Performance Evaluation and Optimization Models for Processing Networks with Queue-Dependent Production Quantities

by

John S. Hollywood

S.B. Applied Mathematics
Massachusetts Institute of Technology, 1996

Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Operations Research

at the

Massachusetts Institute of Technology

June 2000

Signature of Author .....
Department of Electrical Engineering and Computer Science
May 5, 2000

Certified by .....
Stephen C. Graves
Abraham J. Siegel Professor of Management and Co-Director, Leaders for Manufacturing
Thesis Supervisor

Accepted by .....
Cynthia Barnhart
Associate Professor of Civil Engineering
Co-Director, MIT Operations Research Center

Performance Evaluation and Optimization Models for Processing Networks with Queue-Dependent Production Quantities

by
John S. Hollywood

Submitted to the Department of Electrical Engineering and Computer Science
on May 5, 2000 in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Operations Research

Abstract

We consider a class of models for processing networks such as job shops or distributing data-processing systems. The defining features of this class of models are: (1) The network operates in discrete time, such that work is completed during fixed-length periods and work arrivals and transfers occur at the start of these periods. (2) Work arrivals are stochastic, characterized by a finite mean and variance. (3) Work flows are Markovian in that processing requirements do not depend on how the work got to a station. (4) Production at any station depends on work-in-queue levels through some production function. (5) We can write recursion equations (either exact or approximate) relating the moments of production and queue lengths in one period to the moments of production and queue lengths in the next period.

We call models satisfying these properties *Moment-Recursion* (MR) models. MR models support a variety of performance evaluation, optimization, and decision-support applications, such as capacity planning, resource allocation, production smoothing, and inventory control. They apply to a wide range of scenarios in which production depends on work-in-queue levels. Most directly, MR models apply to shops with control rules that minimize production fluctuation, or with machines that show saturation effects. They apply to systems in which people work at varying speeds in response to work-in-queue levels, or show the effects of overtime. MR models also address networks in which resources may be reallocated between different flows of jobs at regular intervals, such as in flexible job shops or data-processing networks.

We consider a variety of MR models, using the recursion equations to calculate the steady-state moments of the stations' productions and queue lengths for all model types. These types include:

- Models whose production functions are linear functions of the queue levels. These models may address basestock job shops, Kanban shops, and shops using other linear rules to minimize production variances.
- Models with nonlinear production functions. We develop approximations for the steady-state moments using Taylor-series expansions of the production functions.
- Models that maintain a constant weighted-inventory constraint. We use the resulting models to find weighted-inventory rules that minimize steady-state production fluctuations.
- Models that incorporate complex work transfer relationships. Here, work completed upstream is converted into jobs (or lots), and then triggers job arrivals downstream through complex probabilistic relationships.

We use the production and queue-length moments to solve a variety of steady-state optimization problems. We formulate maximum performance, maximum throughput, and minimum cost problems, and discuss nonlinear programming techniques to solve them.

Finally, we use the recursion equations to describe the transient behavior of MR-networks. We first derive the exact transient moments for models with linear control rules, both analytically and numerically. We then derive the production functions that minimize quadratic objective functions of the production and queue-length quantities, and find that the optimal functions are always linear control rules.

Thesis Supervisor: Stephen C. Graves
Title: Abraham J. Siegel Professor of Management and Co-Director, Leaders for Manufacturing

Acknowledgements

Eight years is a long time in college, especially at the Massachvsetts Institvte of Technology. My undergraduate house, Epsilon Theta, was a big part of my getting through it. Special thanks goes out to my pledge class, PC '92, along with "Uncle" Matt Condell, Alyse Leung, David Leung, and my nocturnal-but-great roommate, Jason Gratt.

I would like to thank all of my colleagues and friends who I know through student activism, as well. Not just have they done incredible work for MIT for no reason other than their desires to help students, they have provided me with great friendship and support. Luis Ortiz, Matt McGann, Jen Berk, Jen Frank, Chris Rezek, Sarah McDougal, Shawn Kelly, Will Dichtel, Liana Lareau, Chris Beland -- this is for you. Special recognition goes out to best compadres Jeremy Sher and Jake Parrott, who made being one of the "3J's" far more than part of a poker hand or group of student-life advocates. How can I ever forget the last-minute gourmet meals and dinner trips?

Also deserving of thank-yous are the student life administrators who set me on my stressful but rewarding student-activist path, notably Andy Eisenmann, Steve Immerman, "Uncle" Phil Walsh, Margaret Bates, and Mary Rowe.

This is not to say that the Operations Research Center does not deserve my gratitude as well. My colleagues have done much to reinforce the ORC's claim of being the best and friendliest place at MIT. I would especially like to thank my ORC cohort, CPT(P) Andy Armacost, Amy Cohn, Ozie Ergun, Jeremie Gallien, Martin Haugh, Brett Leida, and Marina Zaretsky. I would also like to express my appreciation for the professors who did so much to teach me OR and management, including Tom Magnanti, Jim Orlin, Dimitris Bertsimas, Larry Wein, Jack Rockart, and Starling Hunter, as well as my academic advisor, Dick Larson.

Of course, there are my undergraduate professors, too, who did much to me during my first four years at MIT. Best wishes especially to Martha Weinberg, Charles Stewart, Steve Meyer, and Jon Lendon. Most of all, I would like to express my gratitude to Al Drake, who did so much to confirm my desire to go into Operations Research, both as my freshman advisor and then throughout my undergraduate career.

I would like to thank Steve Graves, my advisor, for putting me on this interesting and important research path. More importantly, though, I would like to thank him for the great support and encouragement he has given me for the past four years. I also thank him for all the Operations Research and professional wisdom he has given me, through our meetings and seminars together.

Finally, and most importantly, I would like to thank my family for their incredible love and support. In particular, I would like to recognize my grandparents and the love they gave me – I only wish I had more time with them. Most of all, I would like to thank my parents for the love and support they have given me over the years. They have always been there for me, for which I am forever grateful. This thesis is dedicated to them.

Contents

Chapter 1: Introduction

1. The Concept of a Moment-Recursion Model

This thesis is devoted to studying the ramifications of a simple recursion equation, derived below. Consider a job shop or other network of processing stations. We model the shop in discrete time, such that stations process some amount of work in process during each time period, and transfer work to other stations (or out of the shop) at the start of the next time period. We also assume that work at a station can be modeled as a fluid (e.g. "3 hours of work at station 2") rather than as a set of distinct jobs. Then, each station i must satisfy the following elementary inventory balance equation,

$$Q_{i,t} = Q_{i,t-1} - P_{i,t-1} + A_{it}, \tag{1}$$

where Q_{it} is the queue level at the start of period t, $P_{i,t-1}$ is the amount of work processed over period $t-1$, and A_{it} is the amount of work that enters station i at the start of period t. We write the above inventory balance equations for all stations simultaneously in matrix form as:

$$\mathbf{Q}_t = \mathbf{Q}_{t-1} - \mathbf{P}_{t-1} + \mathbf{A}_t, \tag{2}$$

where $\mathbf{Q}_t$ is a vector of queue lengths, $\mathbf{P}_t$ is a vector of production quantities, and $\mathbf{A}_t$ is a vector of work arrivals. Suppose we make the following assumptions about $\mathbf{P}_{t-1}$ and $\mathbf{A}_t$.

- $\mathbf{P}_{t-1}$ will be a function of the work in process at the start of period $t-1$. Then $\mathbf{P}_{t-1} = \mathbf{p}_{t-1}(\mathbf{Q}_{t-1})$, where $\mathbf{p}(\cdot)$ is a *production function* or *control rule*. Note that $\mathbf{P}_{t-1}$ usually is a deterministic function of $\mathbf{Q}_{t-1}$, although we will consider situations where $\mathbf{P}_t$ is a probabilistic function.
- $\mathbf{A}_t$ has two components. The first consists of work sent from stations to other stations, and will be a probabilistic function of $\mathbf{P}_{t-1}$. This component incorporates the concept that completed work at one station triggers work at another station. Mathematically, this component is $\mathbf{A}_t^N[\mathbf{P}(\mathbf{Q}_{t-1})]$, where $\mathbf{A}_t^N[\cdot]$ is the *internal arrival function*.
- The second component of $\mathbf{A}_t$ consists of work that arrives to stations from outside the shop. We will also allow this component to depend on $\mathbf{Q}_{t-1}$; modeling the component this way will allow us to explore various control schemes such as constant inventory (or CONWIP) job shops. Mathematically, this component is $\mathbf{A}_t^R[\mathbf{Q}_{t-1}]$, where $\mathbf{A}_t^R[\cdot]$ is the *external arrival function*.

Now, substitute the above expressions for $\mathbf{P}_{t-1}$ and $\mathbf{A}_t$ into the inventory balance equation. This yields the recursion equation that forms the basis of the thesis.

$$\text{(MR)}\quad \mathbf{Q}_t = \mathbf{Q}_{t-1} - \mathbf{P}_{t-1}(\mathbf{Q}_{t-1}) + \mathbf{A}_t^N[\mathbf{P}(\mathbf{Q}_{t-1})] + \mathbf{A}_t^R[\mathbf{Q}_{t-1}] \tag{3}$$

At first glance, this equation does not appear useful since $\mathbf{Q}_t$ is a random vector, and the various functions in the equation are probabilistic functions. However, suppose that the following conditions are satisfied:

- We know the expectation and variance of $\mathbf{Q}_{t-1}$. (Note that $E[\mathbf{Q}_{t-1}]$ is a vector and $\text{Var}[\mathbf{Q}_{t-1}]$ is a covariance matrix).
- We can use (3) to write closed-form expressions for $E[\mathbf{Q}_t]$ and $\text{Var}[\mathbf{Q}_t]$ in terms of $E[\mathbf{Q}_{t-1}]$ and $\text{Var}[\mathbf{Q}_{t-1}]$. These expressions may be exact or approximate, depending on the production and arrival functional forms.

Call shop models that satisfy these two conditions *moment-recursion* (MR) models.

We may generate exact or approximate estimates of $E[\mathbf{Q}_t]$ and $\text{Var}[\mathbf{Q}_t]$ for moment-recursion models. This fact is extremely useful. By repeatedly iterating our closed-form expressions for $E[\mathbf{Q}_t]$ and $\text{Var}[\mathbf{Q}_t]$, we can track the distributions of the shop queues over time. Similarly, by using the inventory balance equation, we can write variants of (3) in terms of the production moments, $E[\mathbf{P}_t]$ and $\text{Var}[\mathbf{P}_t]$. Then we can track the distributions of the production quantities over time, as well.

Alternately, assume we have a job shop that has a defined steady state (all stations in the shop are stable, aperiodic, and ergodic). Then we can iterate the closed-form expressions indefinitely, converging to steady-state values for $E[\mathbf{Q}]$ and $\text{Var}[\mathbf{Q}]$. In some cases, we find closed-form expressions for $E[\mathbf{Q}]$ and $\text{Var}[\mathbf{Q}]$; in other cases, we start with reasonable estimates for $E[\mathbf{Q}]$ and $\text{Var}[\mathbf{Q}]$ and converge to steady-state values. Similarly, we find steady-state values for $E[\mathbf{P}]$ and $\text{Var}[\mathbf{P}]$. In many cases, these moments will be exact; in other cases these moments will be approximate, but quite accurate over the range of reasonably well-behaved networks. Importantly, we will find that the steady-state moments are calculated in $O(n^3)$ time, where n is the number of stations in the network.

We may use the expected queue lengths to calculate the expected waiting times, $E[\mathbf{W}]$, as well. The thesis will show that there is a fast way to calculate the expected steady-state arrival rates at all stations using (3); then, by using Little's Law, knowing $E[\mathbf{Q}]$ immediately yields $E[\mathbf{W}]$.

Knowledge of the moments of production and queue lengths are sufficient to carry out a great number of performance evaluation, optimization, and decision-support tasks. For example, we may use MR models to help optimize networks. We may maximize the performance of a network (as measured by waiting times or inventory levels), minimize the cost of the network (in terms of capacity costs), and maximize the expected throughput of the network. We may also find control policies that reduce production and queue length fluctuations (i.e., minimize the variances) – important in situations requiring output predictability such as Just-in-time manufacturing. We may use a MR-model for "what-if"

analysis, asking what will happen if a station is added, or work is re-routed, and receiving comprehensive results in terms of the production and queue-length moments (both transient and steady-state).

We expect that MR-models will be applicable to the wide range of scenarios in which work is best treated as flows of jobs, and in which production quantities are related to queue levels. Examples of such scenarios include the wide range of environments in which people naturally work faster when more work is present, and slower when less work is present. Similarly, MR models apply to shops in which people work overtime; overtime work is less efficient than regular work, resulting in functional relationships between overtime work-in-queue and overtime production.

Concerning machines, MR models apply to workstations showing saturating behavior (i.e. the machine's work-per-period is flexible, but marginal work decreases with the work-in-queue). They also model flexible job shops, in which some resources may be moved around the shop between work periods, and in which stations support sophisticated production-control policies.

Finally, MR models may be used in flexible data-processing environments, in which computing resources may be regularly reallocated between job processes. They also apply to computers that process multiple jobs simultaneously. Jobs are generally time-shared, so that an increase in jobs initially results in a direct increase in production. As more jobs are added, the computer becomes overloaded, so that the marginal increase in production declines with more jobs.

Consequently, this thesis studies a variety of situations in which we can generate moment-recursion models. It also discusses optimization and decision-support applications for these models.

2. Thesis Outline

This thesis presents steady-state analysis results for a variety of MR model classes, discusses steady-state optimization of MR models, and describes the transient analysis and control of MR models. To describe the models and applications presented in this thesis, we present a classification scheme for moment-recursion models. The scheme describes MR models as *xxx-MR*, where, *xxx* is a 3-letter prefix, and:

- The first letter describes the control rule function that determines how much of the work-in-queue to process each time period.
- The second letter describes the work transfers between stations.
- The third letter describes the work that enters from outside of the shop.

We have the following definitions for the letters in the classification scheme.

- **Lxx-MR**. The control rule is a *linear* function of the work-in-queue. Usually, we assume that the production rule is $P_i = \alpha_i Q_i$, where α_i is a fixed constant in [0,1]. This rule states that stations

process a fixed fraction of their work-in-queue each time period. We also consider the multi-queue linear control rule $P_i = \sum_j \alpha_{ij} Q_j$; this rule sets production to be a weighted sum of inventory levels throughout the network.

- **Gxx-MR**. The control rule is a *general* function of $\mathbf{Q}_t$, which we assume is continuous and twice-differentiable. We will consider single-queue control rules and multi-queue control rules.
- **xLx-MR**. The internal arrival function has the form $\mathbf{A}^N_t[\mathbf{P}_t(\mathbf{Q}_t)] = \mathbf{\Phi P}_{t-1} + \boldsymbol{\varepsilon}_t$. In other words, work arriving to a station is a *linear* combination of work at other stations plus random noise from an independent distribution.
- **xGx-MR**. The internal arrival function is a *general* probabilistic function. (We will not consider xGx-MR models explicitly in this thesis.)
- **xRx-MR**. The *request-based* external arrival function is a special type of function. It dictates that completed jobs, rather than completed work, generate work at downstream stations. In particular, the request-based function first divides completed work into some number of completed jobs. Completed jobs then become work requests at downstream stations. Each request generates a random number of jobs, and each job requires a random number of instructions to complete. The total number of instructions in queue at a station is the "work-in-queue" at the station.
- **xxS-MR**. Arrivals from outside the shop come from an independent distribution.
- **xxL-MR**. Arrivals from outside the shop depend linearly on $\mathbf{Q}_t$, e.g. $\mathbf{A}^R_t = \mathbf{BQ}_t$, where $\mathbf{B}$ is a matrix.
- **xxC-MR**. A special subclass of *xxL-MR*, the external arrival function for this class generates new work to maintain a constant inventory constraint on the job shop. (We will restrict our consideration of xxL-MR models to this subclass.)
- **xxG-MR**. Arrivals from outside the shop come from a stochastic function that depends upon $\mathbf{Q}_t$. (We will not consider xxG-MR models explicitly in this thesis.)

In Chapters 2-5, we explicitly consider the following MR models:

Chapter 2: LLS-MR Models. (*Linear control rules, linear internal arrival functions, and stationary external arrival functions*.) LLS-MR models are the simplest class of MR models. Nonetheless, these models may be applied in a wide range of situations.

We first present the simplest LLS-MR model, the Tactical Planning Model (TPM), first developed by Graves (1986). The TPM uses the simplest linear production rule, $P_i = \alpha_i Q_i$. We also present a modified TPM by Leong (1987), in which the production rule makes up a fraction of an inventory shortfall each period. Mathematically, this rule is $P_i = \alpha_i \cdot (T_i - Q_i)$, where T_i is an inventory target. This rule allows for the modeling of "pull-based" or Kanban systems.

We then expand the TPM to include models with general linear control rules (also known as *affine control rules*). These rules allow production to be a weighted sum of inventory levels at multiple stations, plus a random noise term.

We conclude with two examples showing the utility of LLS-MR models. We first study a network that uses highly sophisticated affine control rules, the *proportional restoration* rules of Denardo and Tang (1997). We then use a LLS-MR model to study a communications capacity-planning problem faced by the U.S. Department of Defense.

Chapter 3: GLS-MR Models. (*General control rules (i.e., nonlinear but twice-differential), linear internal arrival functions, and stationary external arrival functions.*) In this chapter, we show how to analyze MR-models with nonlinear control rules. By doing so, we can model a large number of nonlinear production relationships that occur in practice, such as machine-saturation and overtime effects.

GLS-MR models cannot be analyzed directly. Instead, we develop approximations for the steady-state moments of production and queue lengths by analyzing Taylor-series expansions of the general control rules. We develop two separate algorithms, both of which exactly determine the expected production quantities, find a second-order estimate of the expected queue lengths, and calculate first-order estimates of the variances of production and the queue lengths.

In addition to developing the algorithms for both single-queue and multi-queue control rules, we evaluate the algorithms' performances on a single station, a six-station chain, and a thirteen-station job shop that manufactures mainframe subcomponents, first considered by Fine and Graves (1989). We find that the estimated expected queue lengths closely match the simulated average lengths, and that the estimated variances are reasonable provided that the stations are fairly well-behaved (i.e. stations are not saturated, and the standard deviations of the work arrivals are not greater than the expected work arrivals).

Chapter 4: LLC-MR Models. (*Linear control rules, linear internal arrival functions, and external arrival functions that maintain a constant inventory in the job shop.*) With LLC-MR models, the external arrival function generates new work to maintain a constant weighted inventory in the shop. Mathematically, at the start of each period, the queue lengths satisfy the following constraint:

$$\sum_i w_i Q_{it} = W, \forall t,$$

where the w_i's are a set of weights and W is an inventory target. This extension allows the modeling of constant work-in-progress rules, which are much used strategies to control inventory and production variability.

In the chapter, we first derive equations for the steady-state production and queue length moments, and find sufficient convergence conditions for these equations. We show how to create comparable LLC-MR models from LLS-MR models. We also show how to set the vector of weights to minimize a sum of production standard deviations by solving a nonlinear program. Finally, we compare the performance of LLC-MR models to constant release models (LLS-MR models where external arrivals are kept constant). We find sufficient conditions guaranteeing that LLC-MR models see lower production variability than constant release models. We also compare the performance of LLC-MR models and constant release models on a set of ten-station job shops, and find that the LLC-MR model often generates significantly smaller production fluctuations than the constant release model.

Chapter 5: LRS-MR Models. (*Linear control rules, request-based internal arrival functions, and stationary external arrival functions.*) The other models considered in this thesis treat work as fluid flows. In particular, fluid work completed at one station is multiplied by a constant and becomes fluid work at a downstream station. In practice, however, completed jobs trigger work at downstream stations.

In this chapter, we study a special type of MR model that accounts for work transitions based on completed jobs. LRS-MR models implement flexible *request-based* relationships between the work at upstream and downstream stations. First, the work completed upstream is expressed as some number of completed jobs. These jobs become work requests at downstream stations. The downstream station randomly converts the requests into some number of jobs, then converts the jobs into a quantity of fluid work. The downstream station then processes an amount of work, separates the completed work into jobs, and continues the process by sending the completed jobs further downstream.

Request-based relationships allow the modeling of very general relationships between work completed at one station and work arriving at another station. However, it is far from obvious that analyzing request-based relationships is tractable. As we see in Chapter 5, work arrivals are not simple functions of work completed upstream. Instead, work arrivals become complicated multiple random sums of random variables. Thus, while we can write a formula for the production in period t, the formula contains sums of work requests, jobs, and instructions per job that cannot be written in terms of production in period $t - 1$. The resulting formula is not a recursion equation, and cannot be analyzed directly.

Instead, with some difficulty, we derive linear recursion equations for the expectations and variances of production. (We cannot calculate the variances exactly, but we do find close bounds on the variances.) Then, by iterating these equations, we calculate the steady-state expected production and queue lengths, and find bounds for the steady-state production and queue length variances.

In addition to presenting the derivation of the recursion equations, we apply the resulting model to an eighteen-station data processing network, similar to a United States Department of Defense network.

In addition to steady-state performance evaluation of MR-models, we consider steady-state performance optimization of these models, and the transient analysis of these models.

Chapter 6: Steady-State Optimization. We consider a variety of optimization problems for MR-models, including maximum performance (as measured by minimum waiting times or queue lengths), maximum throughput, and minimum cost problems. We show how to formulate these problems for models with linear control rules and general control rules, and discuss nonlinear programming techniques appropriate for solving the resulting problems. We also present two example optimization problems: maximizing the performance of an 18-station LDS-MR network, and minimizing the capacity cost of a 13-station GLS-MR network.

Chapter 7: Transient Analysis and Optimization. This chapter focuses on the transient analysis of MR networks, as opposed to the steady-state analysis of the previous chapters. Our approach is to use the MR equation, (3), directly and repeatedly to track the moments of production and queue lengths over time. We focus our attention on LLS-MR models, since the moments generated by repeated applications of (3) are exact for LLS-MR models.

We begin by presenting mathematical equations for the transient behavior resulting from several simple changes to expected work arrivals. We also show how to calculate the moments of the aggregate production quantities over multiple periods. Next, we use the MR equations numerically, and track the moments of production and queue lengths in a ten-station job shop facing a variety of major network changes. We conclude by deriving the optimal production policies for MR models seeking to minimize a multi-period, quadratic objective function, and find that the resulting policies are always linear functions of the queue levels.

The final chapter, Chapter 8, summarizes the contributions of the thesis and presents opportunities for future research.

3. Literature Review

We devote the remainder of this chapter to a literature review. We first consider work to date on MR-models. We next compare the features of MR models to other models of processing networks, including deterministic models and queuing-theory models.

3.1 Work to Date on MR-Models

The first paper that presented what may be described as an MR model is Cruickshanks, Drescher, and Graves (1984). In this paper, the authors studied a single production station using a bounded linear control rule, of the form $P_{it} = \{\alpha_i Q_{it}, M_i\}$, where M_i is the maximum single-period production at station i. Using a simulation study, they found that the behavior of the station approached the behavior of a station using the unconstrained linear control rule $P_{it} = \alpha_i Q_{it}$, provided that M_i is sufficiently large.

