambda n^2 t + \lambda n^2 u' \geq s\} \\ &= \sup_{u\geq 0}\{\sum_{i\in\mathcal{P}(k)} \Big[\overline{F}_i^n(\overline{D}_i^n(t)) - \overline{F}_i^n(\overline{D}_i^n(u)) + q_i\overline{D}_i^n(t) - q_i\overline{D}_i^n(u)\Big] \\ &\quad -\lambda t + \lambda u \geq \frac{s}{n^2}\} \\ \Longrightarrow &\quad \sup_{u\geq 0}\{\lambda(u-t)\geq 0\} = t \end{aligned}$$

where $u = n^2u'$, and the convergence follows because $\overline{D}_i^n(\cdot)\Longrightarrow\alpha_i\cdot$ as $n\to\infty$.

Observe that the following relationship holds.

$$\begin{aligned} s &\leq \sum_{i\in\mathcal{P}(k)} [F_i(D_i(t)) - F_i(D_i(\eta(t)))] \qquad \text{(A.79)} \\ &< s + \sup_{i\in\mathcal{P}(k)} \sup_{1\leq k\leq D_i(t)+1} q_k \end{aligned}$$

Scaling using the central limit scaling defined in Section 2.3 yields

$$\begin{aligned}\frac{1}{n}s &\leq \sum_{i\in\mathcal{P}(k)}\left[\overline{F}_i^n(\overline{D}_i^n(t)) - \overline{F}_i^n(\overline{D}_i^n(\eta(t))) + q_i\overline{D}_i^n(t) - q_i\overline{D}_i^n(\eta(t))\right] \\ &\quad +\lambda nt - \lambda\frac{1}{n}\eta(n^2t) \\ &< \frac{1}{n}s + \frac{1}{n}\sup_{i\in\mathcal{P}(k)}\sup_{1\leq k\leq D_i(t)+1} q_k.\end{aligned} \tag{A.80}$$

As $n \to \infty$, setting $s^* = s/n$, this converges in distribution to

$$s^* \leq \quad \tfrac{\lambda}{n}(n^2t - \eta(n^2t)) < \quad s^*, \tag{A.81}$$

which yields

$$\frac{\lambda}{n}(n^2t - \eta(n^2t)) \Longrightarrow s^*, \tag{A.82}$$

from which it follows that

$$\eta^n(t) = \frac{1}{n}(n^2t - \eta(n^2t)) \Longrightarrow \frac{s^*}{\lambda}. \tag{A.83}$$

So

$$\frac{1}{n}(n^2t - \eta(n^2t)) \approx \frac{1}{n}\frac{s^*}{\lambda} \tag{A.84}$$

Next, scaling Equation (A.74) using the central limit scaling of Section 2.3, the Random Time Change Theorem and the Continuous Mapping Theorem give

$$\begin{aligned}\frac{1}{n}L(n^2t) &= \left(\frac{1}{n}W(\eta(n^2t)) + \frac{1}{n}\sum_{j=1}^{\beta(n^2t)} B_j - \frac{1}{n}(n^2t - \eta(n^2t))\right)^+ \\ &\Longrightarrow \left(W^*(t) - \frac{s*}{\lambda}\right)^+,\end{aligned} \tag{A.85}$$

where the summation term goes to zero in distribution because

$$\begin{aligned}\frac{1}{n}\sum_{j=1}^{\beta(n^2t)} &\leq \frac{1}{n}\sup_{1\leq c\leq\beta(n^2t)} B_j \times \sup_{i\in\mathcal{P}(k)}\sup_{1\leq c\leq\beta(n^2t)} q_i(j) \\ &\Longrightarrow 0.\end{aligned} \tag{A.86}$$

Appendix B

Derivation of Downstream Equations in Chapter 7

B.1 Introduction

In this Appendix, we present the derivations for the downstream resumption of flow equations for part types one and two. The procedure is identical in form to the derivations done for the upstream equations.

B.2 Type One, Downstream

The parameter $r_d(i-1,1)$ is the repair probability of the downstream machine of the two-machine subline $L(i-1,1)$. That is, it is the probability that $M_d(i-1,1)$ makes a part in time step $t+1$, given it did not make one, and was not blocked, in time step t. There are three ways that M_i could not have produced a type one part in time step t,m given that it was not starved for type one at time $t-1$. They are:

- M_i was down at time t, and not blocked for type one at time $t-1$,
- M_i was down at time t, and blocked for type one at time $t-1$, or
- M_i was up at time t, and blocked for type one at time $t-1$

Written in terms of the long-line parameters, this is gives the expression

$$\begin{aligned} r_d(i-1,1) \quad = \quad & Pr\left[\alpha_i(t+1) = 1 \cap n_{i-1,1}(t-1) > 0 \cap n_{i,1}(t) < N_{i,1} \right| \\ & \quad (\{\alpha_i(t) = 0 \cap n_{i,1}(t-1) < N_{i,1}\} \cup \\ & \quad \{\alpha_i(t) = 0 \cap n_{i,1}(t-1) = N_{i,1}\} \cup \\ & \quad \{\alpha_i(t) = 1 \cap n_{i,1}(t-1) = N_{i,1}\}) \cap n_{i-1,1}(t-1) > 0] \end{aligned}$$

If we define the following events

$$\begin{aligned}
U &= \{\alpha_i(t+1) = 1 \cap n_{i-1,1}(t-1) > 0 \cap n_{i,1}(t) < N_{i,1}\} \\
X &= \{\alpha_i(t) = 0 \cap n_{i-1,1}(t-1) > 0 \cap n_{i,1}(t-1) < N_{i,1}\} \\
Y &= \{\alpha_i(t) = 0 \cap n_{i-1,1}(t-1) > 0 \cap n_{i,1}(t-1) = N_{i,1}\} \\
Z &= \{\alpha_i(t) = 1 \cap n_{i-1,1}(t-1) > 0 \cap n_{i,1}(t-1) = N_{i,1}\}
\end{aligned}$$

Then, as before, we can rewrite $r_d(i-1,1)$ as

$$r_d(i-1,1) = \frac{Pr\,[U|X]Pr\,[X] + Pr\,[U|Y]Pr\,[Y] + Pr\,[U|Z]Pr\,[Z]}{Pr\,[X \cup Y \cup Z]}$$

Now, proceeding in exactly the same manner as we did for the upstream parameter, we calculate

$$Pr\,[U|X] = r_i \tag{B.1}$$
$$Pr\,[U|Y] = r_i r_d(i,1) \tag{B.2}$$
$$Pr\,[U|Z] = r_d(i,1)(1 - q_u(i,1)) \tag{B.3}$$

Next, we calculate the probability of the event X in exactly the same manner as we did for the upstream machine. In order to do so, we need to classify exactly when the downstream machine of the subline is down and not starved. There are three possibilities:

- M_i is down, and not blocked or starved for type one,
- M_i is down, and blocked, but not starved for type one, or
- M_i is up, and blocked, but not starved for type one.

In other words,

$$\begin{aligned}
\{\alpha_d(i-1,1) = 0 \cap n_{i-1,1} > 0\} &\iff \\
\{\alpha_i(t) = 0 \cap n_{i-1,1}(t-1) > 0 \cap n_{i,1}(t-1) < N_{i,1}\}& \\
\cup \{\alpha_i(t) = 0 \cap n_{i-1,1}(t-1) > 0 \cap n_{i,1}(t-1) = N_{i,1}\}& \\
\cup \{\alpha_i(t) = 1 \cap n_{i-1,1}(t-1) > 0 \cap n_{i,1}(t-1) = N_{i,1}\} &\iff
\end{aligned} \tag{B.4}$$

Since these are disjoint events, it follows from Equations (7.6) and (7.7) that

$$Pr\,[X] \quad = \quad Pr\,[\alpha_d(i-1,1)=0 \cap n_{i-1,1}>0] - \mathbf{p}(i,1;N_{i,1},00) - p_b(i,1)$$