Graves (1986) developed the Tactical Planning Model, or TPM (discussed in detail in Chapter 2). As noted, the TPM calculates the steady-state moments of production and queue lengths for general configurations of stations, provided that all stations use the linear control rule $P_{it} = \alpha_i Q_{it}$, and that all fluctuations in work arrivals come from stationary distributions with a finite mean and variance. Graves also described the makeup of the work queues to analyze the waiting times at each station.

Parrish (1987) presented several extensions to the TPM. He first showed how to model work releases needed to meet a schedule of finished product demand. (The resulting model is similar in character to the constant inventory models discussed in Chapter 4.) He next introduced two service measures, the probability that demand exceeds inventory in any particular period, and the average number of successive periods in which demand is not met. He then showed how to use TPM outputs to generate these service measures, and how adjusting the TPM input parameters changed these measures. Finally, he analyzed the transient behavior of the TPM with respect to three model changes: a one-period impulse, a continuous increase in expected work arrivals, and a steady oscillation between low and high expected work arrivals. (We review Parrish's analysis of transients in Chapter 7.)

Leong (1987) adapted the TPM to model Kanban and other pull-type stations. In a pull system, stations produce to meet a downstream inventory shortfall rather than produce in response to new work at the station. Thus, production is given by the linear control rule $P_{it} = \alpha_i (T_i - Q_{it})$, where T_i is a target inventory level. (We review Leong's work in Chapter 2.)

Graves (1988a) used a single-station model, similar to a one-station TPM, to evaluate requirements for safety stocks designed to protect against stock-outs due to normal demand fluctuations. He also showed that a linear control rule was the optimal solution to quadratic cost problems involving

the station (see Chapter 7), and presented conditions for which a rule of the form $P_{it} = \alpha_i Q_{it}$ was the optimal solution.

Graves (1988b) used another single-station model, again similar to the TPM, to model a repair depot. He showed how to use the model results to determine the optimal size of the depot's work force and the optimal number of spare parts that should be kept in inventory.

Graves (1988c) presented three extensions to the single-station TPM. First, he presented steady-state moment results for a station that failed according to a Bernoulli process such that the station had a probability p of producing zero work in any particular period. Second, he presented approximate steady-state moment results for a station with lot-sizing. In this model, work completed by the station is packaged into lots of fixed size m, and these lots are routed probabilistically. (Chapter 5, which discusses request-based models, may be thought of as a major generalization of this work.) Finally, he presents mathematical bounds on the behavior of a station using a bounded control rule of the form $P_{it} = \min\{\alpha_i Q_{it}, M_i\}$.

Mihara (1988) extended the multi-station TPM to include stations that fail according to a Bernoulli process. He also performed simulation studies of a multi-station TPM in which the station used bounded control rules of the form $P_{it} = \min\{\alpha_i Q_{it}, M_i\}$. Similar to Cruickshanks, Drescher, and Graves, he found that the behavior of the bounded models approached the behavior of the unbounded TPM provided that the M_i's were sufficiently large.

Finally, Fine and Graves (1989) applied a variant of the TPM to a real-world job shop that manufactured thermal conduction modules for IBM mainframes. (In particular, they used Parrish's extensions to model requirements-driven work releases.) They found some empirical evidence for the use of linear control rules in practice.

Several authors have studied a discrete-time system similar in character to MR models. Denardo, Tang, and Lee have studied Markov production models using *proportional control rules.* These rules adjust the probabilities that a job at one station progresses to a downstream station in accordance with a linear control rule. The linear control rules are complicated pull rules: they adjust the production rate (e.g. transition probabilities) at one station to counteract excess inventory or insufficient inventory at all downstream stations. (Chapter 2 discusses an MR model that uses similar control rules.) Despite their similarity, these models are not MR-models, since they track discrete jobs moving between stations in accordance with a Markov chain, whereas MR models treat work as fluid quantities. Major papers on Markov production models with proportional control rules include Denardo and Lee (1987, 1992), and Denardo and Tang (1997).

3.2 Deterministic Models

Several major types of deterministic models are worth comparing to MR models. We consider models for aggregate and capacity planning (similar to steady-state applications of MR models) and models that track the evolution of a manufacturing system over time (similar to transient-analysis applications of MR models). Usually, these models track expected production and queue levels by assuming that these quantities are deterministic quantities.

The major difference between MR models and deterministic models is that MR models are stochastic models that track production and queue length variances. In addition to providing information about the production and queue length distributions, the variances also impact the true expected queue lengths at stations (see Chapter 3). Nonetheless, the fact that these models are deterministic makes it possible for them to model complicated production rules (including discontinuous and piecewise-differentiable control rules), as well as priority policies. These system features cannot be addressed by MR models.

Many deterministic models for aggregate planning and capacity planning model station capacity as a simple hard constraint. Production is given by the bounded control rule $P_{it} = \{Q_{it}, M_i\}$, such that a station processes up to its fixed capacity each period. While a simple and natural way to represent station capacities, this method does not account for capacity-loading effects, lead-time effects, or any other effects that cause production quantities to have direct relationships with queue levels. As noted, MR models cannot process bounded control rules, although LGS-MR models may use concave control rules that roughly approximate bounded control rules. Reviews of these models are presented in Hax (1978), Lin (1986), Baker (1993), and Bitran and Tirupati (1993).

At a lower level, the method of Input / Output control (Wight, 1970) is commonly cited. This method, effectively, is a discrete-period simulation of job arrivals and processing steps, assuming that all stations use a bounded control rule of the form $P_{it} = \{Q_{it}, M_i\}$. The drawbacks of this method are that it assumes a simple capacity bound relationship, and it is almost as complex to use as a real simulation of the job shop.

Kamarkar (1989, 1993) developed a deterministic model for a single facility in which the control rule is a concave nonlinear function designed to model saturation and congestion behavior. He suggests the control rule $P_{it} = M_i Q_{it} / (\beta_i + Q_{it})$, where M_i is an asymptotic maximum capacity and β_i is a parameter determining the rate at which production increases to M_i. He presents transient and steady-state results for the resulting model, along with a nonlinear programming model that minimizes facility costs over a set of periods.

Significant portions of this thesis develop the stochastic form of Karmarkar's model. Chapter 3 extensively discusses MR models using the control rule $P_{it} = M_i Q_{it} / (\beta_i + Q_{it})$, and shows how to develop approximations for the steady-state production and queue length moments. Then, Chapter 6 presents and solves steady-state optimization problems for models using this class of control rules. Nonetheless, it may be difficult to use the MR-equivalent of Karmarkar's model to perform transient analysis and transient optimization. As shown in Chapter 3, the recursion equations for the moments are approximations, so that iterating them repeatedly to perform transient analysis may cause the resulting estimates to be inaccurate.

For some MR models, however, the moment recursion equations are exact (i.e. linear control rules). Thus, they may be used for exact transient analysis (see Chapter 7). As such, these models may be compared to fluid-flow models and systems dynamics models. Fluid-flow models are commonly used to track quantities such as expected queue length and expected total production over a given interval. However, fluid-flow models and systems dynamics models are continuous models that generally assume deterministic behavior on the part of the queues, whereas MR models track distribution information, as well. Some work on probabilistic fluid flow models has been done (c.f. Karandikar and Kulkarni, 1995, and Asmussen, 1995), but these are one-station models assuming that the inventory in a buffer follows a Reflected Brownian Motion process. In Chapter 7, we will track the expectations and variances of the queue lengths and total production quantities over a given interval for all shop stations simultaneously.

3.3 Queueing Models

Queueing models have been widely used to model processing networks, beginning with the work of Jackson (1957, 1963). There is a large body of literature on queueing networks, much of it validating queuing models against simulations of manufacturing systems (c.f. Solberg, 1977; Buzacott and Shantikumar, 1985; Bitran and Tirupati, 1988, etc.). Reviews of queueing models of manufacturing systems include those by Buzacott and Yao (1986) and Suri and Sanders (1993).

Generally, however, queueing models assume that average production rates are constant and that service times are independent and identically distributed. They usually also assume that new arrivals are given by a Poisson process, and departures from a particular station are exponentially distributed (so that a continuous-time Markov chain theoretically represents all the states in the queue and all the transition relationships between the states). There are exceptions to these rules.

First, it is possible to model networks of queues in which the service rates depend on the state of the network, starting with extended Jackson networks (1963). The resulting equilibrium balance equations, however, may prove very difficult to solve. Alternately, one can create queuing networks similar in character to the Tactical Planning Model by having each station be an infinite-server queue; the

analysis of such networks was done by Baskett et. al (1975) and Kelly (1975, 1976). Neither of these formulations truly models queues that adjust their service rates with the total amount of work in queue, however. Further, infinite server queues do not accurately model the fact that jobs wait in queue as opposed to entering service immediately. Among other effects, the waiting-time estimates will differ between MR-models and infinite-server queueing models.

Second, there are models of queueing networks in which the stations need not have exponentially-distributed service times. BCMP networks (Baskett et. al, (1975)) and Kelly networks (Kelly, 1975 and 1976), both allow the distributions of Cox (1955) to generate customer service times, provided that the all queues use the processor-sharing, last-come first-served preemptive resume, or infinite-server disciplines. The Cox distributions include all those distributions with a rational Laplace transform; the importance of these distributions is that they can be constructed by a sequence of exponential stages.

Other models provide approximate results for networks of GI/G/m queues. Whitt (1983a and 1983b) developed a two-moment approximation model, the Queueing Network Analyzer. This model was extended by Bitran and Tirupati (1988). Much recent literature has focused on the development of heavy-traffic models. These models use results that queueing networks may be approximated by Brownian motion models such that departures become exponentially distributed as they become heavily loaded (queue busy at least 90% of the time, usually). Work in this area includes that of Harrison and Williams (1987), Harrison (1988) and Wein (1992). The drawback of both of these types of methods is that they are approximations of varying accuracy; notably, heavy-traffic approximations only work well if the queues are heavily loaded.

An advantage of MR models is that they allow the modeling of splits in workflows (in which one job at a station becomes multiple jobs downstream); this generally is not allowed in queueing models. On the other hand, it is somewhat difficult to model probabilistic job routing in an MR-model, whereas probabilistic job routing is a fundamental feature of queueing models. Nonetheless, Graves (1988c) presents an approximate method for modeling probabilistic job routing in the TPM, and Chapter 5 discusses the approximate modeling of very general relationships between jobs completed at one station and jobs arriving at another station.

In general, we see that queueing models will be preferred when the network has stations with independent service times for individual jobs (especially times close to exponential), operates continuously, and has jobs consisting of distinct classes of "customers" moving randomly around the network. MR models will be preferred when the network has queue-dependent service times as a function of total work, operates either in discrete periods or has discrete control-review periods, and is well defined by work flows and / or streams of jobs rather than by individual customers.

Chapter 2: The Tactical Planning Model and Other Models with Linear Control Rules (Class LLS-MR)

1. Introduction

In this section, we consider moment-recursion models with linear control rules and independent and identically distributed work arrivals. These models are the simplest MR models, but show the basic analytic techniques expanded upon in later chapters. They also apply to a wide range of systems, and in certain cases, may help produce control policies that will be provably optimal with respect to minimizing production variances.

We begin with the simplest, and first, MR-model, the Tactical Planning Model (TPM), developed by Graves (1986). The TPM is a "push" model in which stations deterministically process a fixed fraction of the work in their input buffers each time period. We next review an adaptation of the TPM that models "Pull-based" or Kanban systems, developed by Leong (1987) in which stations process an amount needed to make up a fraction of the shortfall in their output buffers each time period.

We then develop a General Linear Control Rule model, which adds several major extensions to the TPM. First, production quantities can depend on the queue lengths of multiple stations. Second, part of a station's production quantities can be constants. Last, we allow fluctuations in the work actually processed by a station. Together, these extensions allow us to model general linear (or *affine*) control rules, which will allow us to model shops using very sophisticated control policies.

We conclude the section with two examples showing the utility of linear control rule MR models. We first present a model of a shop that uses sophisticated linear control rules, the *proportional restoration* rules of Denardo and Tang (1997). We then conclude the chapter with a digital-communications capacity planning application, which applies MR models to a scenario that one would not normally associate with manufacturing models.

2. The Tactical Planning Model

The Tactical Planning Model (TPM), developed by Graves (1986), was the first moment-recursion model, and is the simplest model in the *LLS-MR* class. It is a discrete-time, continuous-flow model that tracks work flows rather than jobs through a job shop. We assume an underlying time period for the model and express the arrival of work per period in terms of time units (i.e. hours) of work rather than individual jobs. We model production per period at a workstation as the amount of work performed rather than as the number of jobs completed. Individual jobs have no identity in the model. (Note that most MR models assume that work is modeled this way; the *xGx-MR* models form the exception.)

2.1 Model Development

Each station uses a simple linear control rule to determine the amount of work to perform each period:

$$P_{it} = a_i Q_{it}, \tag{1}$$

where P_{it} is the production of work station i in time period t, Q_{it} is the work-in-process or work-in-queue at the start of period t, and the parameter $a_i, 0 < a_i \leq 1$, is a smoothing parameter. In words, production at workstation i is a fixed portion (a_i) of the queue of work remaining at the start of the period. The inverse ($1/a_i$) corresponds to the planned lead-time at workstation i.

To use the control rule, we specify the queue level Q_{it} by the inventory balance equation:

$$Q_{it} = Q_{i,t-1} - P_{i,t-1} + A_{it}, \tag{2}$$

where A_{it} is the amount of work that arrives at workstation i at the start of period t. By using the control rule (1) to replace Q_{it} in the balance equation (2), we get a simple smoothing equation:

$$P_{it} = (1 - a_i)P_{i,t-1} + a_i A_{it}. \tag{3}$$

We next characterize the work arrivals. A workstation receives two types of arrivals. The first type comprises new jobs that have their first processing step at the station. The second type comprises jobs in process that have just completed processing at an upstream station. We model the arrivals to a station from another station by the following equation:

$$A_{ijt} = \phi_{ij} P_{j,t-1} + \varepsilon_{ijt}. \tag{4}$$

In this equation, A_{ijt} is the amount of work arriving to station i from station j at the start of period t, ϕ_{ij} is a positive scalar, and ε_{ijt} is random variable. Thus, we assume that one unit (e.g. hour) of work at station j generates ϕ_{ij} time units of work at station i, on average. The term ε_{ijt} is a noise term that introduces uncertainty into the relationship between production at j and arrivals to i; we assume this term is a serially i. i. d. random variable with zero mean and a known variance.

Then, the arrival stream to station i is:

$$A_{it} = \sum_j A_{ijt} + N_{it}, \tag{5}$$

where N_{it} is an i. i. d. random variable for the work load from new jobs that enter the shop at station i and at time t. Substituting for A_{ijt}:

$$A_{it} = \sum_j \phi_{ij} P_{j,t-1} + \varepsilon_{it}, \text{ where } \varepsilon_{it} = N_{it} + \sum_j \varepsilon_{ijt}. \tag{6}$$

Note that ε_{it} represents the work arrivals not predictable from the production levels of the previous period, and includes work from new jobs and from noise in existing workflows. By assumption, the time series ε_{it} is independent and identically distributed over time.

We next rewrite the equations for production (3) and work arrivals (6) in matrix-vector form:

$$\mathbf{P}_t = (\mathbf{I} - \mathbf{D})\mathbf{P}_{t-1} + \mathbf{D}\mathbf{A}_t, \tag{7}$$

$$\mathbf{A}_t = \Phi_{t-1}\,\mathbf{P}_{t-1} + \varepsilon_t. \tag{8}$$

Here, $\mathbf{P}_t = \{P_{1t}, \ldots, P_{nt}\}'$, $\mathbf{A}_t = \{A_{1t}, \ldots, A_{nt}\}'$, and $\varepsilon_t = \{\varepsilon_{1t}, \ldots, \varepsilon_{nt}\}'$ are column vectors of random variables, n is the number of workstations, $\mathbf{I}$ is the identity matrix, $\mathbf{D}$ is a diagonal matrix with $\{a_1, \ldots, a_n\}$ on the diagonal, and Φ is an n-by-n matrix with elements ϕ_{ij}. By substituting equation (8) into equation (7), we find:

$$\mathbf{P}_t = (\mathbf{I} - \mathbf{D} + \mathbf{D}\Phi)\mathbf{P}_{t-1} + \mathbf{D}\varepsilon_t. \tag{9}$$

By iterating this equation and assuming an infinite history of the system, we rewrite the above equation as the following infinite series:

$$\mathbf{P}_t = \sum_{s=0}^{\infty} (\mathbf{I} - \mathbf{D} + \mathbf{D}\Phi)^s \mathbf{D}\varepsilon_{t-s}. \tag{10}$$

To calculate the moments of the production random vector $\mathrm{P_t}$, we denote the mean and the covariance for the noise vector ε_t by $\mu = \{\mu_1, \ldots, \mu_n\}'$, and by $\Sigma = \{\sigma_{ij}^2\}$, respectively. Then, the expectation of the production vector is given by:

$$\begin{aligned} E[\mathbf{P}_t] &= \sum_{s=0}^{\infty} (\mathbf{I} - \mathbf{D} + \mathbf{D}\Phi)^s \mathbf{D}\mu, \\ &= (I - \Phi)^{-1}\mu, \end{aligned} \tag{11}$$

provided that the spectral radius of Φ is less than one (see Graves 1986). The covariance matrix of production, $\mathbf{S}$, is given by:

$$\begin{aligned} \mathbf{S} = \mathrm{var}(\mathbf{P}_t) &= \sum_{s=0}^{\infty} \mathbf{B}^s \mathbf{D}\Sigma\mathbf{D}\mathbf{B}'^s, \\ \text{where } \mathbf{B} &= \mathbf{I} - \mathbf{D} + \mathbf{D}\Phi. \end{aligned} \tag{12}$$

Again, this infinite series converges if the spectral radius of Φ is less than one. Note that from **S** we have found the production variance for each station, as well as the covariance for each pair of workstations. The relationship $\mathbf{P}_t = \mathbf{D}\mathbf{Q}_t$ immediately implies the moments of the queue lengths:

$$\begin{aligned} \mathbf{E}[\mathbf{Q}_t] &= \mathbf{D}^{-1}\mathbf{P}_t, \text{ and} \\ \text{var}[\mathbf{Q}_t] &= \mathbf{D}^{-1}\mathbf{S}\mathbf{D}^{-1}. \end{aligned} \tag{13}$$

2.2 Model Waiting-Time Statistics

Throughout this thesis, we primarily will calculate the moments of production and queue lengths. However, it is possible to use the TPM to develop some other useful statistics, as well. Graves (1986) showed how to characterize the queue *backlogs*, the amount of work that waits for a particular amount of time at each workstation. Under certain situations, it is possible to use these results to characterize the distribution of the waiting times at each station.

2.2.1 Characterization of Queue Backlogs

We begin by reviewing the characterization of the amount of work that waits for a particular amount of time. Assume that work at each station is always processed in first-in, first-out order (FIFO). Then, define Q_{it}^m to be:

$$\begin{aligned} Q_{it}^m &= Q_{i,t-1}^{m-1} - P_{i,t-1}, \\ &= Q_{i,t-m}^0 - \sum_{s=1}^{m} P_{i,t-s}, \end{aligned} \tag{14}$$

where $Q_{it}^0 = Q_{it}$. In words, if Q_{it}^m is positive, it represents the amount of queued work at station i that has been in queue for at least m periods prior to the start of the current period. If Q_{it}^m is negative, it indicates that none of the current work has been in queue for m periods. Instead, the station has processed $-Q_{it}^m$ worth of more recent arrivals. In matrix notation, we rewrite all Q_{it}^m's simultaneously as:

$$\mathbf{Q}_t^m = \mathbf{Q}_{t-m}^0 - \sum_{s=1}^{m} \mathbf{P}_{t-s}. \tag{15}$$

Using the fact that $\mathbf{Q}_{t-m}^0 = \mathbf{D}^{-1}\mathbf{P}_{t-m}$ yields:

$$\mathbf{Q}_t^m = \mathbf{D}^{-1}\mathbf{P}_{t-m}^0 - \sum_{s=1}^{m} \mathbf{P}_{t-s}, \tag{16}$$

so that the queue level is written entirely in terms of the production vectors. From the development in section 2.1, it is clear that $\mathbf{Q}_t^{\mathbf{m}}$ can be rewritten in terms of the noise vectors, the $\boldsymbol{\varepsilon}_t$'s, as well. From the latter representation, we find that:

$$E(\mathbf{Q}_t^{\mathbf{m}}) = \left(\mathbf{D}^{-1} - m\mathbf{I}\right)(\mathbf{I} - \boldsymbol{\Phi})^{-1}\boldsymbol{\mu}, \text{ and} \tag{17}$$

$$\begin{aligned} \text{var}(\mathbf{Q}_t^{\mathbf{m}}) = &\sum_{j=1}^{m-1}\left(\mathbf{I} + \ldots \mathbf{B}^{j-1}\right)\mathbf{D}\boldsymbol{\Sigma}\mathbf{D}^{-1}\left(\mathbf{I} + \ldots \mathbf{B}^{j-1}\right)' \\ &+ (\mathbf{D}^{-1} - \mathbf{I} - \mathbf{B} - \ldots - \mathbf{B}^{m-1})\mathbf{S}(\mathbf{D}^{-1} - \mathbf{I} - \mathbf{B} - \ldots - \mathbf{B}^{m-1})', \end{aligned} \tag{18}$$

with **S**, **B**, and Σ defined in section 2.1.