Now, from the repair frequency equals failure frequency property of the upstream machine for the two machine line (Equation (6.26)), and setting

$$I_d(i-1,1) = \frac{p_d(i-1,1)}{r_d(i-1,1)} \tag{B.5}$$

and using Equation (6.28), we have the final expression for $Pr\,[X]$.

$$\begin{aligned} Pr\,[X] \quad = \quad & I_d(i-1,1)[E(i-1,1) - q_d(i-1,1)p_s(i-1,1) + \\ & r_d(i-1,1)\mathbf{p}(i-1,1;000)] + \frac{q_d(i-1,1)}{r_d(i-1,1)}p_s(i-1,1) \\ & -\mathbf{p}(i-1,1;000) - \mathbf{p}(i,1;N_{i-1},00) - p_b(i,1) \end{aligned} \tag{B.6}$$

Next, it is straightforward to calculate $Pr\,[Y]$ and $Pr\,[Z]$ in the same way as before.

$$Pr\,[Y] \quad = \quad \mathbf{p}(i,1;N_{i,1},00) \tag{B.7}$$

$$Pr\,[Z] \quad = \quad p_b(i,1) \tag{B.8}$$

Finally, proceeding as before, we get

$$\begin{aligned} Pr\,[X] \quad = \quad & I_d(i-1,1)[E(i-1,1) - q_d(i-1,1)p_s(i-1,1) + \\ & r_d(i-1,1)\mathbf{p}(i-1,1;000)] + \frac{q_d(i-1,1)}{r_d(i-1,1)}p_s(i-1,1) \\ & -\mathbf{p}(i-1,1;000) \end{aligned} \tag{B.9}$$

The resulting equation is then

$$\begin{aligned} r_d(i,1) \quad = \quad & \Bigg(r_{i+1} \bigg[\frac{p_d(i,1)}{r_d(i,1)} \left\{ E(i+1,1) - q_d(i,1)p_s(i,1) + r_d(i,1)\mathbf{p}(i,1;000) \right\} \\ & + \frac{q_d(i,1)}{r_d(i,1)}p_s(i,1) - \mathbf{p}(i,1;N00) - \mathbf{p}(i,1;000) - p_b(i,1) \bigg] \\ & + r_{i+1}r_d(i+1,1)p_b(i+1,1) \end{aligned}$$

$$
\begin{aligned}
& \quad + r_d(i+1,1)(1-q_u(i+1,1))\mathbf{p}(i+1,1;N00)\Big) \\
\div & \left(\frac{p_d(i,1)}{r_d(i,1)}\{E(i+1,1) - q_d(i,1)p_s(i,1) + r_d(i,1)\mathbf{p}(i,1;000)\}\right. \\
& \left. + \frac{q_d(i,1)}{r_d(i,1)}p_s(i,1) - \mathbf{p}(i,1;000)\right).
\end{aligned}
$$

B.3 Type Two, Downstream

The parameter $r_d(i-1,2)$ is the repair probability of the downstream machine of the two-machine subline $L(i-1,2)$. That is, it is the probability that $M_d(i-1,2)$ makes a part in time step $t+1$, given it did not make one, and was not starved, in time step t. This is approximately the same as the probability that M_i makes a type two in step $t+1$, given it did not make type two, and was not starved for type two in step t. There are several states in which M_i does not produce a type two part in time step t, given that it was not starved for type two at time $t-1$, but can produce a type two in time step $t+1$, and can be broken down into the following three categories.

(a) M_i is up at time t, starved or blocked for type one, and blocked for type two at time $t-1$.

(b) M_i is up at time t, and producing a type one, but is about to become either blocked or starved for type one, after which it will produce a type two.

(c) M_i is down at time t, and will produce a type two when it comes up.

As with the upstream case, we define the following states that correspond to the categories above. First, we take the case where the machine is idle, and not starved for type two. This is

$$
I = \{\alpha_i(t) = 1 \cap A^c \cap n_{i-1,2}(t-1) > 0 \cap n_{i,2}(t-1) = N_{i,2}\}
$$

Next we consider the the states that represent M_i making a type one part in step t, and a type two in step $t+1$, as described in point (b), above. This can happen only if the machine is about to become blocked or starved for type one. These events are denoted as follows.

$$
\begin{aligned}
V_1 &= \{\alpha_i(t) = 1 \cap (n_{i-1,1}(t-1) = 1 \cup n_{i,1}(t-1) = N_{i,1} - 1) \\
&\qquad \cap n_{i-1,2}(t-1) > 0 \cap n_{i,2}(t-1) < N_{i,2}\} \\
V_2 &= \{\alpha_i(t) = 1 \cap (n_{i-1,1}(t-1) = 1 \cup n_{i,1}(t-1) = N_{i,1} - 1)
\end{aligned}
$$

$$\cap\ n_{i-1,2}(t-1) > 0 \cap n_{i,2}(t-1) = N_{i,2}\}$$

Last, we consider the two states that represent M_i being down in step t, and making a type two in step $t+1$, as described in point (c), above. This happens is the machine is either blocked or starved for type one, and not blocked or starved for type two, and down. Since we assume that the machine cannot be simultaneously blocked and starved for either part type, these events are represented as follows.

$$\begin{aligned} D &= \{\alpha_i(t) = 0 \cap (n_{i-1,1}(t-1) = 0 \cup n_{i,1}(t-1) = N_{i,1}) \\ &\qquad \cap\ n_{i-1,2}(t-1) > 0 \cap n_{i,2}(t-1) < N_{i,2}\} \end{aligned}$$

With these events defined, we are now ready to begin our analysis of the probability $r_d(i-1,2)$. We start by observing that the six events defined above are disjoint. Define the event, called U, that M_i makes a type two in time step $t+1$. Then

$$\begin{aligned} U &= \{\alpha_i(t) = 1 \cap (n_{i-1,1}(t-1) = 0 \cup n_{i,1}(t-1) = N_{i,1}) \\ &\qquad \cap\ n_{i-1,2}(t-1) > 0 \cap n_{i,2}(t-1) < N_{i,2}\} \end{aligned}$$

Similar to the previous case, we have

$$\begin{aligned} r_d(i-1,2) &= Pr\,[U|I \cup V_1 \cup V_2 \cup D] \\ &= \frac{Pr\,[U|I]Pr\,[I] + Pr\,[U|V_1]Pr\,[V_1] + Pr\,[U|V_2]Pr\,[V_2] + Pr\,[U|D]Pr\,[D]}{Pr\,[I \cup V_1 \cup V_2 \cup D]} \end{aligned}$$

Calculating the conditional probabilities is done exactly as before.

$$\begin{aligned} Pr\,[U|I] &= r_d(i+1,2)\left(\frac{(1-r_u(i,1))p_s(i,1) + (1-r_d(i+1,1))p_b(i,1)}{p_s(i,1)+p_b(i+1,1)}\right) \\ Pr\,[U|D] &= r_{i+1}\left(\frac{(1-r_u(i,1))\mathbf{p}(i,1;000) + (1-r_d(i+1,1))\mathbf{p}(i+1,1;N00)}{\mathbf{p}(i,1;000)+\mathbf{p}(i+1,1;N00)}\right) \\ Pr\,[U|V_1] &= \frac{(1-r_u(i,1))\mathbf{p}(i,1;101) + p_u(i,1)\mathbf{p}(i,1;111)}{\mathbf{p}(i,1;101)+\mathbf{p}(i,1;111)} \\ &+ \frac{(1-r_d(i+1,1)\mathbf{p}(i+1,1;N-1,10)}{\mathbf{p}(i+1,1;N-1,10)+\mathbf{p}(i+1,1;N-1,11)} \\ &\quad + \frac{p_d(i+1,1)\mathbf{p}(i+1,1;N-1,11)}{\mathbf{p}(i+1,1;N-1,10)+\mathbf{p}(i+1,1;N-1,11)} \\ Pr\,[U|V_2] &= r_d(i+1,2) \end{aligned}$$

$$\times \left[\frac{(1-r_u(i,1))\mathbf{p}(i,1;101)+p_u(i,1)\mathbf{p}(i,1;111)}{\mathbf{p}(i,1;101)+\mathbf{p}(i,1;111)} \right.$$
$$+\frac{(1-r_d(i+1,1))\mathbf{p}(i+1,1;N-1,10)}{\mathbf{p}(i+1,1;N-1,10)+\mathbf{p}(i+1,1;N-1,11)}$$
$$\left. +\frac{p_d(i+1,1)\mathbf{p}(i+1,1;N-1,11)}{\mathbf{p}(i+1,1;N-1,10)+\mathbf{p}(i+1,1;N-1,11)} \right]$$