2.2.2 Characterization of the Waiting Times Under Normality Assumptions

The probability that the waiting time at station i is between m and $m-1$ periods, $p_{(m \geq w_i \geq m-1)}$, equals the probability that $Q_{it}^{m-1} \geq 0$ and $Q_{it}^{m} \leq 0$. The first condition states that a positive amount of work has been in queue at least *m-1* periods, while the second implies that no work has been in queue for more than m periods. Mathematically, we have:

$$p_{(m \geq w_i \geq m-1)} = \begin{cases} 1 - p(Q_{it}^{1} \geq 0), m = 1 \\ p(Q_{it}^{m-1} \geq 0) - p(Q_{it}^{m} \geq 0), m > 1. \end{cases} \tag{19}$$

In words, $p_{(m \geq w_i \geq m-1)}$ is the probability that work has been queue at least $m-1$ periods less the probability that work has been in queue for m periods. The special case for $m = 1$ is needed since all work waits during its processing time. If we can calculate the probabilities that $Q_{it}^{m} \geq 0$, we can calculate the distributions of the waiting times. For example, suppose that all the work arrivals are normally distributed. Then, the production quantities will be normally distributed, since the production quantities are sums of the work arrivals. Thus, equation (16) implies that the Q_{it}^{m}'s are normally distributed, as well. Define $Z(x,\sigma)$ to be a normal distribution with mean x and standard deviation σ. Then, we can rewrite (19) as:

$$\begin{aligned} p_{(m \geq w_i \geq m-1)} &= \left[1 - p\left(Z\left(E(Q_{it}^{m-1}), \sigma(Q_{it}^{m-1})\right) \leq 0\right)\right] - \left[1 - p\left(Z\left(E(Q_{it}^{m}), \sigma(Q_{it}^{m})\right) \leq 0\right)\right] \\ &= p\left(Z\left(E(Q_{it}^{m}), \sigma(Q_{it}^{m})\right) \leq 0\right) - p\left(Z\left(E(Q_{it}^{m-1}), \sigma(Q_{it}^{m-1})\right) \leq 0\right), m \geq 2, \\ &\text{and} \\ p_{(1 \geq w_i \geq 0)} &= p\left(Z\left(E(Q_{it}^{1}), \sigma(Q_{it}^{1})\right) \leq 0\right), m = 1. \end{aligned} \tag{20}$$

Again, the special case for $m = 1$ is needed since all work waits in the queue for at least part of one period. Using (20) for all values of m yields the probability mass function for the waiting times at a particular station. To ensure that (20) creates a valid probability distribution, we calculate the corresponding cumulative distribution function:

$$\begin{aligned} p_{(w_i \le M)} &= p\left(Z\left(E(Q_{it}^1), \sigma(Q_{it}^1)\right) \le 0\right) + \sum_{m=2}^{M} \left[p\left(Z\left(E(Q_{it}^m), \sigma(Q_{it}^m)\right) \le 0\right) - p\left(Z\left(E(Q_{it}^{m-1}), \sigma(Q_{it}^{m-1})\right) \le 0\right)\right] \\ &= p\left(Z\left(E(Q_{it}^m), \sigma(Q_{it}^m)\right) \le 0\right) \end{aligned} \quad (21)$$

so that $\lim_{M \to \infty} p_{(w_i \le M)} = 1$, since $\lim_{m \to \infty} E(Q_{it}^m) = -\infty$.

Example. Consider a single work station that receives an average of 5 units of work per period with a standard deviation of 3 units of work (variance of 9 units). Assume that the station is forced to redo a random amount of work each period. The expected random work to be redone is 30% of the production quantity, with a variance of 1 unit. In terms of the TPM input variables, this implies that $\mu_1 = 5$, $\phi_{11} = 0.3$, and $\Sigma_{11} = 9 + 1 = 10$. The smoothing parameter is set to be $\alpha_1 = 1/5$ (so the lead time is 5 periods). Applying the steady-state TPM equations to the station, the station's moments of production and queue lengths are:

E(P_1)	var(P_1)	E(Q_1)	var(Q_1)
7.14	1.24	35.71	6.20

The moments of the queue backlogs are (through $m = 10$ periods):

m	$E(Q_{it}^m)$	var(Q_{it}^m)	m	$E(Q_{it}^m)$	var(Q_{it}^m)
1	28.57	24.58	6	-7.14	15.26
2	21.43	15.55	7	-14.29	21.81
3	14.29	10.63	8	-21.43	30.31
4	7.14	9.27	9	-28.57	40.48
5	0	10.95	10	-35.71	52.08

Using the above formulas, and assuming all arrivals are normally distributed, yields the following distribution for the station's waiting times. Note that the distribution is given in terms of discrete intervals; for example, the probability of "4-5" is the probability that work will wait in queue between 4 to 5 periods.

Waiting Time (periods)	Probability	Waiting Time (periods)	Probability
0-1	.0000	5-6	.4663
1-2	.0000	6-7	.0326
2-3	.0000	7-8	.0011
3-4	.0095	8-9	.0000
4-5	.4905	9-10	.0000

In this example, about 96% of all the work in queue waits between 4 and 6 periods. One of the consequences of using a linear control rule is that waiting times become quite predictable, even with a high variance for work arrivals.

2.2.3 Characterization of the Waiting Times Without Normality Assumptions

If the work arrivals are not normally distributed, we have fewer options. We can calculate the expected waiting time explicitly, using Little's Law. Recall the law states that $\overline{Q_i} = \lambda_i w_i$, where $\overline{Q_i}$ is the expected queue length, λ_i is the arrival rate, and w_i is the expected waiting time. The TPM is a conservative model in which work arrivals equal work departures at each station. Thus, the total arrival rate to a station simply equals the station's expected production, E(P_i). $L = E(Q_i)$ by definition; from (12), we also have that $L = \alpha_i E(P_i)$. Substituting into Little's Law, and simplifying, we find that $w_i = 1/\alpha_i$. Thus, setting the smoothing factor implicitly determines the expected waiting times, or lead times. This result holds regardless of the arrival distributions or the order in which the station processes the jobs. Importantly, the fact that $W = 1/\alpha_i$ sets up tradeoffs between queue waiting times and production and queue length variances; these tradeoffs form the basis of optimization problems discussed in Chapter 6.

Unfortunately, however, the variance of the waiting time generally cannot be calculated explicitly from the moments of production and queue lengths. While there are generalized versions of Little's Law, they require that arrivals be exponentially distributed (for example, Keilson and Servi (1988)), or they require the full distribution of the queue lengths (Bertsimas and Nakazato, (1995)). This leaves approximations. First, one might use the technique developed above under normality assumptions. Since the backlog quantities are expressed as sums of production vectors, this approach should provide reasonable results if the arrival distributions are not too far away from normal and the production quantities are not too highly correlated.

One can also use the production and queue length moments directly to develop simple lower and upper bounds that identify the waiting time variance within a factor of 2. If the queue length is Q_{it}, it will take approximately $Q_{it} / E[P_i]$ periods for the work at the top of the queue to leave the queue. Thus, if we ignore the variations in the production quantities, a lower bound for var(w_i) is $\text{var}[Q_{it}]/(E[P_i])^2$.

To derive an upper bound, we make several assumptions. We assume that the production quantities are independent from Q_{it}. Instead, we assume a constant production quantity, p_i, while the current work is in progress. Further, p_i is randomly selected from a random distribution with mean E[P_i] and variance var[P_i] (the steady-state production moments). Under these assumptions, it will take Q_{it} / p_i

periods for the work to leave the queue. Thus, $\text{var}(Q_{it} / p_i)$ is an upper bound on var(w_i), for two reasons. First, it ignores the relationship relating processing speed to queue length. Second, choosing p_i from a single random draw ignores the fact that the actual production quantities vary between periods while the current work is in queue. The average of several production quantities will have a lower variance than p_i. Both of these effects overestimate var(w_i).

We cannot calculate $\text{var}(Q_{it} / p_i)$ directly, but we can find a first-order approximation of it using a Taylor-Series expansion. We will discuss Taylor-series approximate methods for moment approximations in detail in Chapter 3 (for models with general control rules). For now, we state without proof that the approximate variance is:

$$\text{var}\left(\frac{Q_{it}}{p_i}\right) \approx \text{var}(P_i)\frac{(E(Q_{it}))^2}{(E(P_i))^4} + \text{var}(Q_{it})\frac{1}{(E(P_i))^2}, \tag{22}$$

which is the same as the lower bound plus the addition of a var(P_i) term. Using the relationships between the production and queue length moments, we can write both the upper and lower bounds entirely in terms of production moments, which yields a lower bound of $\text{var}[P_i]/\alpha_i^2(E[P_i])^2$, and an upper bound of:

$$\begin{aligned}\text{var}\left(\frac{Q_{it}}{p_i}\right) &\approx \text{var}(P_i)\frac{(E(P_i))^2}{\alpha_i^2(E(P_i))^4} + \text{var}(P_i)\frac{1}{\alpha_i^2(E(P_i))^2} \\ &\approx \frac{2\,\text{var}(P_i)}{\alpha_i^2(E(P_i))^2}.\end{aligned} \tag{23}$$

Thus, using only the steady-state production moments, and making no assumptions about the waiting time distributions or the relationships between queue lengths and production quantities over time, we have:

$$\frac{\text{var}(P_i)}{\alpha_i^2(E(P_i))^2} \le \text{var}(w_i) \le \frac{2\,\text{var}(P_i)}{\alpha_i^2(E(P_i))^2}. \tag{24}$$

In between these two estimates, one can assume that it will take approximately Q_{it} / p_i^* periods for work to leave the queue, where p_i* is the sum of $E(Q_{it})/E(P_{it})$ independent draws from the production distribution independent from Q_{it}. With this assumption, the estimated waiting time variance becomes:

$$\begin{aligned}\text{var}\left(\frac{Q_{it}}{p_t*}\right) &\approx \frac{\text{var}(P_i)}{(E(Q_i)/E(P_i))}\cdot\frac{(E(P_i))^2}{\alpha_i^2(E(P_i))^4} + \text{var}(P_i)\frac{1}{\alpha_i^2(E(P_i))^2} \\ &\approx \frac{(1+\alpha_i)\text{var}(P_i)}{\alpha_i^2(E(P_i))^2}, \text{since } E(Q_i)/E(P_i) = 1/\alpha_i.\end{aligned} \tag{25}$$

The drawback of this estimate is that it is neither an upper nor a lower bound. The fact that this estimate ignores the relationship between queue lengths and production quantities tends to overestimate var(w_i). However, setting p_i* to be the sum of $E(Q_{it})/E(P_{it})$ independent draws will tend to underestimate var(w_i), since production quantities are correlated over time.

2.3 A General Approach to Develop MR Models

The development of the TPM model uses the following general approach:

1. Write the control rules of the model into equations that relate production quantities to queue length quantities.
2. Write equations describing work arrivals for each station. Include work arriving to the station from outside the network, and work arriving as a result of production at upstream stations.
3. Substitute the control-rule and work-arrival equations into the standard inventory balance equations to each station, yielding a single set of recursion equations relating work (or queue lengths) completed in the last period to work (or queue lengths) in the current period.
4. Iterate the recursion equations infinitely, which will yield a power series expression for the steady-state production (or queue lengths) at all stations. Take the moments of this expression to find the steady-state expectation and variance of production (or queue lengths).
5. Use the control-rule equations to calculate the moments of the queue lengths (or production quantities, if the recursion equations were written in terms of queue lengths).

We will use this approach repeatedly to develop and analyze more complicated MR models. We begin with a simple extension, the application of the TPM to base stock models. For example, we use this approach to analyze models with general linear control rules, discussed in Chapter 4. Models with general control rules, along with the other models we consider in later chapters, are significantly more complicated than the TPM, but we will use the same basic approach to analyze them.

Example: Base Stock Models. As an example, we use the approach to derive an extended TPM that models base stock job shops. As written, the TPM assumes that work enters from outside the shop enters at particular stations (from order requests, for instance), and that this work then pushes its way through the network. In this section we consider a modification to this "push" model, called a *base stock* job shop. Here, completed products are stored in a buffer. Each time an item is removed from the buffer, the buffer sends an order to make a new item. Figure 1 diagrams a single-product base stock job shop.

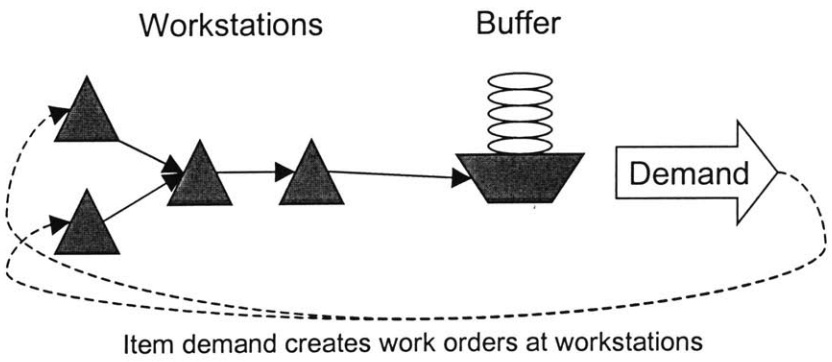


Figure 1 -- A Base Stock Job Shop

To apply the TPM to a basestock model, we first note that all stations except the buffer station behave exactly as in the original TPM. Thus, we consider removals from the buffer and the corresponding orders to the first stations in the work flows.

We assume that the number of units demanded in a given period can be treated as a continuous random variable. As with work arrivals, the demand comes from a serially i.i.d. distribution with finite mean and variance. Thus, the demand from the buffer station at time t is written as ε_{it}, except that ε_{it} is now explicitly negative. The demand has an expected value, $\mu_i < 0$, and a variance, $\Sigma_{ii} > 0$.

We now model the order requests sent to the first stations in the work flow that produce the item. Suppose that station j is one of these first stations. As with all other work arrivals, the orders to replace items leaving the buffers in time t are written as ε_{jt}; to apply the TPM, we need to find the expectation and variance of ε_{jt}. Let station j require h_{ji} time, on average, to produce one unit in the final buffer. Then, removing ε_{it} from the buffer creates $-h_{ji}\varepsilon_{it}$ of work at station j. Using this relationship, the expected quantity of the orders is $\mu_j = -h_{ji}\mu_i$, and the variance of the incoming orders is $\Sigma_{jj} = (h_{ji})^2 \Sigma_{ii}$. Further, since the orders equal a negative multiple of the units leaving the buffer, the correlation coefficient of ε_{it} and ε_{jt} equals –1. Therefore, the Σ_{ij} and Σ_{ji} entries in the input covariance matrix must be those entries that produce a correlation coefficient of –1; these entries are $\Sigma_{ij} = \Sigma_{ji} = -\sqrt{(h_{ji}\Sigma_{ii})^2}$.

In a base stock model, the expected number of products entering the buffer equals the expected number of products demanded from the buffer. (Otherwise, either the buffer's inventory or back orders would grow without bound.) Since we model demand to be negative work arrivals, the expected "production" at the buffer station will equal 0, making the size of the buffer (i.e. the buffer's expected queue length) indeterminate. From the perspective of the model the analyst may set the buffer size as desired. In practice, we may wish to make the buffer as small as possible to meet certain performance guarantees against the probability of back orders. For example, if the product demand is approximately

normal, one may want to make the size of the buffer equal to twice the standard deviation of the "production" at the buffer. This size will provide an approximately 97% chance that the buffer will not run out of items in any particular period.

We have now completed steps 1-2 of the General Approach, defining the production rules (same as the TPM) and characterizing the work arrival processes (modified for stock replenishments). Since we have now written the model parameters as TPM input matrices, we can apply the TPM moment-calculating formulas directly, completing steps 3-5 of the General Approach.

As a numerical example, consider the four-workstation plus buffer model shown in Figure 1. The following table presents a set of inputs corresponding to this model.

Workflow (Φ')	To Station				
From Station	1 (Input 1)	2 (Input 2)	3 (Process 1)	4 (Process 2)	5 (Buffer)
1 (Input 1)			1.0		
2 (Input 2)			1.0		
3 (Process 1)				1.0	
4 (Process 2)					1.0
5 (Buffer)					
Demand (μ_i)	3.0 (60% demand)	2.0 (40% demand)			-5.0
Covariance (Σ)					
1 (Input 1)	1.8				-3.0
2 (Input 2)		0.8			-2.0
3 (Process 1)			0.10		
4 (Process 2)				0.10	
5 (Buffer)	-3.0	-2.0			5.00
Lead Time	4 ($\alpha_i = 0.25$)	4	1	1	1 (N/A)

The resulting moments of production and queue lengths are:

Station	Expected Production	Standard Deviation of Production	Expected Queue Length	Standard Deviation of Queue Length
1 (Input 1)	3.0	0.508	12.0	2.028
2 (Input 2)	2.0	0.338	8.0	1.352
3 (Process 1)	5.0	0.687	5.0	0.687
4 (Process 2)	5.0	0.756	5.0	0.756
5 (Buffer)	0	2.360	N/A	N/A

As discussed, a good estimate for the size of the buffer would be twice the standard deviation of "production" at the buffer, which here is 4.72 units (or 5 units if the buffer only stores discrete objects).

3. The Tactical Planning Model for Pull-Based Systems

Much of the rest of this thesis may be seen as creating extensions and generalizations of the TPM. The first extension we consider is an adaptation allowing the TPM to model "pull-based" systems. The TPM by itself is a "push-based" system, in which each station processes a fixed fraction of the amount of

inventory in its input buffer. In 1987, however, Leong showed how to apply the TPM to "pull-based" systems, such as the popular Kanban systems. In these systems, each station now has an output buffer, from which downstream stations draw inventory; each buffer has a "target" inventory level. Production quantities no longer depend on work in the input buffer. Instead, each period each station produces enough to make up a fixed fraction of the shortfall between a target level, T_i, and the amount of inventory actually in the buffer. (For example, if the target level is 6 hours worth of on-hand inventory, the buffer only contains 4 hours of on-hand inventory, and the fixed fraction, α_i, is 0.5, the station will produce one hour's worth of inventory over the next period.)

The development of the TPM for pull-based systems is very similar to that of the original TPM. The one key difference is that the inventory balance equations are no longer written in terms of $\mathbf{Q}_t$, the inventory actually at the stations. Instead, the balance equations are written in terms of $\mathbf{V}_t$, the difference between the target inventory levels and the actual inventory levels. Mathematically, the new production rule is:

$$\begin{aligned} P_{it} &= \alpha_i V_{it}, \text{ where} \\ V_{it} &= T_i - Q_{it}. \end{aligned} \tag{26}$$

The inventory-balance equation now represents the inventory shortfall in each time period:

$$V_{it} = V_{i,t-1} - P_{i,t-1} + A_{it}, \tag{27}$$

where A_{it} is work that leaves stage i's output buffer at the start of time t, thus increasing the shortfall that must be filled. The equation for the arrivals appears the same as it was in the TPM:

$$A_{it} = \sum_j \left(\phi_{ij} P_{j,t-1} + \varepsilon_{ijt}\right) + N_{it}. \tag{28}$$

However, the meaning of the terms is different. ϕ_{ij} is the amount of work that one unit of production at stage j pulls from stage i. N_{it} is the quantity demanded externally from stage i at period t. This new characterization of work arrivals might allow one to model assembly stations that draw product components from upstream stations, for example.

Despite the change in meaning of the variables, the recursion equations have remained the same; thus, the recursion equation $\mathbf{P}_t$ is exactly what it was for the TPM, $\mathbf{P}_t = (\mathbf{I} - \mathbf{D} + \mathbf{D}\Phi)\mathbf{P}_{t-1} + \boldsymbol{\varepsilon}_t$. Consequently, we find the moments of the production and the shortfalls the same way, as well:

$$\begin{aligned}
\mathbf{E[P]} &= (\mathbf{I}-\mathbf{\Phi})^{-1}\boldsymbol{\mu} \\
\mathrm{var}[\mathbf{P}] &= \sum_{s=0}^{\infty}\mathbf{B}^s(\mathbf{D\Sigma D\Sigma D})\mathbf{B}^{s\prime}, \text{ where } \mathbf{B} = \mathbf{I}-\mathbf{D}+\mathbf{D\Phi} \\
\mathbf{E[V]} &= \mathbf{D}^{-1}\mathbf{E[P]} \\
\mathrm{var}[\mathbf{V}] &= \mathbf{D}^{-1}\mathbf{var[P]}\mathbf{D}^{-1}.
\end{aligned} \tag{29}$$

The relationship $V_{it} = T_i - Q_{it}$ immediately implies the moments of the actual inventory levels, $\mathbf{Q}_t$: we have that $\mathrm{E}[Q_i] = T_i - \mathrm{E}[V_i]$, and $\mathrm{var}[Q_i] = \mathrm{var}[V_i]$. In practice, we will want to set the T_i's large enough to ensure that the inventory levels are non-negative.

4. The General LLS-MR Model

In this chapter, we consider the general form of LLS-MR models, where production is a general linear function of the work-in-queue at multiple stations. The production rule is:

(GLR) $$P_{it} = \left(\sum_j \alpha_{ij} Q_{jt}\right) + \beta_i + \gamma_{it}, \tag{30}$$

where the parameters α_{ij} are smoothing parameters, β_i is a constant, and γ_{it} is a serially-iid random variable with zero mean and finite variance. Note that individual α_{ij}'s need not be between 0 and 1, provided that the resulting production is neither negative nor requires more inventory than that in queue (or is very unlikely to be negative or require more inventory than that in queue). Indeed, some α_{ij}'s could be negative; a negative α_{ij} implies that production is slowed down as a queue level increases. Negative α_{ij}'s can model policies that seek to prevent upstream stations from sending excess inventory to downstream stations. For example, the pull networks discussed in section 3, which produce work to eliminate shortfalls at downstream stations, may be modeled using (GLR).

Similarly, the pull network formulation discussed in section 3 models general control rules with negative α_{ij}'s. The formulation of a general control rule for pull networks is:

$$P_{it} = \left(\sum_j \alpha_{ij} V_{jt}\right) + \beta_i + \gamma_{it},$$

where V_{jt} is the inventory shortfall at station j at time t. With this convention, all α_{ij}'s are positive. Further, V_{jt} does not need to be positive. A negative V_{jt} implies that a queue has too much inventory, so that upstream stations will reduce production. The expanded V_{it} allows the modeling of

"counterbalancing" control rules; we will present an example of a model that uses counterbalancing control rules (the Denardo-Tang model) in the next section.

4.1 Motivation for the General LLS-MR Model

Control Theory. The use of general linear control rules has an important advantage: if the objective of the shop is to minimize a quadratic cost function of production and inventory levels, a linear control rule will minimize the objective. For example, consider the following dynamic minimization problem:

$$(V\text{-}MIN) \qquad \min E\left\{\sum_{t=0}^{\infty} d^t\left[(\mathbf{P}_t - \mathbf{E[P]})'\mathbf{C}_P(\mathbf{P}_t - \mathbf{E[P]}) + (\mathbf{Q}_{t-1} - \mathbf{Q}^*)'\mathbf{C}_Q(\mathbf{Q}_{t-1} - \mathbf{Q}^*)\right]\right\} \tag{31}$$
$$\text{s.t. } \mathbf{Q}_t = \mathbf{Q}_{t-1} + (\mathbf{\Phi} - \mathbf{I})\mathbf{P}_{t-1} + \boldsymbol{\varepsilon}_t, \forall t,$$

where d is a discount factor, $\mathbf{Q}^*$ is a vector of target queue levels, $\mathbf{C}_P$ is a matrix setting the cost for production quantity fluctuations, and $\mathbf{C}_Q$ is a matrix setting the cost for queue level fluctuations. The objective function seeks to minimize the variances of production and queue lengths over a discounted infinite time horizon. The recursion equations for $\mathbf{Q}_t$ are the TPM recursion equations, with ε_t being a serially-i.i.d. noise vector, as previously discussed . Consequently, problem *V-MIN* seeks the production quantities that will minimize the production and queue length variances within a discrete-time manufacturing model over an infinite discounted time horizon– a problem we expect might be of great interest in a variety of manufacturing applications.

The solution to *V-MIN* will be discussed in detail in Chapter 7, which examines dynamic control of MR-models. For now, however we state that the formula for the optimal production quantities will be a linear function of the queue lengths of the form:

$$\mathbf{P}_t = \mathbf{E[P]} + \mathbf{K}_t(\mathbf{Q}_t - \mathbf{Q}^*), \tag{32}$$

where $\mathbf{K}_t$ is a matrix. Further, in steady state, $\mathbf{K}_t$ will converge to a constant matrix $\mathbf{K}$ provided that $d < 1$. (These results come from the application of dynamic programming theory to linear systems with quadratic cost; see, for example, Bertsekas (1995a)). Then, the optimal steady-state production quantities in the discrete-time manufacturing model will be given by a general linear control rule.