Now we wish to calculate the probabilities of the unconditioned events. We begin with event D, as it is the most complicated. In order to calculate this, it will be useful to make the following observation.

$$\begin{aligned}
&\{\alpha_d(i-1,2)=0 \cap n_{i-1,2}>0\} \iff \\
&\{\alpha_i(t)=1 \cap n_{i-1,1}(t-1)>0 \cap n_{i,1}(t-1)<N_{i,1} \\
&\qquad \cap\, n_{i-1,2}(t-1)>0 \cap n_{i,2}(t-1)<N_{i,2}\} \cup \\
&\{\alpha_i(t)=1 \cap n_{i-1,1}(t-1)>0 \cap n_{i,1}(t-1)<N_{i,1} \\
&\qquad \cap\, n_{i-1,2}(t-1)>0 \cap n_{i,2}(t-1)=N_{i,2}\} \cup \\
&\{\alpha_i(t)=1 \cap (n_{i-1,1}(t-1)=0 \cup n_{i,1}(t-1)=N_{i,1}) \\
&\qquad \cap\, n_{i-1,2}(t-1)>0 \cap n_{i,2}(t-1)=N_{i,2}\} \cup \\
&\{\alpha_i(t)=0 \cap n_{i-1,1}(t-1)>0 \cap n_{i,1}(t-1)<N_{i,1} \\
&\qquad \cap\, n_{i-1,2}(t-1)>0 \cap n_{i,2}(t-1)<N_{i,2}\} \cup \\
&\{\alpha_i(t)=0 \cap (n_{i-1,1}(t-1)=0 \cup n_{i,1}(t-1)=N_{i,1}) \\
&\qquad \cap\, n_{i-1,2}(t-1)>0 \cap n_{i,2}(t-1)<N_{i,2}\} \cup \\
&\{\alpha_i(t)=0 \cap n_{i-1,1}(t-1)>0 \cap n_{i,1}(t-1)<N_{i,1} \\
&\qquad \cap\, n_{i-1,2}(t-1)>0 \cap n_{i,2}(t-1)=N_{i,2}\}
\end{aligned}$$

We can simplify this by combining the first and fourth events and the second and last events. Using the notation defined in the previous section, this becomes.

$$\begin{aligned}
&\{\alpha_d(i-1,2)>0 \cap n_{i-1,2}>0\} \iff \{A \cap B\} \\
&\cup \{A^c \cap n_{i-1,2}(t-1)>0 \cap n_{i,2}(t-1)=N_{i,2}\} \\
&\cup \{\alpha_i(t)=1 \cap A^c \cap n_{i-1,2}(t-1)>0 \cap n_{i,2}(t-1)=N_{i,2}\} \\
&\cup \{\alpha_i(t)=0 \cap A^c \cap B\}
\end{aligned}$$

Since these events are disjoint, the probability of the unions is the sum of the probabilities. It follows from the previous section that the probability of the second event is given as $\mathbf{p}(i-1,2;000)$, and the probability of the third event is just $p_s(i-1,2)$.