Processing Time Predictability. In addition to minimizing production and queue length variances, linear control rules make waiting times predictable, as well. In the single-station example in Section 2, 96% of all the work in queue waited between 4 and 6 periods, despite fairly high fluctuations in the arrival stream to the station.

Modeling of Adaptive Control Rules. Beyond optimal policies for quadratic-cost systems, the general LLS-MR model allows the analyst to model other control rules which modify station production rates to limit production and inventory fluctuations throughout the entire shop. This modeling ability is important for just-in-time manufacturing applications, for instance.

One example of adaptive control rules is the *proportional restoration* control rule suggested by Denardo and Tang (1997). These rules are pull models where stations' productions are controlled by the need to restock downstream inventories. In particular, the production at each station is the amount needed to restock some fraction of the shortfall across all downstream inventories. Job shops with proportional restoration rules will be discussed in the next section.

Production Fluctuations. Allowing random fluctuations in production quantities is also an important extension. This addition allows the analyst to model situations in which production does not depend exactly on queue lengths – which will likely include many situations.

Empirical Evidence for Linear Control Rules. There is empirical evidence that stations may follow linear control rules in practice. For example, Fine and Graves (1989) showed that many stations at a mainframe subcomponent manufacturing plant empirically obeyed a linear control rule.

4.2 Model Development

We develop the general linear control rule model for push networks. The equations for pull networks are similar, except that the $\mathbf{Q}_t$'s are replaced by $\mathbf{V}_t$'s. We write the production rules simultaneously in matrix-vector form as:

$$\mathbf{P}_t = \mathbf{D}\mathbf{Q}_t + \boldsymbol{\beta} + \boldsymbol{\gamma}_t, \text{or } \mathbf{Q}_t = \mathbf{D}^{-1}(\mathbf{P}_t - \boldsymbol{\beta} - \boldsymbol{\gamma}_t). \tag{33}$$

Here, $\mathbf{D}$ is a non-diagonal matrix, assumed to be non-singular. Solving the equation for $\mathbf{Q}_t$ and substituting the results into the inventory balance equation, $\mathbf{Q}_t = \mathbf{Q}_{t-1} - \mathbf{P}_{t-1} + \mathbf{A}_t$, yields:

$$\begin{aligned} \mathbf{D}^{-1}(\mathbf{P}_t - \boldsymbol{\beta} - \boldsymbol{\gamma}_t) &= \mathbf{D}^{-1}(\mathbf{P}_{t-1} - \boldsymbol{\beta} - \boldsymbol{\gamma}_{t-1}) - \mathbf{P}_{t-1} + \mathbf{A}_t, \text{where } \mathbf{A}_t = \boldsymbol{\Phi}\mathbf{P}_{t-1} + \boldsymbol{\varepsilon}_t \\ \Rightarrow \mathbf{P}_t &= (\mathbf{I} - \mathbf{D} + \mathbf{D}\boldsymbol{\Phi})\mathbf{P}_{t-1} + \mathbf{D}\boldsymbol{\varepsilon}_t + (\boldsymbol{\gamma}_t - \boldsymbol{\gamma}_{t-1}). \end{aligned} \tag{34}$$

The constant production term, β, cancels out of (30). The term will reappear in the equations for the queue length moments, however.

By iterating (30) and assuming an infinite history of the system, we find the following power series equation (similar to the results for the TPM):

$$\mathbf{P}_t = \boldsymbol{\gamma}_t + \sum_{s=0}^{\infty} \mathbf{B}^s \left(\mathbf{D}\boldsymbol{\varepsilon}_{t-s} + (\mathbf{B}-\mathbf{I})\boldsymbol{\gamma}_{t-s-1} \right), \quad \text{where } \mathbf{B} = \mathbf{I} - \mathbf{D} + \mathbf{D}\boldsymbol{\Phi}. \tag{35}$$

Since the mean of $\boldsymbol{\gamma}_t$ equals zero, E[**P**] is:

$$\begin{aligned} E[\mathbf{P}] &= (\mathbf{I}-\mathbf{B})^{-1}\mathbf{D}\boldsymbol{\mu} \\ &= (\mathbf{I}-\boldsymbol{\Phi})^{-1}\boldsymbol{\mu}. \end{aligned} \tag{36}$$

The expected production in this model is exactly the same in the general linear model and in the TPM. This is not a coincidence. In the next chapter, we will show that if the production rule is any general function of the queue lengths such that the resulting network is stable, the expected production at each station will always be $E[\mathbf{P}] = (\mathbf{I}-\boldsymbol{\Phi})^{-1}\boldsymbol{\mu}$.

The steady-state variance of **P** is similar to what it was for the TPM. To account for the $\boldsymbol{\gamma}_t$'s, we add an additional term to the infinite series:

$$\mathbf{S} = \mathrm{var}(\mathbf{P}_t) = \boldsymbol{\Gamma} + \mathbf{D}\boldsymbol{\Sigma}\mathbf{D}' + \sum_{s=1}^{\infty} \mathbf{B}^s \left(\mathbf{D}\boldsymbol{\Sigma}\mathbf{D}' + (\mathbf{B}-\mathbf{I})\boldsymbol{\Gamma}(\mathbf{B}-\mathbf{I})' \right)\mathbf{B}'^s, \quad \text{where } \mathbf{B} = \mathbf{I} - \mathbf{D} + \mathbf{D}\Phi, \text{and } \boldsymbol{\Gamma} = \mathrm{cov}(\boldsymbol{\gamma}_t) \tag{37}$$

The moments of $\mathbf{P}_t$ imply the moments of the queue lengths, $\mathbf{Q}_t$:

$$\begin{aligned} E[\mathbf{Q}] &= \mathbf{D}^{-1}(\mathbf{E}[\mathbf{P}] - \boldsymbol{\beta}), \text{ and} \\ \mathrm{var}[\mathbf{Q}] &= \mathbf{D}^{-1}(\mathbf{S} - \boldsymbol{\Gamma})(\mathbf{D}^{-1})' \end{aligned} \tag{38}$$

The calculation of E[**Q**] follows directly from the equation $\mathbf{Q}_t = \mathbf{D}^{-1}(\mathbf{P}_t - \boldsymbol{\beta} - \boldsymbol{\gamma}_t)$. The calculation of var[**Q**] is a more subtle. Note that $\boldsymbol{\gamma}_t$ appears in the recursion equation for $\mathbf{P}_t$, whereas $-\boldsymbol{\gamma}_t$ appears in the equation for $\mathbf{Q}_t$. Thus, the contribution of $\boldsymbol{\gamma}_t$ is negated, so the variance of $\boldsymbol{\gamma}_t$ must be removed from the variance of $\mathbf{Q}_t$. (This is not an issue with E[**Q**], since E[$\boldsymbol{\gamma}_t$] = 0.)

For pull models, we find the moments of the shortfalls, E[**V**] and var[**V**] rather than E[**Q**] and var[**Q**]. We find the moments of the queue lengths using the same formulas as the TPM for pull networks: $\mathbf{E}[\mathbf{Q}] = \mathbf{T} - \mathbf{E}[\mathbf{V}]$, where **T** is the vector of target inventories, and $\mathbf{var}[\mathbf{Q}] = \mathbf{var}[\mathbf{V}]$.

5. Example: A Stochastic Production System Using Proportional Restoration Control Rules

In this section, we apply general linear control rule models to a stochastic production system using the *proportional restoration* rules developed by Denardo and Tang (1997). As discussed, these rules are pull

models where stations' productions are controlled by the need to restock downstream inventories. In particular, the production at each station is the amount needed to restock some fraction of the shortfall across all downstream inventories. We first derive the MR model that uses proportional restoration rules, and then apply the model to a problem suggested by Denardo and Tang (1997).

5.1 Model Development

Mathematically, the production of station i at time t is given by:

$$P_{it} = E(P_i) + \sum_{j:\Phi_{ji}>0}^{N} \left(E(Q_j) - Q_{jt}\right) r_j \Phi_{ij}^* + \gamma_{it}, \tag{39}$$

or, in accordance with the notation of the pull models,

$$P_{it} = E(P_i) + \sum_{j:\Phi_{ji}>0}^{N} r_j \Phi_{ij}^* V_{jt} + \gamma_{it} \tag{40}$$

Here, $E(Q_j)$ is a *target buffer size,* or desired queue level for station j, r_j is a control factor set by the user such that $0 \le r_j \le 1$, and Φ_{ij}^* is the estimated work that must be completed at i to get one unit of work to arrive at station j. In words, this rule says that the production at station i is the expected production plus or minus a restoration quantity that partially restores the queues at downstream stations to the desired expected queue lengths. On average, r_j of the shortfall (or excess) at station j will be restored each period.

As with the TPM models discussed previously, we continue to assume that ε_t represents fluctuations in work arrivals. As before, ε_t is a serially-i.i.d. random variable with mean μ and covariance matrix Σ.

The proportional control rule is a general linear control rule for a pull network with:

$$\begin{gathered} \alpha_{ij} = r_j \Phi_{ij}^*, \\ \beta_i = E(P_i), \text{and} \\ \gamma_{it} \text{ is a serially-iid random variable with zero mean.} \end{gathered} \tag{41}$$

To use the model equation derived in the previous section, we simply need to find the $E(P_i)$'s and Φ_{ij}*'s. As discussed in the previous section, regardless of the α_{ij}'s, $E(\mathbf{P}) = (\mathbf{I} - \mathbf{\Phi})^{-1}\boldsymbol{\mu}$. Using the derivation of pull models discussed in Section 3, Φ_{ij}^* equals the expected amount of work one unit of demand at station j induces at station i. Thus, each Φ_{ij}^* element is the (ij) element of the matrix $\mathbf{\Phi}^* = (\mathbf{I} - \mathbf{\Phi})^{-1}$.

We apply the equations in Section 4 directly, yielding E(**P**), var(**P**), E(**V**), var(**V**), E(**Q**), and var(**Q**). E(**V**) merits a special comment; using the formula in Section 4, we see that

$$E[\mathbf{V}] = \mathbf{D}^{-1}\left(\mathbf{E}[\mathbf{P}] - \boldsymbol{\beta}\right) = \mathbf{D}^{-1}\left(\mathbf{E}[\mathbf{P}] - \mathbf{E}[\mathbf{P}]\right) = 0\,, \tag{42}$$

implying that the expected queue lengths will be whatever the target buffer sizes are. Mathematically, we have freedom to choose any target buffer sizes. In practice, we want to choose target buffer sizes large enough that the probability of stockouts are small. If the production fluctuations are approximately normally distributed, a good choice is to make the buffer sizes twice the standard deviation of the queue lengths (this is the approach recommended by Denardo and Tang (1997)). This rule gives rise to nonlinear programs to find the r_j's minimizing the required buffer sizes; these nonlinear programs will be discussed in some detail in Chapter 6, on the optimization of MR Models.

We conclude the development by noting that Denardo and Tang's model is not a MR model. Denardo and Tang modeled production systems using an underlying Markov chain, such that production fluctuations are due to work discarded in accordance with a binomial distribution. They also assumed that the shop contains no cycles in the work flows.

The MR-models behave differently. They do include a Markov assumption that work requirements at a station do not depend on how a particular workflow got to that station. However, instead of Binomial distributions, the fluctuations in production and work transfers may be modeled by any distributions *independent* from the production and queue length quantities. The latter assumption prevents MR models from directly copying Denardo and Tang's models, since the latter's production variances are directly proportional to the amount of work produced. Further, MR-models do allow cycles in work flows; transitions showing work returning to upstream station are treated like any other transitions. The latter ability models job shops that subject items failing quality tests to rework, for example.

5.2 An Example Production System

Denardo and Tang (1997) present a six-station assembly line controlled using proportional restoration rules. Finish products are removed from the buffer at station 1, while station 6 receives inventories from an infinite source. The assembly line is shown in Figure 2.

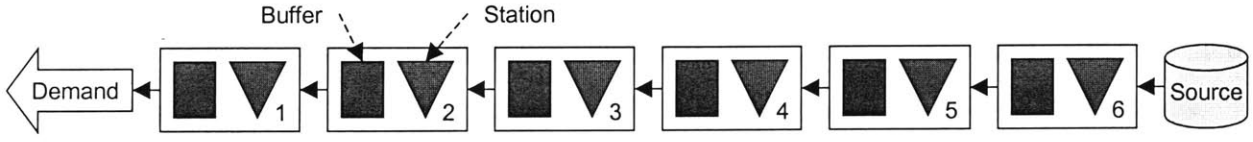


Figure 2 – A Six-Station Assembly Line

External demand at station 1 is Poisson with a mean and variance of 20 units. At each station, the failure rate is 15%, so one unit of demand at station i induces an average of 1/.85 units of demand at station $i + 1$.

As discussed, we cannot model Binomial-distribution failures directly with MR-models, so we approximate the variance caused by failures at each station to be $\Sigma_{ii} = E(P_i)(0.15)(0.85)$, using the fact that the true variance of Binomial distribution failures would be $\Sigma_{ii} = P_i(0.15)(0.85)$. (This particular approximation underestimates the variances somewhat.)

Finally, Denardo and Tang suggest the control parameters $r_1 = 0.25$, $r_2 = 0.35$, $r_3 = 0.43$, $r_4 = 0.51$, $r_5 = 0.60$, and $r_6 = 0.68$. Together, these inputs yield the following model parameters:

Workflow, Φ'	To Station					
From Station	1	2	3	4	5	6
1		1.176				
2			1.176			
3				1.176		
4					1.176	
5						1.176
6						
Demand, μ_i	23.53	0	0	0	0	0
Expected production, E(P_i)	20/0.85 = 23.53	23.53/0.85 = 27.68	32.57	38.31	45.07	53.03
Input variance, Σ_{ii}	23.53*0.15*0.85 + 20 = 23.00	27.68*0.15* 0.85 = 3.53	4.15	4.88	5.75	6.76
r_i	0.25	0.35	0.43	0.51	0.60	0.68

Running the model yields the following results. Note that the recommended buffer size is twice the standard deviation of the queue length:

Station	Expected Production	Standard Deviation of Production	Standard Deviation of Queue Lengths	Recommended Buffer Size
1	23.53	2.51	7.25	14.50
2	27.68	2.87	3.65	7.30
3	32.57	3.30	3.66	7.32
4	38.31	3.79	3.74	7.48
5	45.07	4.36	3.86	7.72
6	53.03	5.03	4.05	8.10

6. Example: A Communications Problem

6.1 Introduction

Linear-MR models provide a great deal of flexibility to model (and optimize) a wide variety of situations which can be represented by a "processing network." Here, we consider a communications application similar to capacity-allocation problems faced by the US Department of Defense.

In this example, a set of antennas broadcast signals at probabilistic intervals; broadcasts may be correlated (positively or negatively) with each other. Each broadcast signal uses a random amount of bandwidth with a finite mean and standard deviation. These signals are received by a set of remote receivers; each receiver transmits the signals it receives to a communications satellite, which transmits collections of signals to a central base. Analysts at the central base then send control signals back to the remote receiver stations requesting additional information about particular signals, which the receiver stations provide through the same connections as the original signal. It is assumed that a receiver or a satellite cannot store signals; signals must be transmitted immediately or dropped. The objective is to determine how much bandwidth must be installed at each receiver and satellite to meet certain levels of service (i.e. avoid dropping too many signals.) Figure 3 graphically shows the situation.

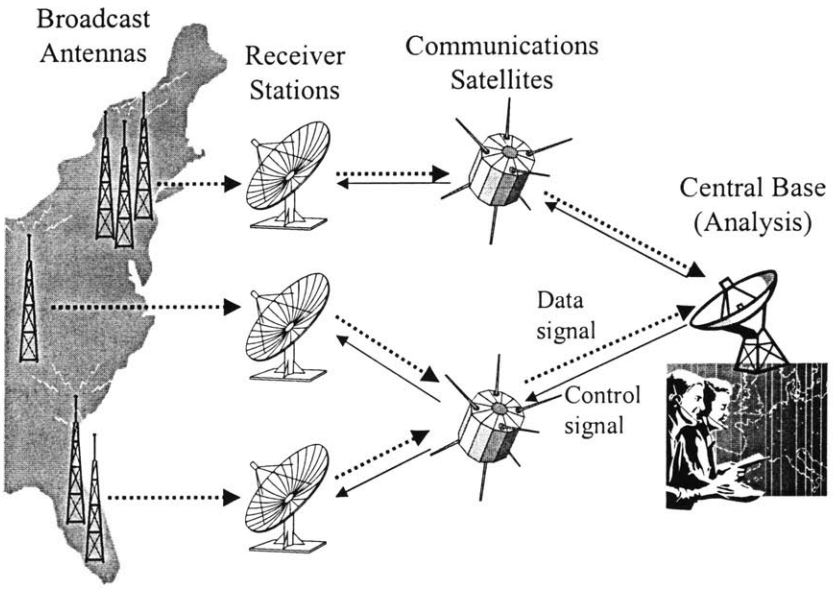


Figure 3 -- A Communications Example from the Department of Defense

6.2 Model Development

At first glance, this scenario appears not to be relevant to MR models; no work is queued anywhere in the model, and the Markovian assumption is not satisfied directly (different signals require separate handling by the same receivers, satellites, and analysts). However, this situation is amenable to MR modeling with a few minor adjustments.

First, "no work queued anywhere in the system" is equivalent to requiring the lead times at all stations to be one. This forces the stations to process all work within a single period. Assuming the "period" for this communications period is quite short (a second or less), the resulting MR model should realistically represent near-instantaneous communications. Second, the signal bandwidth at any time is

the same across all stations handling that signal, since the bandwidth depends on the signal itself, not interactions between stations handling the signal. Thus, in the MR model, we have one station that "transmits" the signal; this station sends work flows directly to all the stations that receive the signal. For example, a signal's "antenna" station directly sends work to a receiver station, a satellite station, and a base station. The receiver station, satellite station, and base station do not send work to each other.

To address distinct signal handling, we add stations to the model as follows: there is one dummy station representing every particular signal handled by a particular communications station (receiver, satellite, or analysis base). We then apply the steady-state moment equations to the resulting "extended" model. Using the linearity of expectation, we know that the expected throughput at a communications station is the sum of all the dummy stations associated with the station. The variance of the throughput is the sum of the variances of the dummy stations, plus all the covariance terms between those stations. The covariance terms are especially important in this case; in addition to multiple signals being correlated with each other, all the stations that handle a single signal are perfectly correlated with each other as they will see the exact same bandwidth at any given time.

Knowing the expectation and variance of the throughput implies how much capacity will be needed to meet desired levels of service; the capacity that needs to be assigned to each communications station is given by:

$$C_i = E[P_i] + k_i \sqrt{\text{var}(P_i)}\,, \tag{43}$$

where C_i is the required capacity, and k_i is a *safety factor* used to translate variance of production information into level of service requirements. A factor of 2 is often used in practice, since, if the production levels were normally distributed, the resulting capacity would be able to handle the required load about 97% of the time. In our scenario, however, "production levels" (i.e. data signals) will not be normally distributed given that they are convolutions of a distribution determining whether a source is broadcasting and a distribution determining how much data is generated assuming the source is broadcasting. Thus, simply applying a safety factor of 2 will be inappropriate.

Nonetheless, there are more appropriate ways to calculate the safety factors. We will discuss an analytic approach to estimating the needed safety factor at the end of this section. Alternately, analysts may perform simple Monte-Carlo simulations to translate the expectation and variance results to capacity levels required to meet service requirements. These simulations would be simple single-distribution runs that could be completed in a few tenths of a second; discrete event simulations are not required.

We now present a small example of the communications problem. We have six broadcast antennas, three receivers, and two communications satellites. The six antennas can be divided into three groups, with 1, 2, and 3 antennas per group. If two antennas are in the same group, their broadcasts are

positively correlated; if two antennas are in different groups, their broadcasts are independent. Figure 3 (presented previously) shows how the three groups of antennas, the receiver stations, and the communications satellites communicate with each other.

To model this scenario in terms of a linear MR model, we create one work station for every particular signal handled by a communications device. Here, each signal travels through a receiver, a satellite and a base; the signal then creates a corresponding control stream that travels from the base, back through the satellite to the receiver. Thus, there are six stations per signal: one representing the broadcast antenna, one representing the base, and four representing the data and control streams to and from the receiver and satellite. Since there are six signals, the model has a total of 36 stations. Figure 4 shows the signal flows between the stations that will be used in the MR model.

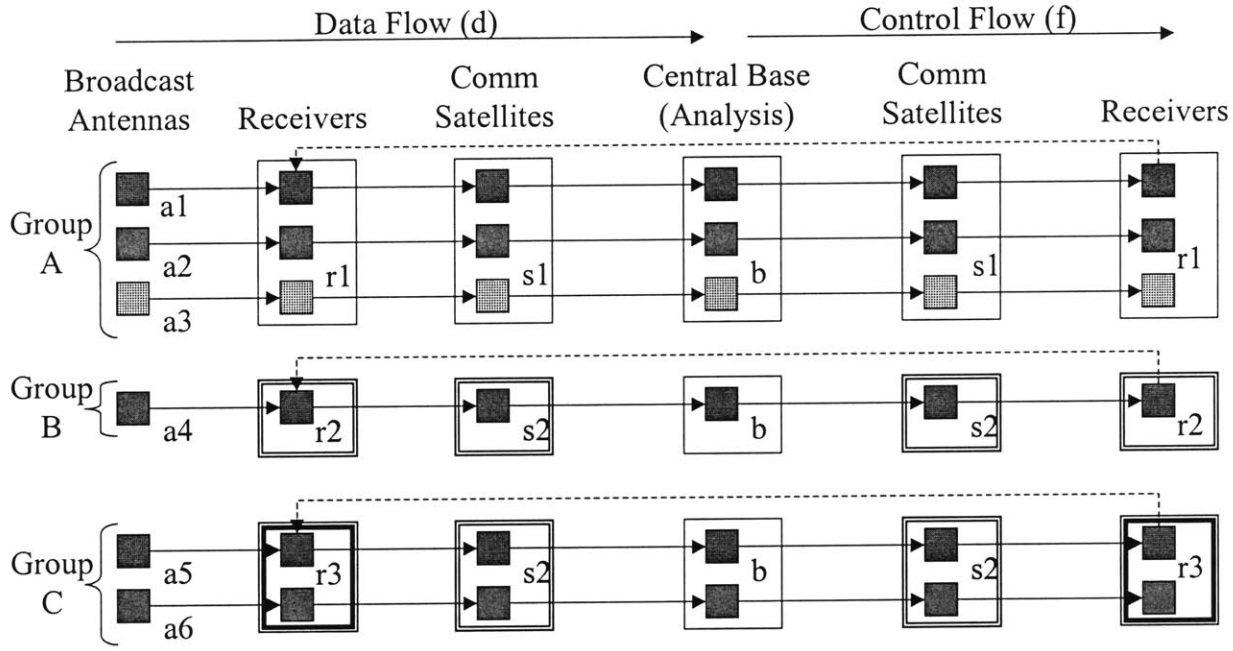


Figure 4 – Signal Flows in the Physical Communications Network

In the figure, the arcs represent data flows between stations. Note that not all the arcs between the control-flow receiver stations and the data-flow receiver stations are shown. The boxes group the stations that correspond to a processing step for a correlated group of signals on a single piece of equipment. For example, there is a box corresponding to receiver *r1* processing all signals within signal group *1*.

Although Figure 4 shows all the stations in the MR model, and the physical signal flows between these stations, the figure does not show the actual layout of the model. As previously discussed, all the stations handling a given signal will see the same bandwidth at any time (data stations will see the whole signal, and control stations will see a fraction of the signal). It is not the case that the stations transmit

signal chunks across multiple time periods. This situation is modeled accurately by having each signal's antenna station send work to the stations that receive the signal simultaneously. Figure 5 shows the antenna stations broadcasting to the receiver, satellite, and base stations simultaneously, and thus shows the true layout of the MR model.

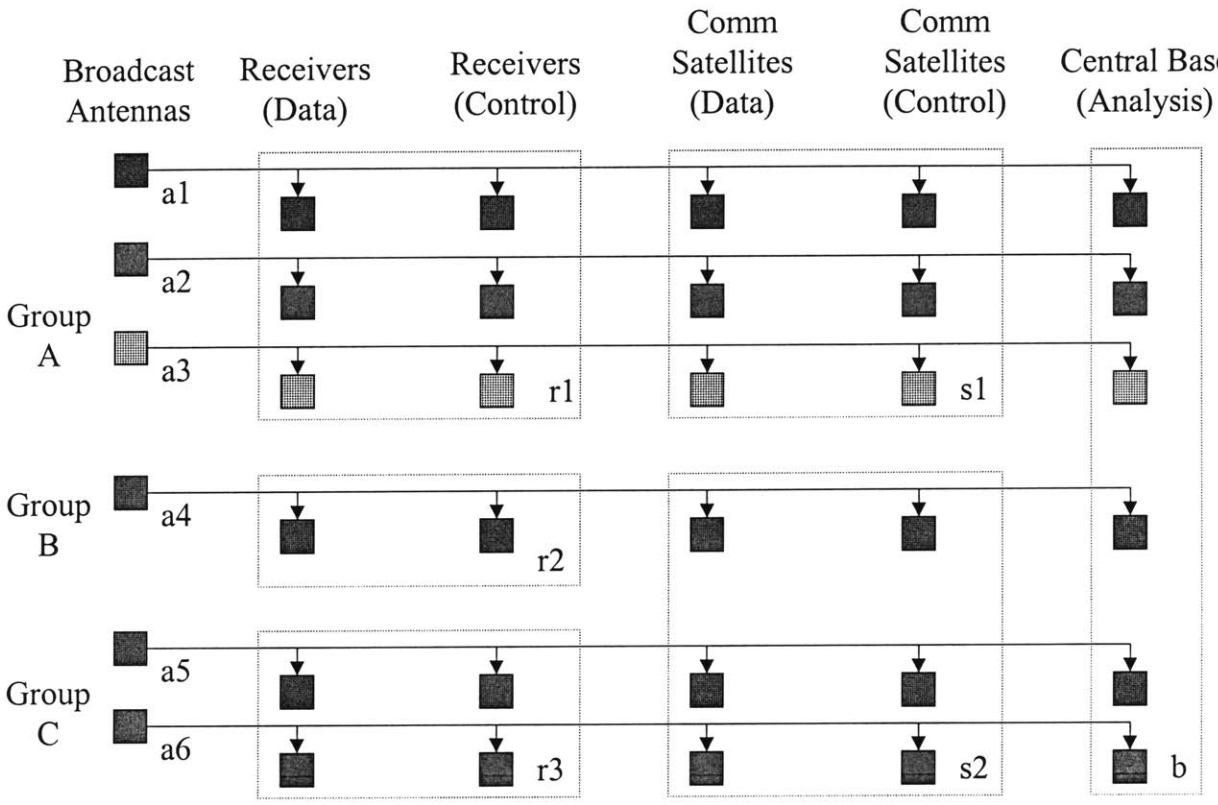


Figure 5 -- Model Representation of the Network Example

Figure 5 also shows the stations that comprise the signal handling required by each communications device. Recall that we are most interested in the aggregate capacity required for each communications device, which involves combining the results for all the stations corresponding to a piece of communications equipment. For example, the aggregate capacity on communications satellite*s2* is a function of the six stations in the box labeled "*s2*".

We are also interested in the aggregate capacity throughout the system required for a particular signal. This estimate involves combining the results for all the stations corresponding to that signal. For example, the aggregate capacity required by group 1's third signal is a function of all the stations connected to station *a3*.

As noted, the aggregate capacity recommended for a device is a function of the expectation and variance of the total throughput through that device. The total expected throughput equals the sum of the

expected throughputs at all stations corresponding to that device. The total throughput variance equals the sum of all the variance and covariance terms between the stations corresponding to the device.

The time required to estimate the throughput moments is proportional to the cube of the number of stations. While 36 stations should not be a problem, stations with hundreds or thousands of signals could cause serious computational difficulties. Fortunately, in this problem we can decompose our single model representation into multiple representations. In our models, stations that correspond to separate signals do not send information to each other. The only way these stations interact with each other at all is if their corresponding signals are in the same group (i.e. correlated with each other). If not, the stations do not interact with each other, so any aggregate results will not reflect any interactions between the stations. Thus, we can decompose the single model into a completely separate model for the stations corresponding to signals in each group. In our example, we therefore have three independent models, shown in Figure 3 by the lines separating the "Group A", "Group B", and "Group C" stations. If a communications device is represented in multiple models (for example, the satellite $s2$), its aggregate expected throughput is the sum of the corresponding expected throughputs for each model, and the aggregate throughput variance is the sum of the corresponding throughput variances for each model. (The latter statement holds since all covariance terms between stations in separate models are zero.)

6.3 Converting System Data to Model Inputs

We have the following numerical data for each signal:

Signal	**1**	**2**	**3**	**4**	**5**	**6**
Signal Group	A	A	A	B	B	C
Broadcast Probability	0.2	0.3	0.1	0.05	0.15	0.2
Broadcast Correlation Coefficient for Group	0.3	0.3	0.3	0.5	0.5	NA
Mean Signal Bandwidth (Mbits/sec)	2	3	4	10	1	5
Standard Deviation of Signal Bandwidth (Base)	0.2	0.05	1.5	5	0.8	2
Mean Bandwidth of Corresponding Control Stream as a fraction of signal bandwidth	0.2	0.3	0.4	0.2	0.2	1
Standard Deviation of Control Bandwidth	0.1	0.1	0.1	0.1	0.1	0.4
Standard Deviation of Signal Bandwidth that results from control stream instructions	0.2	0.2	0.5	1	0	0.5

We now use this numerical data to define the matrices used in the MR equations. In defining the entries for the matrices, we use the following subscripts:

- a: broadcast antenna station
- r: receiver station
- s: satellite station
- b: central base

- d: station that is part of a signal data flow to the base
- f: station that is part of a control flow from the base
- i,j: indices of a particular signal
- k,l: indices of a particular communications device

Entries for broadcast antenna stations. These are the only stations that receive arrivals from outside the network of stations, representing the generated signals. Three factors enter into the matrix entries. First, each station has a probability that it will broadcast at any particular time. Second, these broadcasting periods are positively correlated with each other. Third, when broadcasting, each station generates a signal according to a random distribution with the mean and standard deviation given above.

Then, the overall mean signal generated by antenna station ai is:

$$\mu_{ai} = p_{ai} m_{ai}, \tag{44}$$

and the input variance of the signal generated by antenna station ai is:

$$\Sigma_{ai,ai} = p_{ai} s_{ai}^2 + p_{ai}(1-p)_{ai}(m_{ai})^2, \tag{45}$$

where p_{ai} is the probability that station ai is broadcasting, m_{ai} is the mean bandwidth when the station is broadcasting, and s_{ai}^2 is the variance of the bandwidth when the station is broadcasting. (These formulas both result from the moment formulas for a random sum of random variables.) The covariance between each pair of broadcast stations in the same group is:

$$\Sigma_{ai,aj} = \rho_{ai,aj} \sqrt{m_{ai} p_{ai}(1-p_{ai}) m_{bi} p_{aj}(1-p_{aj})}, \tag{46}$$

where $\rho_{ai,aj}$ is the correlation coefficient between the broadcasting of stations i and j. As discussed, the covariance between stations not in the same group is 0, which for separate models representing each group of signals. Note also that the covariance does not depend on the variance of the bandwidth when the stations are actually broadcasting, because signal bandwidths are assumed to be independent.

We conclude by noting that the broadcast antenna stations do not receive any flow from other stations. Thus, all workflow matrix (Φ) entries representing expected work arriving from other station to antenna stations equal 0.

Entries for receiver stations. There are two sets of receiver stations; the first set represents signal data (marked by subscript d), and the second set represents control flows (marked by subscript f).

We consider the signal-processing stations first. The base signal that enters the receiver station is the same signal emitted from the broadcast antenna. Thus, the workflow matrix entry representing flow between antenna and receiver stations equals 1, or:

$\Phi_{rdi,ai} = 1$, for all signals *i*.

The control flow for each signal affects the amount of the base signal that the signal-processing stations have to transmit. This effect is modeled through a random variable with zero mean and a finite variance. It is the only effect that adds to the variance of the throughput at this station. Thus, the input variance for signal-processing stations is:

$\Sigma_{rdi,rdi} = \sigma^2_{rsi,rfi}$,

where the second term is the variance that control-flow station *rfi* induces at station *rsi*.

We next consider the control-processing stations. They receive the control flow transmitted to them by their communications satellite. As previously discussed, this transmission is modeled by a direct flow from the signal's antenna station to the receiver station. Thus, the workflow matrix entries are:

$\Phi_{rfi,ai} = \phi_{fi}$, for all signals *i*,

where ϕ_{fi} is the expected control bandwidth as a percentage of the signal bandwidth.

The input variance entries are:

$\Sigma_{rfi,rfi} = \sigma^2_{fi}$, for all signals i,

where σ^2_{fi} is the variance of the control bandwidth.

Entries for satellite stations. Again, there are signal data and control flow satellite stations. Signal data stations receive the signals passed to them their receivers, which is just the base signal data. Then the workflow matrix entries are:

$\Phi_{sdi,ai} = 1$, for all signals *i*.

The input variance entries are the same as for the data receiver stations. The signal fluctuations entering the receiver station are duplicated in the satellite stations, since the communications satellite does not alter the signal data it receives. The only signal fluctuations result from the influence of the signal control flows, as described in the discussion of data receiver stations. Thus, the input variance entries are:

$\Sigma_{sdi,sdi} = \sigma^2_{rsi,rfi}$,

the same as the data receiver stations.

Control-flow satellite stations are the first stations to receive control signals (broadcasted from the base), but this fact is of little modeling significance; like all other stations, they receive work directly

from the their signal's antenna station. Thus, the workflow matrix entry representing the control signal's flow equals:

$\Phi_{sfi,bi} = \phi_{fi}$, for all signals *i*,

where the second term is the expected control bandwidth as a percentage of the signal bandwidth. The input variance entries are:

$\Sigma_{sfi,rfi} = \sigma^2_{fi}$, for all signals *i*,

where the second term equals the variance of the control bandwidth

Entries for base stations. There is only a single set of base stations, which both receive signals from the satellites and transmit control flows to the communications satellites. In the model, only the base stations' receipt of the data signal appears, since all stations receive work from their signal's antenna station. Base stations receive the same bandwidth as the other data stations. Thus, the workflow matrix entries are:

$\Phi_{bi,ai} = 1$, for all signals *i*,

and the input variance entries are:

$\Sigma_{bi,bi} = \sigma^2_{rsi,rfi}$,

where the second term shows the influence of the control flow on the amount of the base signal transmitted.

6.4 Calculating the Results

We have developed MATLAB scripts that convert the signal data into model inputs, generate the resulting three MR models (one model for each group), calculate the throughput moments for each station, and aggregate the results for each communications device in the system. The results are shown below. The following table displays the expected throughput in Mbits/sec (first number) and standard deviation of the throughput (second number) for each of the 36 stations in the model.

	Signal Data								Signal Control			
Signal	**Antennas**		**Receivers**		**Satellites**		**Base**		**Satellites**		**Receivers**	
1	0.4	0.8	0.4	0.83	0.4	0.83	0.4	0.83	0.08	0.19	0.08	0.19
2	0.9	1.38	0.9	1.39	0.9	1.39	0.9	1.39	0.27	0.43	0.27	0.43
3	0.4	1.29	0.4	1.38	0.4	1.38	0.4	1.38	0.16	0.84	0.16	0.84
4	0.5	2.45	0.5	2.65	0.5	2.65	0.5	2.65	0.1	0.54	0.1	0.54
5	0.15	0.53	0.15	0.53	0.15	0.53	0.15	0.53	0.06	0.23	0.06	0.23
6	1	2.19	1	2.25	1	2.25	1	2.25	1	2.25	1	2.25

We then find the expectation and standard deviation for the throughput for each communications device. Recall that the 36 stations each correspond to the data or control processing of one signal on a particular

device; the stations corresponding to each device were labeled in Figure 2. The expected throughput on a device equals the sum of the expected throughput of each of the device's stations. The variance of a device's throughput equals the sum of all of the variance and covariance terms of each of the device's stations. The following table displays the expectation and standard deviation of the throughput for each of the communications devices. The table also displays the signal flows processed by each device.

Communications Device	Signals Processed	Expected Throughput	Standard Deviation
Antenna 1 (a1)	1, data	0.4	0.8
Antenna 2 (a2)	2, data	0.9	1.38
Antenna 3 (a3)	3, data	0.4	1.29
Antenna 4 (a4)	4, data	0.5	2.45
Antenna 5 (a5)	5, data	0.15	0.53
Antenna 6 (a6)	6, data	1	2.19
Receiver 1 (r1)	1, 2, 3, data and control	2.21	3.4
Receiver 2 (r2)	4, data and control	0.6	3.11
Receiver 3 (r3)	5,6, data and control	2.21	4.73
Satellite 1 (s1)	1, 2, 3, data and control	2.21	3.4
Satellite 2 (s2)	4, 5, 6, data and control	2.81	5.66
Base (b)	All, data	3.35	4.45

6.5 *Interpreting the Results*

6.5.1 Single Station Results

Recall that the recommended capacity for each station (and communications device) is a function of the expectation and standard deviation of the throughput at that station, as follows:

$$C_i = E(T_i) + k_i \cdot \sigma_{Ti} \ ,$$

where C_i is the recommended capacity for station i, $E(T_i)$ is the expected throughput, σ_{Ti} is the standard deviation of the throughput, and k_i is the "safety" factor that converts the standard deviation of the throughput to a performance guarantee. This equation is well-defined except for k_i.

As discussed, a k_i of 2 is not appropriate in this scenario. The throughput distributions are not normally distributed – they are the convolution of a Bernoulli trial and a general distribution. Usually, the throughput is 0, corresponding to the antenna not broadcasting a signal. Other times, with probabilityp_i, the throughput's value will come from the distribution for a signal's bandwidth given that the signal is being broadcast. Thus, setting the safety factor to 2 does not translate to a performance guarantee.

However, we can use the fact that the distribution at each station is a convolution of a Bernoulli trial and another distribution to generate more appropriate k-factors for individual stations. The Department of Defense is interested in performance guarantees conditional on a signal actually being transmitted. For example, the Department might wish to guarantee that a signal, when transmitted, has a

97% chance of being processed at any given moment. Assuming the distribution for a signal's bandwidth is approximately normal, the capacity needed to process the signal is:

$$C_i = E(B_i) + 2 \cdot \sigma_{Bi},$$

where $E(B_i)$ is the expected bandwidth of the signal, and σ_{Bi} is the standard deviation of the bandwidth used by the signal, when transmitted. We would like the capacity recommendations that result from the throughput moments and the above equation to be the same. Thus, we can equate the two formulas for capacity and solve for k_i:

$$\begin{aligned} & E(T_i) + k_i \cdot \sigma_{Ti} = E(B_i) + 2 \cdot \sigma_{Bi} \\ \Rightarrow\ & k_i = \frac{E(B_i) - E(T_i) + 2 \cdot \sigma_{Bi}}{\sigma_{Ti}}. \end{aligned} \tag{47}$$

The terms $E(T_i)$ and σ_{Ti} are given by the MR model, while $E(B_i)$ and σ_{Bi} are not known directly. However, we now use the fact that the throughput distribution at each station is a convolution of a Bernoulli trial (whose parameter, p_i, is known) and the distribution for the signal's bandwidth given the signal is being transmitted. The moments of the latter distribution are simply $E(B_i)$ and σ_{Bi}. We then solve the expressions for the throughput moments for $E(B_i)$ and σ_{Bi}. We first find $E(B_i)$:

$$\begin{aligned} & E(T_i) = p_i E(B_i) \\ \Rightarrow\ & E(B_i) = E(T_i) / p_i. \end{aligned} \tag{48}$$

Similarly, we find σ_{Bi}:

$$\begin{aligned} & \sigma_{Ti}^2 = p_i \sigma_{Bi}^2 + p_i(1 - p_i)\left(E(B_i)\right)^2 \\ \Rightarrow\ & \sigma_{Bi}^2 = \left(\sigma_{Ti}^2 / p_i\right) - (1 - p_i)\left(E(B_i)\right)^2 \\ \Rightarrow\ & \sigma_{Bi} = \sqrt{\left(\sigma_{Ti}^2 / p_i\right) - (1 - p_i)\left(E(B_i)\right)^2}. \end{aligned} \tag{49}$$

Substituting, we find the desired expression for k_i:

$$k_i = \frac{(1/p_i - 1)E(T_i) + 2 \cdot \sqrt{\left(\sigma_{Ti}^2 / p_i\right) - (1 - p_i)\left(E(T_i)/p_i\right)^2}}{\sigma_{Ti}}. \tag{50}$$

The following table shows the resulting safety factor and recommended capacity for each of the 36 stations in the example model. Note that most of the safety factors are significantly greater than 2.

	Signal Data								Signal Control			
	Antennas		Receivers		Satellites		Base		Satellites		Receivers	
Signal	k_i	C_i	k_i	C_i	k_i	C_i	k_i	C_i	k_i	C_i	k_i	C_i
1	2.48	2.40	3.11	2.98	3.11	2.98	3.11	2.98	4.09	0.85	4.09	0.85
2	1.60	3.10	2.04	3.74	2.04	3.74	2.04	3.74	2.35	1.27	2.35	1.27
3	5.11	7.00	5.75	8.36	5.75	8.36	5.75	8.36	5.32	2.96	5.32	2.96
4	7.96	20.00	8.66	23.42	8.66	23.42	8.66	23.42	8.18	4.19	8.18	4.19
5	5.41	3.00	5.41	3.00	5.41	3.00	5.41	3.00	5.54	1.35	5.54	1.35
6	3.65	9.00	3.82	9.58	3.82	9.58	3.82	9.58	3.90	9.90	3.90	9.90

6.5.2 Commmunications Device Results (Multiple-Station Results)

The above results apply to single stations. However, planners are most interested in determining capacity for communications devices, which comprise multiple stations. Again, the recommended capacity is a function of the expectation and standard deviation of the device's throughput, as follows:

$$C_d = E(T_d) + k_d \cdot \sigma_{Td}, \tag{51}$$

where d is the index of the communications device. Let D be the set of model stations that comprise device d. Due to the linearity of expectation, and the bilinearity property of covariance, we find:

$$\begin{aligned} E(T_d) &= \sum_{i \in D} E(T_i), \\ \sigma_{Td} &= \sqrt{\sum_{i \in D} \sum_{j \in D} \mathrm{cov}(T_i, T_j)}. \end{aligned} \tag{52}$$

The model computes the expectation terms for every station, the variance of every station, and the covariance of every pair of stations. Thus, $E(T_d)$ and σ_{Td} are readily calculated. This leaves the calculation of the safety factor. As before, simply setting k_d to be 2 will be unsatisfactory, as the throughput distribution is not approximately normally distributed.

Unfortunately, determining k will be much more complicated for communications devices than for individual stations, as the device represents the throughput of multiple signals rather than a single signal. Each signal has its own underlying Bernoulli process and performance requirements. The complexity of the situation is compounded by the fact that the joint distribution of signal broadcasts can be quite complicated (since we allow signal broadcasts to be correlated).

We consider two strategies for estimating device capacities.

Exact Strategy. The first strategy assumes perfect knowledge of the joint distribution of the signal broadcasts. It has the following steps:

1. *Calculate the* C_i's, *the capacities required for each individual signal at a particular device when that signal is broadcasting.* We create a separate MR model corresponding to each signal that includes only the stations corresponding to that signal. The model inputs assume that the signal is always broadcasting ($p_i = 1$). The result will be the expectation and variance of the signal's throughput when broadcasting. We thus find each C_i to be $C_i = E(T_i) + 2 \cdot \sigma_{Ti}$.
2. *Use the* C_i's *and the joint distribution of the signal broadcasts to find the* C_d's, *the capacities required at each communications device.* We consider all combinations of signal broadcasts that might flow through a particular device. For each combination, we multiply the probability of the combination times the capacity required to process the combination (sum of the C_i's for the signals being broadcast). This gives us the combination's *mass* (the fraction of all signals flowing through the device that will fall into that particular configuration). We then order all the combinations by the capacity required for each combination. Then, using the mass information, we choose a C_d such that the total fraction of all the signals successfully processed by the device meets a performance requirement for that device. We do this by making C_d large enough to process enough signal combinations to meet the performance requirement.

If very precise capacity estimates are needed, this method may be required. However, the method is complicated, and may not be tractable if many signals flow through a particular device.

Approximate Strategy. This method uses the expectation and standard deviation of the throughput of the device directly to recommend a capacity for the device. As previously stated, however, the difficulty with this method is calculating the device's safety factor, k_d. Here, we approximate k_d to be the weighted sum of the safety factors of the stations comprising the device. The weights are the normalized expected throughputs at each of the stations. Specifically, we approximate k_d to be:

$$k_d = \frac{\sum_{i \in D} k_i E[T_i]}{\sum_{i \in D} E[T_i]}. \tag{53}$$

The justification for this safety factor: at any time, the bandwidth demand on the device is some combination of demands required by separate signals. Each signal's demand requires a certain safety factor, k_i. The probability that a certain amount of demand is due to a particular signal is the expected bandwidth of that signal divided by the total expected bandwidth. Thus, the expected safety factor required for any particular unit of demand is given by the above equation for k_d.

The following table shows the safety factor, k_d, for each communications device in the example model.