Furthermore, observe that the last event corresponds precisely to the event D, whose probability we are trying to determine. Therefore, the appropriate substitutions and manipulations give

$$\begin{aligned} Pr\,[D] &= Pr\,[\alpha_d(i-1,2)=0 \cap n_{i,2}>0] - p_b(i,2) - \mathbf{p}(i,2;N00) \\ &\quad -1 + p_s(i-1,1) + \mathbf{p}(i-1,1;000) + p_b(i,1) + \mathbf{p}(i,1;N00) \end{aligned}$$

Now, as was done for the part type one upstream resumption of flow equation, we can use Equation (6.25) to rewrite the first term of the above equation as

$$\begin{aligned} Pr\,[\alpha_d(i-1,2)=0 \cap n_{i,2}>0] &= I_d(i-1,2)\,(E(i-1,2) \\ &\quad -q_d(i-1,2)p_s(i-1,2) + r_d(i-1,2)\mathbf{p}(i-1,2;000)) \\ &\quad + \frac{q_d(i-1,2)}{r_d(i-1,2)}p_s(i-1,2) - \mathbf{p}(i-1,2;000), \end{aligned}$$

where

$$I_d(i-1,2) = \frac{p_d(i-1,2)}{r_d(i-1,2)} \tag{B.10}$$

Putting everything together, we finally arrive at

$$\begin{aligned} Pr\,[D] &= I_d(i-1,2)\,(E(i-1,2) - q_d(i-1,2)p_s(i-1,2) \\ &\quad +r_d(i-1,2)\mathbf{p}(i-1,2;000)) \\ &\quad +\frac{q_d(i-1,2)}{r_d(i-1,2)}p_s(i-1,2) - \mathbf{p}(i-1,2;000) - p_b(i,2) - \mathbf{p}(i,2;N00) \\ &\quad -1 + p_s(i-1,1) + \mathbf{p}(i-1,1;000) + p_b(i,1) + \mathbf{p}(i,1;N00) \end{aligned} \tag{B.11}$$

Next, we wish to calculate the probability of the event I. It follows immediately from the previous section that

$$Pr\,[I] = p_b(i,2) \tag{B.12}$$

It remains to calculate the probabilities of the states V_1 and V_2. Unfortunately, exact analysis of these is not possible, so we have to make some approximations.

We have

$$Pr\,[V_1] = \sum_{n=0}^{N-1} (\mathbf{p}(i,2;n01) + \mathbf{p}(i,2;n00)) \tag{B.13}$$

and

$$Pr\,[V_2] = \mathbf{p}(i,2;N00) \tag{B.14}$$

Bibliography

[1] F. Baskett, K. M. Chandy, R. R. Muntz, and F. G. Palacios. Open, closed, and mixed networks of queues with different classes of customers. *Journal for the Association of Computing Machinery*, 22(2):248–260, April 1975.

[2] Donald W. Berry and Bernard W. Lindgren. *Statistics, Theory and Methods*. Brooks/Cole Publishing Company, Pacific Grove, CA, 1990.

[3] Patrick Billingsley. *Convergence of Probability Measures*. John Wiley and Sons, New York, 1968.

[4] Patrick Billingsley. *Probability and Measure*. Wiley-Interscience, New York, third edition, 1995.

[5] A. A. Borovkov. Some limit theorems in the theory of mass service. ii. *Theor. Probability Appl.*, 10:375–400, 1965. (English translation).

[6] A. A. Borovkov. On limit laws for service processes in multi-channel systems. *Siberia J. Math.*, 8:746–763, 1967. (English translation).

[7] Mitchell H. Burman. *New Results in Flow Line Analysis*. PhD thesis, Massachusetts Institute of Technology, May 1995.

[8] John A. Buzacott and J. George Shanthikumar. *Stochastic Models of Manufacturing Systems*. Prentice-Hall, Englewood Cliffs, New Jersey, 1993.

[9] J. G. Dai. *Steady-State Analysis of Reflected Brownian Motions: Characterization, Numerical Methods, and Queueing Applications*. PhD thesis, Stanford University, July 1990.

[10] J.G. Dai and T.G. Kurtz. The sufficiency of the basic adjoint relationship. unpublished manuscript, 1991.

[11] Yves Dallery and Stanley B. Gershwin. Manufacturing flow line systems: A review of models and analytical results. *Queueing Systems Theory and Applications, Special Issue on Queueing Models of Manufacturing Systems*, 12(1-2):3–94, december 1997.