Communications Device	Signals Processed	Weighted k-factor	Recommended
Antenna 1 (a1)	1, data	2.48	2.40
Antenna 2 (a2)	2, data	1.60	3.10
Antenna 3 (a3)	3, data	5.11	7.00
Antenna 4 (a4)	4, data	7.96	20.00
Antenna 5 (a5)	5, data	5.41	3.00
Antenna 6 (a6)	6, data	3.65	9.00
Receiver 1 (r1)	1, 2, 3, data and control	3.26	13.27
Receiver 2 (r2)	4, data and control	8.58	27.26
Receiver 3 (r3)	5,6, data and control	4.01	21.20
Satellite 1 (s1)	1, 2, 3, data and control	3.26	13.27
Satellite 2 (s2)	4, 5, 6, data and control	4.99	31.05
Base (b)	All, data	4.28	22.40
			To process signal 4: 23.42

As with any approximation based on expected values, care must be taken that the resulting recommended capacities actually meet system requirements. For example, consider the recommended capacity for the Base; it is less than the capacity required to process signal 4 by itself. This recommendation is not surprising, since signal 4 broadcasts infrequently in comparison to the other signals. If processing signal 4 is a requirement, however, the recommended capacity at the base station must be at least that required to process signal 4.

Chapter 3: Models with General Control Rules (Class GLS-MR)

1. Introduction

In Chapter 2, we considered LLS-MR models, which used linear control rules. These control rules were either single-queue rules (form $P_{it} = \alpha_i Q_{it}$, where α_i is a constant) or multi-queue rules (form $P_{it} = \sum_j \alpha_{ij} Q_{jt}$, where the α_{ij}'s are constants). In this chapter, we consider GLS-MR models. We recall the taxonomy of MR models in Chapter 1 to identify the properties of these models:

- *General control rules.* Here, the control rules are general functions of the queue lengths. We will consider single-queue control rules of the form $P_{it} = p_i(Q_{it})$, and multi-queue control rules of the form $P_{it} = p_i(\mathbf{Q}_t)$.
- *Linear inter-station work transfers.* Work arrivals from upstream stations are linear functions of the production at upstream stations plus a random noise term.
- *Stationary and independent exogenous arrivals.* Work arrivals from outside the network come from stationary and independent distributions. Combined with linear work arrivals from upstream stations, we see that GLS-MR models use the arrival process discussed in chapter 2. Work arrivals are defined by $\mathbf{A}_t = \mathbf{\Phi P}_{t-1} + \boldsymbol{\varepsilon}_t$, where Φ is a constant matrix and ε_t is a random noise vector.

The defining feature of GLS-MR models is the fact that the control rules are general functions of the queue lengths. Unlike linear control rules, we will not be able to find the first two moments of the production quantities and queue lengths exactly; instead, we will develop approximations of them using Taylor-series expansions.

Throughout this chapter, we make the following assumptions about the control rules and other model dynamics:

- The production functions, $p_i(\mathbf{Q}_t)$, are continuous, twice differentiable, and monotonic over the ranges of production and queue length quantities seen by the stations in the job shop. Mathematically, these conditions hold for $\mathbf{P}_{\min} \le \mathbf{P}_t \le \mathbf{P}_{\max}$ and $\mathbf{Q}_{\min} \le \mathbf{Q}_t \le \mathbf{Q}_{\max}$, where P_{min} and P_{max} are vectors denoting the minimum and maximum production quantities, and Q_{min} and Q_{max} are vectors denoting minimum and maximum queue lengths.
- The production functions have inverse functions, $q_i(\mathbf{P}_t)$, that are continuous and twice differentiable over the ranges of possible production and queue length quantities. (Note that the conditions on the production functions imply the existence of the inverse production functions.)

We do not require that these assumptions be met outside the ranges of possible production and queue lengths. Thus, we can analyze control rules that violate the assumptions outside the ranges. For example,

we will consider control rules that become infinitely negative as Q_{it} approaches some negative value; this is of no concern since Q_{it} is assumed to be non-negative.

Exploring models with general control rules has practical value. Below, we present some applications of models for networks using general control rules.

Manufacturing. A conventional approach to job-shop capacity planning is to assume that the amount of work a station produces per period is constrained by a fixed capacity limit. This production function (called the bounded control rule) has the following form:

$$p_i(Q_{it}) = \begin{cases} Q_{it}, Q_{it} < M_i, \\ M_i, Q_{it} \geq M_i, \end{cases} \tag{1}$$

where M_i is the maximum production level of station i. The capacity level has no impact on the production quantities, unless the maximum production level is reached. Examples of this approach include Hax (1978), Lin (1986), Baker (1993) and Bitran and Tirupati (1993).

In practice, the bounded control rule often does not hold exactly. Instead, machines often show *saturation* or *congestion* effects. With these effects, machines increase their service rate somewhat to account for increased work-in-queue; for example, operators will try to push more work through a machine in a given period. However, the rate of production increases diminishes with increased work-in-queue, due to machine clogging, increased probability of quality problems, and so on.

Karmarkar (1989, 1993) discussed production functions called *clearing functions* that account for saturation and congestion. The basic form of a clearing function is:

$$p_i(Q_i) = \frac{M_i Q_i}{\beta_i + Q_i} \tag{2}$$

Here, M_i is an upper bound on the service rate, and β_i controls the rate at which the production approaches the upper bound. The parameter M_i is the maximum production, asymptotically approached as Q_i approaches infinity. The parameter β_i determines the rate at which the production approaches M_i. Clearing functions are continuous and twice-differentiable, making them amenable to the modeling approach we develop in this chapter.

Karmarkar uses queuing theory to justify clearing functions. Consider an M/M/1 queue. Let λ_i be the arrival rate at station i, and let μ_i be the service rate. The average waiting time in an M/M/1 queue, $\overline{W_i}$, is given by:

$$\overline{W_i} = \frac{1/\mu_i}{1-(\lambda_i/\mu_i)}, \tag{3}$$

Using Little's Law, we have that $\overline{W_i} = \overline{Q_i} / \lambda_i$, where $\overline{Q_i}$ is the expected queue length. Substituting for $\overline{W_i}$ in (3), and solving the resulting equation for λ_i yields:

$$\lambda_i = \frac{\mu_i \overline{Q_i}}{1 + \overline{Q_i}}. \tag{4}$$

More generally, for any queuing model in which the waiting time is proportional to $1/(1-\lambda_i/\mu_i)$, (such as an M/G/1 queue, for example), we find that the expected arrival rate will be proportional to $\mu_i \overline{Q_i}/(1+\overline{Q_i})$. If the queue is stable, the expected arrival rate equals the expected rate of production. Thus, we have that the expected rate of production is proportional to $M_i \overline{Q_i}/(1+\overline{Q_i})$ as well.

Now consider a discretized version of (4). Rather than have an expected production rate (which equals the expected arrival rate), we have an expected quantity produced each time period, and an expected queue length at the start of each time period. The resulting discretized version of (4) is:

$$E(P_i) = \frac{M_i E(Q_i)}{1 + E(Q_i)} \tag{5}$$

We now use (5) to suggest a control rule. We change (5) from a relationship between production and queue length moments to a relationship between the actual production and queue length quantities. The resulting control rule is:

$$P_{it} = \frac{M_i Q_{it}}{1 + Q_{it}}, \tag{6}$$

which has the form of a clearing function. Indeed, clearing functions are generalizations of (6), with 1 replaced by β_i.

Thus, queuing models, commonly used to model manufacturing networks (reviewed, for instance, by Buzacott and Yao (1986) and Suri and Sanders (1993)) suggest the use of clearing functions in discrete-period networks.

Staffing in Manufacturing. Graves (1988) considers the staffing levels in a repair depot, and suggests the following piecewise linear production function:

$$P_{it} = \min[Q_{it}, K + Q_{it}/\beta], \tag{7}$$

where K and β are constants. At low work-in-queue levels, employees simply complete the work in the queue within a period. At higher work-in-queue levels, employees complete work according to a linear control rule. The rule at higher levels might correspond to overtime situations, for example, in which employees can only complete a portion of the additional work.

This piecewise linear production function cannot be evaluated directly; as a result, Graves suggests using the approximation $P_{it} = K + Q_{it} / \beta$, assuming that the work-in-queue will almost always be greater than the quantity dictating that $P_{it} = K + Q_{it} / \beta$. (For reference, this quantity is $Q_{it} = K - K / \beta$.) This assumption will not be valid, however, for shops that regularly oscillate between comparatively low and high demands. In such an environment, a concave production function, such as a clearing function, may be a better approximation than a linear control rule. Indeed, an analysis of the work at an actual shop may show that production quantities are better explained by a concave production function than a piecewise linear function, especially if productivity returns diminish with increasing overtime (see next example).

Staffing in General. Duncan and Nevison (1996) present empirical research that compares real productivity against hours worked. They found that the resulting curve was a concave function; for example, working 60 hours yielded, on average only 50 hours of real work, and working 70 hours yielded little more. (This result applies to a single week of overtime; after 10 weeks, working 70 hours yielded only 30 hours of real work.)

Section 2 presents the basic methodology used to approximate the production and queue length moments. Section 3 develops algorithms that approximate the steady-state production and queue length moments for single-queue control rules. Section 4 analyzes the empirical behavior of the approximation algorithms on a single station, a six-station assembly line, and a thirteen-station job shop that manufactures mainframe subcomponents, originally considered by Fine and Graves (1989). Section 5 extends the algorithms to multi-queue control rules. Finally, Section 6 derives a strict lower bound on expected queue lengths for concave production functions.

2. The Delta Method for Production and Queue Length Moments

We assume that the control rules, $P_{it} = p_i(Q_{it})$, are continuous functions that are twice differentiable with respect to Q_{it}. Then we can use the *Delta Method* (c.f. Rice, 1995, pp. 149-154) to find approximations for E(Q_i) and var(Q_i). In this section, we discuss single-queue control rules; production functions of multiple queue lengths are discussed in Section 5.

2.1 Method Development

The Delta Method approximates a function of a random variable with Taylor series expansions of that function around the mean of the random variable. Suppose that:

$$Y = g(X), \tag{8}$$

where X is a random variable with mean μ_X and variance σ_X^2. Then the first-order Taylor expansion of Y is:

$$Y \approx g(\mu_X) + (X - \mu_X)g'(\mu_x), \tag{9}$$

and the second-order expansion of Y is:

$$Y \approx g(\mu_X) + (X - \mu_X)g'(\mu_x) + \frac{1}{2}(X - \mu_X)^2 g''(\mu_X). \tag{10}$$

Using the second-order Taylor expansion of Y, a second-order aproximation of E(Y) is:

$$E(Y) \approx g(\mu_X) + \frac{1}{2}\sigma_X^2 g''(\mu_X). \tag{11}$$

Taking the variance of the first-order Taylor expansion of Y, we find that a first-order approximation of Var(Y) is:

$$Var(Y) \approx \sigma_X^2 [g'(\mu_X)]^2. \tag{12}$$

Note that we use only the first-order expansion, since using the second-order expansion would require the computation of fourth moments.

We now use the Delta Method to develop two sets of approximate recursion equations for the GLS-MR model. In the first set, the recursion equations are in terms of queue lengths. This set gives us recursion equations for the expectations and variances of the queue lengths. In the second set, the recursion equations are in terms of the production quantities. This set gives us recursion equations for the expectations and variances of the production quantities. As we will see, both sets will be useful in analyzing GLS-MR models.

2.2 Recursion Equations for the Queue Lengths

Consider the recursion equation for the LLS-MR model:

$$\mathbf{Q}_t = \mathbf{Q}_{t-1} + (\mathbf{\Phi} - \mathbf{I})\mathbf{D}\mathbf{Q}_{t-1} + \boldsymbol{\varepsilon}_t. \tag{13}$$

Let $\mathbf{P}_t = \mathbf{p}(\mathbf{Q}_{t-1})$ rather than $\mathbf{D}\mathbf{Q}_{t-1}$, where $\mathbf{p}(\mathbf{Q}_{t-1})$ is a continuous and twice differentiable vector-valued function of $\mathbf{Q}_{t-1}$. Assume that the control rule for each station depends only on the work in queue at that station; or, mathematically, $p_i(\mathbf{Q}_{t-1})$ depends only on $Q_{i,t-1}$. This yields a new recursion equation for the GLS-MR model:

$$\mathbf{Q}_t = \mathbf{Q}_{t-1} + (\mathbf{\Phi} - \mathbf{I})\mathbf{p}(\mathbf{Q}_{t-1}) + \boldsymbol{\varepsilon}_t. \tag{14}$$

The first-order expansion of the recursion equation is:

$$\mathbf{Q}_t \approx \mathbf{Q}_{t-1} + (\mathbf{\Phi} - \mathbf{I})\mathbf{p}(\overline{\mathbf{Q}}_{t-1}) + (\mathbf{\Phi} - \mathbf{I}) \cdot \mathbf{\Pi}_{t-1} \cdot [\mathbf{Q}_{t-1} - \overline{\mathbf{Q}}_{t-1}] + \boldsymbol{\varepsilon}_t, \tag{15}$$

and the second-order expansion of the recursion equation is:

$$\mathbf{Q}_t \approx \mathbf{Q}_{t-1} + (\mathbf{\Phi} - \mathbf{I})\mathbf{p}(\overline{\mathbf{Q}}_{t-1}) + (\mathbf{\Phi} - \mathbf{I}) \cdot \mathbf{\Pi}_{t-1} \cdot [\mathbf{Q}_{t-1} - \overline{\mathbf{Q}}_{t-1}] + \frac{1}{2}(\mathbf{\Phi} - \mathbf{I}) \cdot \mathbf{\Pi}_{2,t-1} \cdot [\mathbf{Q}_{t-1}^{m2}] + \boldsymbol{\varepsilon}_t, \tag{16}$$

where:

- $\overline{\mathbf{Q}}_{t-1}$ is the vector of expected queue levels at time *t-1*;
- $\mathbf{\Pi}_{t-1}$ is a diagonal matrix whose *ii*'th element is $p_i'(\overline{Q}_{i,t-1})$;
- $\mathbf{\Pi}_{2,t-1}$ is a diagonal matrix whose *ii*'th element is $p_i''(\overline{Q}_{i,t-1})$; and
- $\mathbf{Q}_{t-1}^{m2}$ is a vector whose *i*th element is $(Q_{i,t-1} - \overline{Q}_{i,t-1})^2$.

Then the expectation of the second-order approximation is:

$$\overline{\mathbf{Q}}_t \approx \overline{\mathbf{Q}}_{t-1} + (\mathbf{\Phi} - \mathbf{I}) \cdot \mathbf{p}(\overline{\mathbf{Q}}_{t-1}) + \frac{1}{2}(\mathbf{\Phi} - \mathbf{I}) \cdot \mathbf{\Pi}_{2,t-1} \cdot [\mathbf{Q}_{t-1}^{\sigma}] + \boldsymbol{\mu}_t, \tag{17}$$

where $\mathbf{Q}_{t-1}^{\sigma}$ is a vector such that $Q_{i,t-1}^{\sigma} = \text{var}(Q_{t-1})$. (Since $\mathbf{Q}_{t-1}^{\sigma}$ will not be known exactly, we will use the first-order approximation for $\mathbf{Q}_{t-1}^{\sigma}$ described below).

The variance of the first-order approximation is:

$$\begin{gathered} \text{var}(\mathbf{Q}_t) \approx \mathbf{B} \cdot \text{var}(\mathbf{Q}_{t-1}) \cdot \mathbf{B}^T + \Sigma, \\ \text{where } \mathbf{B} = \mathbf{I} + (\mathbf{\Phi} - \mathbf{I})\Pi_{t-1} \end{gathered} \tag{18}$$

Equations (17) and (18) are approximate recursion equations for the moments of the queue lengths, as desired.

2.3 Recursion Equations for the Production Quantities

Consider the basic recursion equation for the GLS-MR model, $\mathbf{Q}_t = \mathbf{Q}_{t-1} + (\mathbf{\Phi} - \mathbf{I})\mathbf{p}(\mathbf{Q}_{t-1}) + \boldsymbol{\varepsilon}_t$. We rewrite this equation in terms of production quantities, using the relationship $\mathbf{Q}_t = \mathbf{q}(\mathbf{P}_t)$:

$$\mathbf{q}(\mathbf{P}_t) = \mathbf{q}(\mathbf{P}_{t-1}) + (\mathbf{\Phi} - \mathbf{I})\mathbf{P}_{t-1} + \boldsymbol{\varepsilon}_t, \tag{19}$$

where $\mathbf{q}(\mathbf{P}_t)$ is the continuous and twice-differentiable function of $\mathbf{P}_t$ that is the inverse of $\mathbf{p}(\mathbf{Q}_t)$. A first-order expansion of (19) around E(**P**) is:

$$\mathbf{q}(\overline{\mathbf{P}}_t) + \mathbf{\Psi}_t \cdot (\mathbf{P}_t - \overline{\mathbf{P}}_t) \approx \mathbf{q}(\overline{\mathbf{P}}_{t-1}) + \mathbf{\Psi}_{t-1} \cdot (\mathbf{P}_{t-1} - \overline{\mathbf{P}}_{t-1}) + (\mathbf{\Phi} - \mathbf{I})\mathbf{P}_{t-1} + \boldsymbol{\varepsilon}_t, \tag{20}$$

and the second order expansion is:

$$\mathbf{q}(\overline{\mathbf{P}}_t)+\boldsymbol{\Psi}_t\cdot(\mathbf{P}_t-\overline{\mathbf{P}})+\frac{1}{2}\boldsymbol{\Psi}_{2,t}\cdot\left[\mathbf{P}_t^{m2}\right]\approx\mathbf{q}(\overline{\mathbf{P}}_{t-1})+\boldsymbol{\Psi}_{t-1}\cdot(\mathbf{P}_{t-1}-\overline{\mathbf{P}}_{t-1})+\frac{1}{2}\boldsymbol{\Psi}_{2,t-1}\cdot\left[\mathbf{P}_{t-1}^{m2}\right]+(\boldsymbol{\Phi}-\mathbf{I})\mathbf{P}_{t-1}+\boldsymbol{\varepsilon}_t,\quad(21)$$

where:

- $\overline{\mathbf{P}}_t$ is the vector of expected queue levels at time t;
- $\boldsymbol{\Psi}_t$ is a diagonal matrix whose ii'th element is $q_i'(\overline{P}_{it})$;
- $\boldsymbol{\Psi}_{2,t}$ is a diagonal matrix whose ii'th element is $q_i''(\overline{P}_{it})$; and
- $\mathbf{P}_t^{m2}$ is a vector whose ith element is $(P_{i,t}-\overline{P}_{it})^2$.

Then the expectation of the second-order approximation is:

$$\mathbf{q}(\overline{\mathbf{P}}_t)+\frac{1}{2}\boldsymbol{\Psi}_{2,t}\cdot\left[\mathbf{P}_t^{\sigma}\right]\approx\mathbf{q}(\overline{\mathbf{P}}_{t-1})+\frac{1}{2}\boldsymbol{\Psi}_{2,t-1}\cdot\left[\mathbf{P}_{t-1}^{\sigma}\right]+(\boldsymbol{\Phi}-\mathbf{I})\overline{\mathbf{P}}_{t-1}+\boldsymbol{\mu}_t,\quad(22)$$

where $\mathbf{P}_t^{\sigma}$ is a vector such that $P_{it}^{\sigma}=\mathrm{var}(P_{it})$. (Since $\mathbf{P}_t^{\sigma}$ will not be known exactly, we will use the first-order approximation for $\mathbf{P}_t^{\sigma}$ described below).

The variance of the first-order approximation is:

$$\boldsymbol{\Psi}_t\cdot\mathbf{var}(\mathbf{P}_t)\cdot\boldsymbol{\Psi}_t^{\mathsf{T}}=(\boldsymbol{\Phi}-\mathbf{I}+\boldsymbol{\Psi}_{t-1})\cdot\mathbf{var}(\mathbf{P}_{t-1})\cdot(\boldsymbol{\Phi}-\mathbf{I}+\boldsymbol{\Psi}_{t-1})^T+\boldsymbol{\Sigma}.\quad(23)$$

Equations (22) and (23) are approximate recursion equations for the production moments, as desired.

2.4 Using the Recursion Equations for Transitive Analysis

One use of the recursion equations is transitive analysis. Suppose we begin at time 0 with estimates of the expectation and variances of the production quantities and queue lengths. Then we repeatedly apply equations (17) and (18) to find the approximate expectations and variances in periods 1, 2, 3... t. (Note that equation (17) needs the queue length variances to calculate the next expected queue length vector; we use the approximate variances calculated by equation (18).)

However, it is not simple to use (22) and (23) directly for transitive analysis. These equations do not give explicit formulas for the production moments. Indeed, (22) and (23) create simultaneous nonlinear equations for E(P_{it}) and var(P_{it}) at each station i. As a simpler approach, we can apply the Delta Method to $p_i(Q_{it})$ for each period t. Doing so yields:

$$E(P_{it})\approx p_i(\overline{Q}_{it})+\frac{1}{2}p_i''(\overline{Q}_{it})\cdot\mathrm{var}(Q_{it}),\text{ and}\quad(24)$$

$$\text{var}(P_{it}) \approx \left(p_i'(\bar{Q}_{it})\right)^2 \text{var}(Q_{it}) \tag{25}$$

The drawback of (24) and (25) is that the estimated production moments are "approximations of approximations." While this is unlikely to be a major problem for $E(P_{it})$, which is a second-order approximation, the first-order estimates of $\text{var}(P_{it})$ may err significantly.

Of course, the above discussion highlights a major concern with the recursion equations: each successive set of estimates comprises approximation of approximations, so that the estimation errors may grow with the number of periods. Thus, in practice, we would only use the recursion equations to "look ahead" a few periods.

The same methodology applies to the transitive analysis of a network starting in a fixed state, with initial inventory levels $\mathbf{Q_0}$. We simply apply the methodology with $\mathbf{E(Q_0) = Q_0}$, $\mathbf{E(P_0) = p(Q_0)}$, and $\mathbf{var(Q_0) = var(P_0) = 0}$.

We are interested in steady-state analysis, as well. Steady-state analysis is the topic of Section 3, where we use the recursion equations to derive expressions for the steady-state production and queue length moments.

3. Steady-State Analysis for Models with Single-Queue Control Rules

In this section, we derive approximations of the steady-state moments, E(**Q**), E(**P**), var(**Q**), and var(**P**) for networks with single-queue production functions. To do so, we assume that the networks do have steady-state moments. Thus, as necessary conditions, we assume that the networks are aperiodic and ergodic.

3.1 Equations for the Steady-State Moments

We could derive approximations for the steady-state moments by repeatedly iterating the recursion equations from Section 2, starting from $E(\mathbf{Q_0})$ and $\text{var}(\mathbf{Q_0})$, and wait for the estimates to converge. However, as highlighted at the end of Section 2, this approach has several problems.

- *Accuracy.* Note that the model "converges" to estimates of E(**Q**) and var(**Q**) by taking approximations of approximations each iteration. The method raises questions of how accurate the steady-state approximations are, or even if the resulting approximations converge.
- *Convergence speed.* There is no immediately obvious way to accelerate the recursion-equation calculations, since the control function results and derivatives have to be recalculated after every iteration.