[12] M. Donsker. An invariance principal for certain probability limit theorems. *Mem. Amer. Math. Soc.*, 6, 1951.

[13] I. Duenyas and M. F. Keblis. Release policies for assembly systems. *IIE Transactions*, 1995.

[14] Christopher Flores. Diffusion approximations for computer communications networks. *Proceedings of Symposia in Applied Mathematics*, 31:83–124, 1985.

[15] Stanley B. Gershwin. Assembly/disassembly systems: An efficient decomposition algorithm for tree-structured networks. *IEE Transactions*, 23(4):302–314, december 1991.

[16] Stanley B. Gershwin. *Manufacturing Systems Engineering.* Prentice-Hall, Englewood Cliffs, New Jersey, 1994.

[17] Stanley B. Gershwin and O. Berman. Analysis of transfer lines consisting of two unreliable machines with random processing times and finite storage buffers. *AIEE Transactions*, 13(1):2–11, March 1981.

[18] Stanley B. Gershwin and Y.F. Choong. A decomposition method for the approximate evaluation of capacitated transfer lines with unreliable machines and random processing times. *IEE Transactions*, 19(2):150–159, 1987.

[19] Stanley B. Gershwin and I. C. Schick. Continuous model of an unreliable two-stage material flow system with a finite interstage buffer. *Massachusetts Institute of Technology Laboratory for Information and Decision Science Report LIDS-R-1039*, 1980.

[20] Stanley B. Gershwin and I. C. Schick. Modeling and analysis of three-stage transfer lines with unreliable machines and finite buffers. *Operations Research*, 31:354–380, 1983.

[21] Paul Glasserman and Yashan Wang. Leadtime-inventory trade-offs in assembly-to-order systems. *Operations Research*, 1998. to appear.

[22] Peter W. Glynn. Diffusion approximations. In *Handbooks in Operations Research and Management Science.* North-Holland, 1990.

[23] J. Michael Harrison. Assembly-like queues. *Journal of Applied Probability*, 10:354–367, 1973.

[24] J. Michael Harrison. The heavy traffic approximation for single server queues in series. *Journal of Applied Probability*, 10:613–629, 1973.

[25] J. Michael Harrison. *Brownian Motion and Stochastic Flow Systems.* Robert E. Krieger Publishing Company, Malabar, Florida, 1990.

[26] J. Michael Harrison and Viên Nguyen. Brownian models of multiclass queueing networks: Current status and open problems. *Queueing Systems*, 13:5–40, 1993.

[27] J. Michael Harrison and Martin I. Reiman. On the distribution of multidimensional brownian motion. *SIAM Journal of Applied Mathematics*, 41(2):345–361, October 1981.

[28] J. Michael Harrison and R. J. Williams. Brownian models of open queueing networks with homogeneous populations. *Stochastics*, 22:77–115, 1987.

[29] J. Michael Harrison and R. J. Williams. Brownian models of feedforward queueing networks: Quasireversibility and product form solutions. *Annals of Applied Probability*, 2(2):263–293, 1992.

[30] J. Michael Harrison, R. J. Williams, and H. Chen. Brownian models of closed queueing networks with homogeneous customer populations. *Stochastics and Stochastics Reports*, 29:37–74, 1990.

[31] J. Michael Harrrison. The diffusion approximation for tandem queues in heavy traffic. *Advances in Applied Probability*, 10:886–905, 1978.

[32] W. Hausman, H. Lee, and A. Zhang. Order fulfillment time reliability in a multi-item inventory system. 1995.

[33] Stefan Helber. Approximate analysis of unreliable transfer lines with rework or scrapping of parts. 1997.

[34] D.L. Iglehart and W. Whitt. Multiple channel queues in heavy traffic. i. *Advances in Applied Probability*, 2:150–177, 1970.

[35] D.L. Iglehart and W. Whitt. Multiple channel queues in heavy traffic. ii: Sequences, networks, and batches. *Advances in Applied Probability*, 2:355–369, 1970.

[36] Donald L. Iglehart. Weak convergence in queueing theory. *Advances in Applied Probability*, 5:570–594, 1973.