Consequently, we will focus on a different approach: finding fixed-point solutions for the recursion equations. Clearly, the estimates for the moments will be fixed-point solutions to the recursion equations (although the reverse need not be true).

The queue-length recursion equations, (17) and (18) provide the following system of equations for E(**Q**) and var(**Q**):

$$\begin{aligned} (\Phi - \mathbf{I})\mathbf{p}(\overline{\mathbf{Q}}) + \frac{1}{2}(\Phi - \mathbf{I}) \cdot \Pi_2 \cdot \left[\mathbf{Q}^{\sigma}\right] + \mu &= \mathbf{0} \\ \left[\mathbf{I} + (\Phi - \mathbf{I})\mathbf{p}(\overline{\mathbf{Q}})\right] \cdot \mathrm{var}(\mathbf{Q}) \cdot \left[\mathbf{I} + (\Phi - \mathbf{I}) \cdot \Pi\right]^T - \mathrm{var}(\mathbf{Q}) + \Sigma &= \mathbf{0} \end{aligned} \tag{26}$$

Here, Π is a diagonal matrix whose *ii*'th element is $p_i'(\overline{Q})_i$ and Π_2 is a diagonal matrix whose *ii*'th element is $p_i''(\overline{Q})_i$. Note that the first equation in (26) contains fewer terms than (17), since $\overline{\mathbf{Q}}_t = \overline{\mathbf{Q}}_{t-1} = E(\mathbf{Q})$ in steady state.

Similarly, the production recursion equations, (22) and (23) provide the following system of equations for E(**P**) and var(**P**):

$$\begin{aligned} (\mathbf{\Phi} - \mathbf{I})E(\mathbf{P}) + \boldsymbol{\mu} &= \mathbf{0} \\ (\mathbf{\Phi} - \mathbf{I} + \mathbf{\Psi}) \cdot \mathbf{var(P)} \cdot (\mathbf{\Phi} - \mathbf{I} + \mathbf{\Psi})^T - \mathbf{\Psi} \cdot \mathbf{var(P)} \cdot \mathbf{\Psi}^{\mathbf{T}} + \mathbf{\Sigma} &= \mathbf{0} \end{aligned} \tag{27}$$

Here, $\mathbf{\Psi}$ is a diagonal matrix whose *ii*'th element is $q_i'(\overline{P}_i)$, and $\mathbf{\Psi}_2$ is a diagonal matrix whose *ii*'th element is $q_i''(\overline{P}_i)$. The first equation of (27) is much simpler than (22), since

$$\mathbf{q}(\overline{\mathbf{P}}_t) + \frac{1}{2}\mathbf{\Psi}_{2,t} \cdot \left[\mathbf{P}_t^{\sigma}\right] = \mathbf{q}(\overline{\mathbf{P}}_{t-1}) + \frac{1}{2}\mathbf{\Psi}_{2,t-1} \cdot \left[\mathbf{P}_t^{\sigma}\right] \text{ in steady state.} \tag{28}$$

To calculate the production and queue length moments, we could solve both sets of equations. Alternately, we can solve system (26) and approximate the production moments by applying the Delta Method to the queue length moments or solve system (27) and approximate the queue length moments by applying the Delta Method to the production moments.

Following term-by-term multiplication for either set of equations, we find that there are approximately $n + n^2/2$ equations for $n + n^2/2$ variables for each set (the expectations and the distinct covariance terms). Trying to solve a system this large by brute force could be quite difficult; indeed, iterating the recursion equations might produce faster estimates. Further, with systems this large and complicated, there is no guarantee that the returned solution would comprise the true moment approximations.

Instead, we take advantage of the structure of the systems, which yield approaches that make finding the fixed-point solutions fairly simple. The key insight is that the expected production can be

found analytically, regardless of the control rules. Note that the steady-state equation for the expected production in (27) is extremely simple – it implies that $E(\mathbf{P}) \approx (\mathbf{I} - \mathbf{\Phi})^{-1}\boldsymbol{\mu}$. Proposition 1 shows that this expression is exact.

Proposition 1. Suppose we have an ergodic and aperiodic network, and that all the network's stations are stable (i.e., stations eventually process all the work they receive). Then the steady-state expected production vector is $\overline{\mathbf{P}} = (\mathbf{I} - \Phi)^{-1}\boldsymbol{\mu}$, regardless of the form of the control rule.

Proof. Assume that a steady-state production vector, $\overline{\mathbf{P}}$, exists. Then the vector of expected arrivals equals $\mathbf{\Phi}\overline{\mathbf{P}} + \boldsymbol{\mu}$, since work arrivals are defined by $\mathbf{A}_t = \mathbf{\Phi}\mathbf{P}_{t-1} + \boldsymbol{\varepsilon}_t$. But then, $\overline{\mathbf{P}}$ must equal the expected arrivals, or else $Q_t \to \pm\infty$. Solving $\overline{\mathbf{P}} = \mathbf{\Phi}\overline{\mathbf{P}} + \boldsymbol{\mu}$ for $\overline{\mathbf{P}}$ yields $\overline{\mathbf{P}} = (\mathbf{I} - \Phi)^{-1}\boldsymbol{\mu}$. □

Proposition 1 is a powerful result; it guarantees that we can find the expected production at every station in a job shop regardless of the control rule used (assuming the control rule used results in the shop being stable). We will see that knowing the expected production quantities greatly simplifies the calculation of the expected queue lengths.

If we can find an estimate for E(**Q**), knowing E(**P**) allows the computation of the expected waiting times. Little's Law says that the expected queue length at a station equals the arrival rate at the station times the expected waiting time. In steady-state, however, the arrival rate must equal the expected production for the system to be stable. Then, once we obtain an estimate for a station's expected queue length, an estimate for the waiting time is just the expected queue length divided by the expected production.

Below, we use E(**P**) to create two algorithms to estimate the steady-state moments. The first algorithm solves the queue-length recursion equations, system (26). The second solves the production recursion equations, system (27).

3.2 Estimating the Steady-State Moments Using Queue Lengths

In this section, we develop an iterative algorithm to solve the queue-length recursion equations. We begin by using E(**P**) to find a first-order approximation for E(**Q**). Recall that the first-order approximation for E(P_i), given E(Q_i), is $p_i(\overline{Q}_i)$. Since we know E(P_i) from Proposition 1, a first-order approximation for E(Q_i) is the solution to the equation $p_i(\overline{Q}_i) = \overline{P}_i$. (Alternately, if the inverse production function, $q_i(P)$, is known analytically, a first-order approximation for E(Q_i) is $\overline{Q}_i \approx q_i(\overline{P}_i)$.)

We next find a first-order estimate for var(**Q**) given an estimate for E(**Q**). Consider the first order recursion equation for the variance, $\text{var}(\mathbf{Q}_t) \approx \mathbf{B} \cdot \text{var}(\mathbf{Q}_{t-1}) \cdot \mathbf{B}' + \Sigma$, where $\mathbf{B} = \mathbf{I} + (\Phi - \mathbf{I}) \cdot \Pi_{t-1}$. In steady-state, and given an estimate for E(**Q**), this equation becomes

$$\mathbf{var(Q)} \approx \mathbf{B} \cdot \mathbf{var(Q)} \cdot \mathbf{B}^{\mathrm{T}} + \Sigma, \text{ where } \mathbf{B} = \mathbf{I} + (\Phi - \mathbf{I}) \cdot \Pi, \tag{29}$$

and where $\mathbf{\Pi}$ is a diagonal matrix whose *ii*'th element is $p_i'(\overline{\mathbf{Q}})$. But then, the resulting **B** is a constant matrix, as with the linear control rule models discussed in Chapter 2. Then we find the steady-state solution to the first-order recursion equation by iterating the recursion equation infinitely, as with the linear control rule models. The result is following power series:

$$\begin{gathered} \text{var}(\mathbf{Q}) \approx \sum_{s=0}^{\infty} \mathbf{B}^s \Sigma \mathbf{B}'^s, \\ \text{where } \mathbf{B} = \mathbf{I} + (\Phi - \mathbf{I}) \cdot \Pi. \end{gathered} \tag{30}$$

The power series may be evaluated numerically or analytically using the techniques discussed in Chapter 2. Indeed, the power series expression for var(**Q**) is the same as that for linear control rules, with the smoothing matrix **D** replaced by the diagonal matrix Π. Thus, the first-order approximation maps a linear control rule onto the general control rule, with the smoothing factors being $p_i'(\overline{Q}_i)$ rather than α_i. The resulting "lead times" of the fitted TPM model are given by $L_i = 1/p_i'(\overline{Q}_i)$. Consequently, the linear approximation of var(**Q**) is heavily dependent on the first derivative of the control rule at the expected queue length. In particular, the approximation requires the first derivative to be between zero and one. If the first derivative is more than one, the approximation will not converge, since in the corresponding linear model the station is processing more than its queue each time period. First derivatives approaching zero correspond to having longer and longer lead times, which smooth production but increase the queue lengths – and the variances of the queue lengths.

We now find a second-order estimate for E(**Q**) given an estimate for var(**Q**). Recall that the second-order approximation for E(P_i), given E(Q_i), is $p_i(\overline{Q}_i) + \frac{1}{2} p_i''(\overline{Q}_i) \cdot \text{var}(Q_i)$. In this case, we know E(P_i), so a second-order approximation for E(Q_i) is the solution to the equation $p_i(\overline{Q}_i) + \frac{1}{2} p_i''(\overline{Q}_i) \cdot \text{var}(Q_i) = \overline{P}_i$.

The above discussion suggests an iterative algorithm to solve system (26), yielding the desired estimates for E(**Q**) and var(**Q**).

Algorithm MomentsQ

1. **Initialization**. Calculate the first-order approximation of E(**Q**) by solving $p_i(\overline{Q}_i) = \overline{P}_i$ for $\overline{Q}_i$. Use this E(**Q**) to calculate a first-order approximation of var(**Q**), using the formula $\mathbf{var(Q)} \approx \sum_{s=0}^{\infty} \mathbf{B}^s \Sigma \mathbf{B}'^s$, where $\mathbf{B} = \mathbf{I} + (\Phi - \mathbf{I}) \cdot \Pi$, and $\mathbf{\Pi}$ is a diagonal matrix whose ii'th element is $p_i'(\overline{\mathbf{Q}})$.
2. **Iteration**. Use the previous estimate of var(**Q**) to calculate a new second-order estimate of E(**Q**) by solving the equation $p_i(\overline{Q}_i) + \frac{1}{2} p_i''(\overline{Q}_i) \cdot \mathrm{var}(Q_i) = \overline{P}_i$ for $\overline{Q}_i$. Use the new E(**Q**) to calculate a new first-order estimate of var(**Q**).
3. **Convergence**. Repeat the iterative step until the estimates of E(**Q**) and var(**Q**) converge to within desired limits.

MomentsQ is not guaranteed to converge for all possible control rules. The analyst would need to examine specific control rules to determine whether MomentsQ would converge. However, we expect that MomentsQ usually will converge for the family of control rules considered in this chapter (monotonic, concave, and twice-differentiable).

Also, the above algorithm does not generate an estimate of var(**P**). (Recall that E(**P**) is found exactly using Proposition 1.) We can find an estimate of var(**P**) using the first-order estimate of var(**Q**). Since **P** = **p**(**Q**), a first-order estimate of var(P) is:

$$\mathbf{var(P)} \approx \mathbf{\Pi} \cdot \mathbf{var(Q)} \cdot \mathbf{\Pi}^{\mathrm{T}}. \tag{31}$$

The drawback of this procedure is that it creates a first-order approximation of a first-order approximation; the resulting var(**P**) may be inaccurate. (In Section 4, however, we will see that there are certain conditions under which this approximation is quite good.)

Finally, MomentsQ has the drawback that it is an iterative algorithm. In addition to convergence uncertainty, it will be slower than an analytic algorithm. In the next section, we will see that the algorithm based on the production recursion equations is analytic.

Nonetheless, MomentsQ does have certain advantages. As discussed, the queue-length recursion equations are much easier to work with when doing transitive analysis. Further, MomentsQ works particularly well on an important class of multi-queue control rules. (Multi-queue control rules are discussed in Section 5).

Example. Suppose that the stations in a job shop use clearing functions of the form:

$$p_i(Q_i) = \frac{M_i Q_i}{\beta_i + Q_i}. \tag{32}$$

Here, M_i is an upper bound on the service rate, and β_i controls the rate at which the production approaches the upper bound.

Given E(P_i), the equation for the first-order approximation of E(Q_i) is:

$$\frac{M_i \overline{Q}_i}{\beta_i + \overline{Q}_i} = \overline{P}_i, \tag{33}$$

which is a linear equation with the unique solution:

$$\begin{aligned} \overline{Q}_i &= \frac{\beta_i \overline{P}_i}{M_i - \overline{P}_i} = \frac{\beta_i \rho_i}{1 - \rho_i}, \\ \rho_i &= \overline{P}_i / M_i. \end{aligned} \tag{34}$$

This expression is quite similar to common queuing theory expressions for expected queue lengths, in that the expected queue length is proportional to the inverse of the service utilization, $\rho_i = \overline{P}_i / M_i$. In addition, the expected queue length is proportional to the rate at which the production approaches the maximum service rate.

To estimate the first-order variances, we need to find the entries of Π given an estimate for E(**Q**). Recall that $\Pi_{ii} = p_i'(\overline{Q}_i)$. Applying this formula to the clearing function yields:

$$\Pi_{ii} = \frac{M_i \beta_i}{(\beta_i + \overline{Q}_i)^2}, \tag{35}$$

so we immediately use formula (30) to estimate var(**Q**).

To generate second-order estimates for E(**Q**), we need to solve $p_i(\overline{Q}_i) + \frac{1}{2} p_i''(\overline{Q}_i)\sigma_{Qi}^2 = \overline{P}_i$ for each E(Q_i). With clearing functions, we need to solve the following equation for E(Q_i):

$$\frac{M_i \overline{Q}_i}{\beta_i + \overline{Q}_i} - \frac{M_i \beta_i}{(\overline{Q}_i + \beta_i)^3} \cdot \text{var}(\overline{Q}_j) = \overline{P}_i. \tag{36}$$

Equation (36) is not easy to solve analytically, since it is a cubic equation. However, it can be solved using standard numerical techniques for one-variable equations.

With all the formulas defined, MomentsQ may now be applied to the shop. We present MomentQ results for job shops using clearing functions in Section 4.

3.3 Estimating the Steady-State Moments Using Production Quantities

In this section, we develop an analytic algorithm to solve the production recursion equations, and use the resulting moments to estimate the moments of the queue lengths analytically.

We begin by calculating E(**P**) exactly. Recall that Proposition 1 tells us that $\overline{\mathbf{P}} = (\mathbf{I} - \Phi)^{-1}\mu$.

We next consider var(**P**). Recall that the first-order expansion of the production recursion equation was:

$$\mathbf{q}(\overline{\mathbf{P}}_t) + \Psi_t \cdot (\mathbf{P}_t - \overline{\mathbf{P}}_t) \approx \mathbf{q}(\overline{\mathbf{P}}_{t-1}) + \Psi_{t-1} \cdot (\mathbf{P}_{t-1} - \overline{\mathbf{P}}_{t-1}) + (\Phi - \mathbf{I})\mathbf{P}_{t-1} + \varepsilon_t . \tag{37}$$

In steady state, the expected queue lengths and production quantities are the same in all periods, so equation (37) becomes:

$$\mathbf{q}(\overline{\mathbf{P}}) + \Psi \cdot (\mathbf{P}_t - \overline{\mathbf{P}}) \approx \mathbf{q}(\overline{\mathbf{P}}) + \Psi \cdot (\mathbf{P}_{t-1} - \overline{\mathbf{P}}) + (\Phi - \mathbf{I})\mathbf{P}_{t-1} + \varepsilon_t , \tag{38}$$

where Ψ is a diagonal matrix whose ii'th element is $q_i'(\overline{P}_i)$. Simplifying (38) and solving the resulting equation for $\mathbf{P}_t$ yields:

$$\mathbf{P}_t = (\mathbf{I} - \Psi^{-1} + \Psi^{-1}\Phi)\mathbf{P}_{t-1} + \Psi^{-1}\varepsilon_t . \tag{39}$$

Taking the variance of (39) yields a recursion equation for var($\mathbf{P}_t$):

$$\begin{gathered} \mathbf{var}(\mathbf{P}_t) = \mathbf{B} \cdot \mathbf{var}(\mathbf{P}_{t-1}) \cdot \mathbf{B}^{\mathbf{T}} + \Psi^{-1}\Sigma\Psi^{-1}, \\ \text{where } \mathbf{B} = \mathbf{I} - \Psi^{-1} + \Psi^{-1}\Phi. \end{gathered} \tag{40}$$

Iterating this equation infinitely yields the desired first-order estimate for var(**P**):

$$\begin{gathered} \mathrm{var}(\mathbf{P}) \approx \sum_{s=0}^{\infty} \mathbf{B}^s \Psi^{-1}\Sigma\Psi^{-1}\mathbf{B}'^s, \\ \text{where } \mathbf{B} = \mathbf{I} - \Psi^{-1} + \Psi^{-1}\Phi. \end{gathered} \tag{41}$$

The power series may be evaluated numerically or analytically using the techniques discussed in Chapter 2. As with the development of MomentsQ, the power series expression for var(**Q**) is the same as that for linear control rules, with the smoothing matrix **D** replaced here by the diagonal matrix Ψ^{-1}. Thus, the first-order approximation maps a linear control rule onto the general control rule, with the smoothing factors being $1/q_i'(\overline{P}_i)$ rather than α_i. The resulting "lead times" of the fitted TPM model are given by $L_i = q_i'(\overline{P}_i)$. Here, the first derivatives of the inverse production function impact var(**P**). High first derivatives correspond to heavy smoothing by having long queue lengths, which will lower var(**P**) but

will likely raise var(**Q**). Low first derivatives correspond to less smoothing; indeed, the power series will not converge if $q_i'(\overline{P}_i) < 1$.

We use the production moments to estimate the queue length moments. Using the Delta Method applied to the inverse production functions, the second order-estimate of each E(Q_i), given estimates of E(**P**) and var(**P**), is:

$$E(Q_i) = q_i(\overline{P}_i) + \frac{1}{2} \cdot q_i''(\overline{P}_i) \cdot \text{var}(P_i) \tag{42}$$

We have two choices to estimate var(**Q**). First, we can use the Delta Method directly. Since $\mathbf{Q}_t = \mathbf{q}(\mathbf{P}_t)$, a first-order estimate of var(**Q**) is:

$$\mathbf{var(Q)} \approx \boldsymbol{\Psi} \cdot \mathbf{var(P)} \cdot \boldsymbol{\Psi}^{\mathbf{T}} \tag{43}$$

The drawback of this approximation is that the resulting var(**Q**) is a first-order approximation of a first-order approximation, and may be inaccurate.

Alternately, we can use the second-order estimate of E(**Q**) to estimate var(**Q**). We simply apply the power-series expression for var(**Q**) found in the previous section, so that:

$$\text{var}(\mathbf{Q}) \approx \sum_{s=0}^{\infty} \mathbf{B}^s \Sigma \mathbf{B}'^s \text{, where } \mathbf{B} = \mathbf{I} + (\Phi - \mathbf{I}) \cdot \Pi \tag{44}$$

This technique yields a first-order approximation of a second-order approximation, and should be more accurate. Indeed, we will see in Section 4 that this approximation for var(**Q**) usually is virtually identical to the approximation for var(**Q**) from MomentsQ.

The above discussion suggests an analytic algorithm to find the production and queue length moments.

Algorithm MomentsP

1. Calculate $\mathbf{E(P)} = (\mathbf{I} - \Phi)^{-1} \mu$.
2. Use E(**P**) to estimate $\text{var}(\mathbf{P}) \approx \sum_{s=0}^{\infty} \mathbf{B}^s \boldsymbol{\Psi}^{-1} \Sigma \boldsymbol{\Psi}^{-1} \mathbf{B}'^s$, where $\mathbf{B} = \mathbf{I} - \boldsymbol{\Psi}^{-1} + \boldsymbol{\Psi}^{-1} \boldsymbol{\Phi}$, and where $\boldsymbol{\Psi}$ is a diagonal matrix whose *ii*'th element is $q_i'(\overline{P}_i)$.
3. Use E(**P**) and var(**P**) to find E(**Q**), using the formulas $E(Q_i) = q_i(\overline{P}_i) + \frac{1}{2} \cdot q_i''(\overline{P}_i) \cdot \text{var}(P_i)$ for all stations *i*.
4. Use E(**Q**) to estimate $\text{var}(\mathbf{Q}) \approx \sum_{s=0}^{\infty} \mathbf{B}^s \Sigma \mathbf{B}'^s$, where $\mathbf{B} = \mathbf{I} + (\Phi - \mathbf{I}) \cdot \Pi$.

MomentsP has a significant advantage over MomentsQ in that it is an analytic rather than iterative algorithm. Thus, in general, we prefer to use MomentsP rather than MomentsQ. We will see an important case where MomentsQ is preferred, though, when we discuss multi-queue control rules in Section 5. Further, MomentsP assumes that we can work with the inverse production functions and their first and second derivatives directly. If doing so is difficult, i.e. the production functions cannot be inverted analytically, MomentsQ will be preferred, since this algorithm works directly with the production functions.

Example. Suppose that the stations in a job shop use clearing functions, which have the form $p_i(Q_i) = M_i Q_i / (\beta_i + Q_i)$. The inverse production functions have the form:

$$q_i(P_i) = \frac{\beta_i P_i}{M_i - P_i}. \tag{45}$$

The calculation of E(**P**) is automatic, and is given by $\mathbf{E(P)} = (\mathbf{I} - \Phi)^{-1}\mu$.

To calculate var(**P**), we need to calculate the entries of Ψ. Recall that $\Psi_{ii} = q_i'(E(P_i))$; applying this formula to the clearing functions yields:

$$q_i'(\overline{P}_i) = \frac{M_i \beta_i}{(M_i - \overline{P}_i)^2}, \tag{46}$$

so we immediately use formula (41) to estimate var(**P**).

The second-order estimate for E(**Q**) uses the formulas $E(Q_i) \approx q_i(\overline{P}_i) + \frac{1}{2} \cdot q_i''(\overline{P}_i) \cdot \text{var}(P_i)$ for all stations *i*. Applying these formulas to clearing functions yields:

$$E(Q_i) \approx \frac{\beta_i \overline{P}_i}{M_i - \overline{P}_i} + \frac{M_i \beta_i}{(M_i - \overline{P}_i)^3} \cdot \text{var}(P_i) \tag{47}$$

Finally, to estimate var(**Q**) we use the same formula used by MomentsQ, (30), with the expected queue length terms coming from the second-order estimates for E(**Q**). To do, we need to calculate the entries of the Π matrix, which again are:

$$\Pi_{ii} = \frac{M_i \beta_i}{(\beta_i + \overline{Q}_i)^2}. \tag{48}$$

We present MomentsP results for job shops using clearing functions in Section 4.