[37] Frank P. Kelly. *Reversibility and Stochastic Networks.* John Wiley, New York, NY, first edition, 1979.

[38] J. F. C. Kingman. The single server queue in heavy traffic. *Proceedings of the Cambridge Philosophical Society*, 57:902–904, 1961.

[39] J. F. C. Kingman. On queues in heavy traffic. *Journal of the Royal Statistical Society*, 24:383–392, 1962.

[40] J. F. C. Kingman. On queues in heavy traffic. In *Proc. Symposium on Congestion Theory*, pages 137–159, Chapel Hill, NC, 1965. Univ. of North Carolina Press.

[41] Leonard Kleinrock. *Queueing Systems, Volume 1: Theory.* Wiley-Interscience, New York, 1975.

[42] Averill M. Law and W. David Kelton. *Simulation Modeling and Analysis.* McGraw-Hill, New York, NY, second edition, 1991.

[43] A. J. Lemoine. Networks of queues — a survey of weak convergence results. *Management Science*, 24:1175–1193, 1978.

[44] Viên Nguyen. *Heavy traffic analysis of processing networks with parallel and sequential tasks.* PhD thesis, Department of Operations Research, Stanford University, 1990.

[45] Viên Nguyen. The trouble with diversity: Fork-join networks with heterogeneous customer population. *Annals of Applied Probability*, 4(1):1–25, 1994.

[46] Viên Nguyen. Fluid and diffusion approximations of a two-station mixed queueing network. *Mathematics of Operations Research*, 20(2), May 1995.

[47] Viên Nguyen. A multiclass hybrid production center in heavy traffic. *Operations Research*, (3):S13–S25, 1998.

[48] William P. Peterson. A heavy traffic limit theorem for networks of queues with multiple customer types. *Mathematics of Operations Research*, 16(1), February 1991.

[49] P.C. Rao and R. Suri. Approximate queueing network models for closed fabrication/assembly systems. part i: Single level systems. *Production and Operations Management*, 4, 1994.

[50] Martin I. Reiman. Open queueing networks in heavy traffic. *Mathematics of Operations Research*, 9(3):441–458, August 1984.

[51] Sheldon M. Ross. *Introduction to Probability Models.* Academic Press, San Diego, CA, fifth edition, 1993.

[52] A. V. Skorohod. Limit theorems for stochastic processes. *Theor. Probability Appl.*, 1:261–290, 1956.

[53] Jing-Sheng Song. On the order fill rate in a multi-item inventory system. *Operations Research*, 1998. to appear.

[54] Jing-Sheng Song, Susan H. Xu, and Bin Liu. Order fulfillment performance measures in an assembly-to-order system with stochastic leadtimes. *Operations Research*, 1998. to appear.

[55] Ulrich A. Tetzlaff and Harold J. Kushner. Optimal control of assemble-to-order systems under heavy traffic. Working paper, 1998.

[56] S. Varma. *Heavy and Light Traffic Approximations for Queues With Synchronization Constraints*. PhD thesis, University of Maryland, 1990.

[57] J. Walrand. A probabilistic look at networks of quasi-reversible queues. *IEEE Transactions on Information Theory*, 29:825–831, 1983.

[58] Ward Whitt. Multiple channel queues in heavy traffic. iii: Random server selection. *Advances in Applied Probability*, 2:370–375, 1970.

[59] Ruth J. Williams. Diffusion approximations for open multiclass queueing networks:sufficient conditions involving state space collapse. preprint.

[60] Ruth J. Williams. On the approximation of queueing networks in heavy traffic. to appear.

[61] Ruth J. Williams. Semimartingale reflecting brownian motions in the orthant. to appear.

[62] R. David Y. Dallery and X.-L. Xie. An efficient algorithm for analysis of transfer lines with unreliable machines and finite buffers. *IEEE Transactions on Automatic Control*, 34:943–953, 1988.

[63] Alex X. Zhang. Demand fulfillment rates in an assemble-to-order system with multiple products and dependent demands. 1995.

[64] Rachel Q. Zhang. Expected time delay in a multi-item production-inventory system. preprint.