Numerical Example. Suppose we have a single station which uses a clearing rule, with parameters $M_i = 15$ and $\beta_i = 15$. The expected work arrival to the station is $\mu_i = 10$, and the variance of the work arrivals is $\Sigma_{ii} = 16$. We apply the MomentsP algorithm to this station.

Step 1: Calculate E(P). For a single station with no feedback, $E(P_i) = \mu_i = 10$.

Step 2: Estimate var(P). We apply (41) to estimate var(P_i). To do so, we first find $\Psi_{ii} = q_i'(E(P_i))$. Equation (46) is the first derivative of an inverse clearing function; using (46) yields $\Psi_{ii} = 15 \cdot 15/(15-10)^2 = 9$. Now, by defnition, $\mathbf{B} = \mathbf{I} - \mathbf{\Psi}^{-1} + \mathbf{\Psi}^{-1}\mathbf{\Phi}$. For a single station with no feedback, **B** becomes $1-(1/9)+(1/9)\cdot 0 = 8/9$. Then, applying (41), a first-order estimate for var(P_i) is: $\text{var}(P_i) \approx \sum_{s=0}^{\infty}(8/9)^{2s}(1/9)^2(16) = 0.9412$.

Step 3: Estimate E(Q). We apply (42) to estimate E(Q_i). To do so, we calculate $q_i(E(P_i))$ and $q_i''(E(P_i))$. Equation (45) is the inverse clearing function; applying it yields $q_i(E(P_i)) = 15 \cdot 10/(15-10) = 30$. Equation (47) gives the second derivative of the inverse clearing function; using it yields $q_i''(E(P_i)) = 2 \cdot 15 \cdot 10/(15-10)^3 = 3.6$. Then, applying (42) yields $E(Q_i) = 30 + 0.5 \cdot 3.6 \cdot 0.9412 = 31.6941$.

Step 4: Estimate var(Q) from E(Q). Finally, we apply (30) to estimate var(Q_i), using the second-order estimate for E(Q_i). To do so, we first find $\Pi_{ii} = p_i'(E(Q_i))$. Equation (48) is the first derivative of a clearing function; using (48) yields $\Pi_{ii} = 15 \cdot 15/(15+31.6941)^2 = 0.1032$. Now, the **B** used to calculate var(Q_i) is $\mathbf{B} = \mathbf{I} + (\mathbf{\Phi} - \mathbf{I}) \cdot \mathbf{\Pi}$. For a single station with no feedback, **B** becomes $1 + (0-1) \cdot 0.1032 = 0.8968$. Then, applying (30) yields $\text{var}(Q_i) \approx \sum_{s=0}^{\infty}(0.8968)^{2s}(16) = 81.7408$.

We compare the MomentsP approximations to simulation results. We performed two hundred 1500-period simulations of the station. The following table shows the results.

	MomentsP Results	Simulated Results (with 2-sigma conf. intervals)	Percent Difference
Expected Production	10.0000	10.0007 ± 0.0142	0.01%
Variance of Production	0.9412	0.9249 ± 0.0146	1.76%
Expected Queue Length	31.6941	31.7018 ± 0.1395	0.02%
Variance of Queue Lengths	81.7408	81.2414 ± 1.6177	0.61%

In this case, the MomentsP and simulation results were nearly identical, with all approximations except var(P_i) falling within the simulation confidence intervals. We will see in Section 4 that the MomentsP results closely track the simulation results if the maximum capacity (M_i) is significantly larger than the expected work arrival, and if the arrival variances are not large.

Extension: Generalized Single-Queue Control Rules. We extend the MomentsP algorithm to cover generalized single-queue control rules of the form $P_{it} = p_i(Q_{it}) + \beta_i + \gamma_{it}$, where β_i is a constant and γ_{it} is a random noise term with zero mean and finite variance. This generalized rule allows the modeling of production constants and fluctuations.

To begin the extension, we note that $Q_{it} = q_i(P_{it} - \beta_i - \gamma_{it})$, or in matrix-vector form,

$$\mathbf{Q}_t = \mathbf{q}(\mathbf{P}_t - \boldsymbol{\beta} - \boldsymbol{\gamma}_t) \tag{49}$$

Substituting (49) into equation (37) yields:

$$\mathbf{q}(\mathbf{P}_t - \boldsymbol{\beta} - \boldsymbol{\gamma}_t) = \mathbf{q}(\mathbf{P}_{t-1} - \boldsymbol{\beta} - \boldsymbol{\gamma}_{t-1}) + (\boldsymbol{\Phi} - \mathbf{I})\mathbf{P}_{t-1} + \boldsymbol{\varepsilon}_t . \tag{50}$$

From Proposition 1, $\mathbf{E(P)} = (\mathbf{I} - \Phi)^{-1}\mu$. To find var(**P**), we consider a first-order expansion of (50) around E(**P**):

$$\begin{aligned} \mathbf{q}(\overline{\mathbf{P}}) + \boldsymbol{\Psi}\cdot(\mathbf{P}_t - \boldsymbol{\beta} - \boldsymbol{\gamma}_t - \overline{\mathbf{P}}) &= \mathbf{q}(\overline{\mathbf{P}}) + \boldsymbol{\Psi}\cdot(\mathbf{P}_{t-1} - \boldsymbol{\beta} - \boldsymbol{\gamma}_t - \overline{\mathbf{P}}) + (\boldsymbol{\Phi} - \mathbf{I})\mathbf{P}_{t-1} + \boldsymbol{\varepsilon}_t, \\ \Rightarrow \mathbf{P}_t &= (\mathbf{I} - \boldsymbol{\Psi}^{-1} + \boldsymbol{\Psi}\boldsymbol{\Phi}^{-1})\mathbf{P}_{t-1} + (\boldsymbol{\gamma}_t - \boldsymbol{\gamma}_{t-1}) + \boldsymbol{\Psi}^{-1}\boldsymbol{\varepsilon}_t . \end{aligned} \tag{51}$$

Consequently, the first-order recursion equation for var($\mathbf{P}_t$) is:

$$\mathbf{var}(\mathbf{P}_t) = \mathbf{B}\cdot\mathbf{var}(\mathbf{P}_{t-1})\cdot\mathbf{B}^{\mathrm{T}} + (\Gamma_t + \Gamma_{t-1}) + \boldsymbol{\Psi}^{-1}\cdot\Sigma\cdot\boldsymbol{\Psi}^{-1}, \text{ where } \mathbf{B} = \mathbf{I} - \boldsymbol{\Psi}^{-1} + \boldsymbol{\Psi}^{-1}\boldsymbol{\Phi}, \tag{52}$$

and where Γ_t is the covariance matrix of γ_t. (Γ_t is assumed to equal a constant Γ for all t; we use the separate subscripts for the sake of clarity.) We iterate the recursion equation for var($\mathbf{P}_t$) infinitely, which yields the following power series:

$$\begin{gathered} \mathrm{var}(\mathbf{P}) \approx \Gamma + \sum_{s=0}^{\infty} \mathbf{B}^s\left(\boldsymbol{\Psi}^{-1}\cdot\boldsymbol{\Sigma}\cdot\boldsymbol{\Psi}^{-1\prime} + (\mathbf{B}-\mathbf{I})\Gamma(\mathbf{B}-\mathbf{I})'\right)\mathbf{B}'^s, \\ \text{where } \mathbf{B} = \mathbf{I} - \boldsymbol{\Psi}^{-1} + \boldsymbol{\Psi}^{-1}\Phi, \text{and } \boldsymbol{\Gamma} = \mathrm{cov}(\boldsymbol{\gamma}_t), \forall t \end{gathered} \tag{53}$$

We now use the two moments of **P** to calculate a second-order estimate for E(**Q**). We use the formula given in equation (42), modified for the fact that $Q_{it} = q_i(P_{it} - \beta_i - \gamma_{it})$ instead of $Q_{it} = q_i(P_{it})$. Then a second-order estimate for the expected queue length at each station i is given by:

$$E(Q_i) = q_i(\overline{P}_i - \beta_i) + \frac{1}{2}\cdot q_i''(\overline{P}_i - \beta_i)\cdot(\mathrm{var}(P_i) - \mathrm{var}(\gamma_i)) . \tag{54}$$

To calculate var(**Q**), we apply the new form of the production function to the queue-length recursion equation, (14),

$$\mathbf{Q}_t = \mathbf{Q}_{t-1} + (\boldsymbol{\Phi} - \mathbf{I})(\mathbf{p}(\mathbf{Q}_{t-1}) + \boldsymbol{\beta} + \boldsymbol{\gamma}_t) + \boldsymbol{\varepsilon}_t , \tag{55}$$

and taking the variance of (55) yields

$$\text{var}(\mathbf{Q}_t) \approx \mathbf{B} \cdot \text{var}(\mathbf{Q}_{t-1}) \cdot \mathbf{B}^T + (\Phi - \mathbf{I})\Gamma(\Phi - \mathbf{I})^T + \Sigma, \quad \text{where } \mathbf{B} = \mathbf{I} + (\Phi - \mathbf{I}) \cdot \Pi. \tag{56}$$

Then, by iterating (56) infinitely, a first-order estimate for var($\mathbf{Q}_t$) is:

$$\text{var}(\mathbf{Q}) \approx \sum_{s=0}^{\infty} \mathbf{B}^s \cdot \left[(\Phi - \mathbf{I})\Gamma(\Phi - \mathbf{I})^T + \Sigma\right] \cdot \mathbf{B}'^s, \quad \text{where } \mathbf{B} = \mathbf{I} + (\Phi - \mathbf{I}) \cdot \Pi. \tag{57}$$

The above estimates for the production and queue length moments use equations very similar to those used in MomentsP. Thus, they define a generalized MomentsP algorithm, as desired.

4. Empirical Behavior of Models with Single-Queue Control Rules

In this section, we study the empirical behavior of the approximation algorithms on models with single-queue control rules. All of our examples use single-queue clearing function for control rules. We begin by examining the behavior of a single station. From there, we consider a six-station chain. Finally, we study a thirteen-station network based on a mainframe subcomponent plant, considered by Fine and Graves (1988).

4.1 A Single Station

In this subsection, we examine the behavior of the approximation algorithms on an individual station. We have two objectives in doing so. First, we develop a general sense of how well the algorithms approximate the simulated performance of a single station, and determine general guidelines for when using the algorithms is appropriate. Second, we determine whether the MomentsQ or MomentsP algorithm provides markedly better performance. Note that unless the iterative algorithm provides markedly better performance, the analytic algorithm will be preferred because of its faster running time.

4.1.1 Simulation Parameters

In the simulation, the station receives work from a Gamma distribution, with an expectation of 10 units and one of 10 standard deviation settings ranging from 0.5 to 15. The station processes work via a clearing function with a maximum service rate, M_i, ranging from 105% to 300% of the arrival rate (10.5 to 30); the increase rate parameter, β_i, = M_i for all tests.

In practice, β_i will vary in order to model the behavior of the station being analyzed. Nonetheless, when modeling "generic" stations, setting β_i, = M_i is a natural choice. With this β_i, the first

derivative of the control rule approaches one as the queue length approaches zero. Thus, for low queue lengths, the station will behave (approximately) as if it processed all the work in its queue over the course of a period. Completely finishing small amounts of work in a single period is generally expected.

The following input standard deviation and maximum service rates were used:

- $M_i = \{10.5, 11, 12, 13, 15, 17, 20, 23, 26, 30\}$
- $\sigma_i = \{0.5, 1, 2, 3, 4, 6, 8, 10, 13, 15\}$

For each of the 100 scenarios, two hundred 1500-period simulations were performed.

4.1.2 Comparison of the MomentsQ and MomentsP Algorithms

The first graph, Figure 6, (on the next page) compares the differences between the simulated expected queue lengths and the approximated queue lengths. It presents the percentage differences between the simulated results and the MomentsQ and MomentsP results in ten separate charts. Each chart displays the results for all ten simulations with the same maximum service rate (M_i). Within each chart, the percentage differences are plotted against the input standard deviation, which ranges from 0.5 to 15. The ten charts share a common *Y*-axis, which shows the percent difference between the approximate queue lengths and the simulated queue lengths, with the formula:

$$\text{Error} = [(\text{Approximate } E(Q)) / (\text{Simulated } E(Q))] - 1.$$

Thus, a negative error value corresponds to the approximate E(Q) being lower than the simulated E(Q), and a positive value corresponds to the approximate E(Q) being higher than the simulated E(Q).

In general, the two approximations for E(Q) perform well, and perform approximately the same. The analytic approximations tend to be a bit higher than the iterative approximations, but the differences usually are minor. The only scenarios for which the approximations do not perform well are where the maximum service rates approach the arrival rates (M_i = 10.5, 11, 12), and the input standard deviations are high (10 or higher). These results are not surprising. Recall that estimating the queue lengths involves taking quadratic approximations of the inverse production functions. With maximum service rates that are only 105% and 110% of expected arrivals, the inverse production functions will be nearly vertical, implying that approximations of the inverse production function will not be terribly accurate. Indeed, with M_i = 11, for example, the simulated average queue length ranged between 110 and 207, depending on the input standard deviation. These queue lengths are much greater than the expected production of 10 units. With such high queue lengths, significant errors would be expected. All differences were less than 25%, even with the lowest M_i's, which is quite good.

Similarly, greater input standard deviations are associated with greater differences. These results are not surprising. Greater input standard deviations increase the second-order variance terms in the

estimation formulas. Larger second-order terms naturally increase approximation errors, and the larger variances inflate the effects of variance-estimation errors.

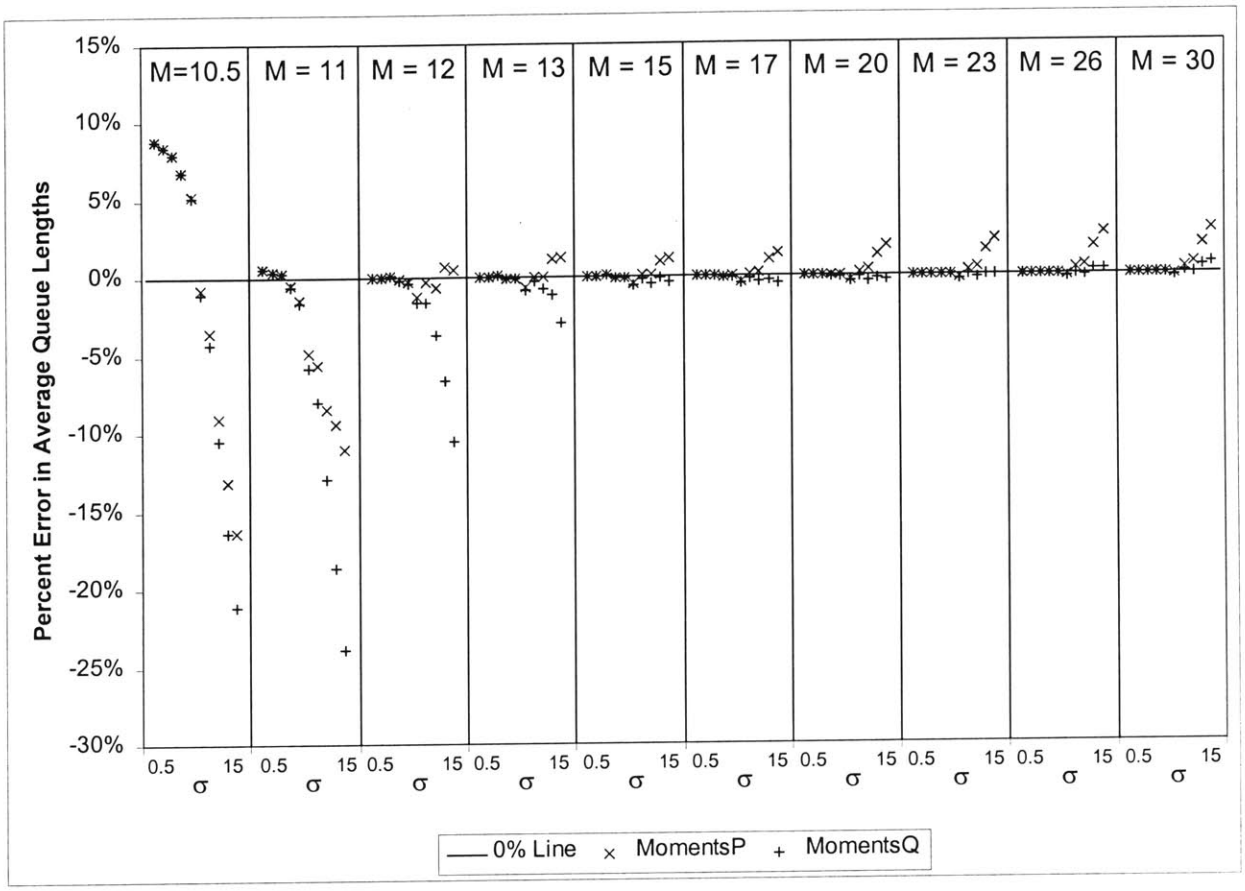


Figure 6 -- Simulated Queue Lengths v. Approximate Queue Lengths

The MomentP approximations are biased to be greater than the MomentQ approximations. As a result, the MomentP approximations performed significantly better on scenarios with low maximum service rates (such as 12), and the MomentQ approximations performed better on scenarios with high maximum service rates and input standard deviations (MSR > 15, and input standard deviations > 1). As will be shown shortly, errors estimating production variances cause these queue length errors. Nonetheless, if the maximum service rate was greater than 120% of the expected arrival rate, and the input standard deviation was less than the expected arrival rate, the errors between the simulated and approximate results were less than 5%, and usually less than 2%. Also, there never appears to be a strong reason to use the iterative MomentQ algorithm unless great accuracy is desired for a model with high maximum service rates and input variances.

The next graph, Figure 7, compares the differences between the simulated queue length standard deviations and the approximate queue-length standard deviations. Three approximations to the queue length standard deviations are compared:

- MomentsQ, which uses E(**Q**) to calculate var(**Q**).
- MomentsP, which uses the second-order estimate of E(**Q**) to calculate var(**Q**).
- A variant of MomentsP, which uses the first-order estimate of var(**P**) to calculate var(**Q**).

The format of Figure 2 is identical to that of Figure 1. Figure 2 presents ten charts, each displaying the percentage differences for the ten simulations with the same M_i. Within charts, the percent differences are plotted by the input standard deviations.

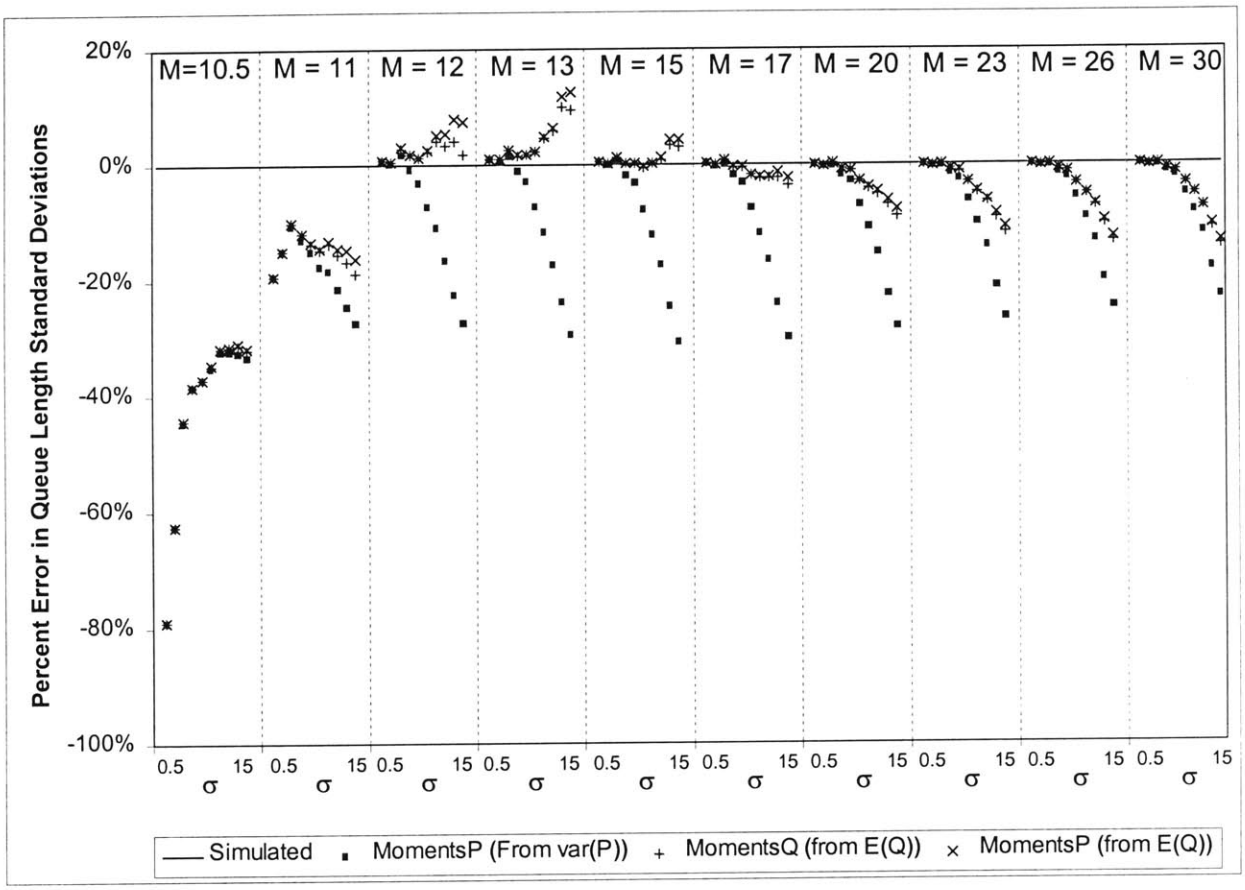


Figure 7 -- Simulated Queue Length Standard Deviations v. Approximate Queue Length Standard Deviations

We first note that the estimation errors for the queue-length standard deviations are significantly worse than the estimation errors for the average queue lengths. This result is not surprising given that the queue length standard deviation approximations are first-order estimates. Again, the approximations degrade with low maximum service rates and high input variances, to the point that the estimates with very low maximum service rates are not useful. In general, though, if the maximum service rate is greater than 120% of the arrival rate, and the input standard deviation is less than the arrival rate, the errors were under 10%.

Note that the E(**Q**)-based approximations were much more accurate than the var(**P**)-based approximation. Recall that the latter estimate is a first-order approximation of a first-order approximation, so its poor performance was not surprising. However, the two E(**Q**)-based approximations were virtually identical, so there appears to be no reason to use the more expensive MomentsQ algorithm with clearing functions.

Finally, Figure 8 compares the difference between the simulated production standard deviations and the approximate standard deviations. Three approximations to the standard deviations are compared:

- MomentsQ, which uses the first-order estimate of var(**Q**) to estimate var(**P**).
- MomentsP, which uses E(**P**) to estimate var(**P**).
- A variant of MomentsP, which uses the first-order estimate of var(**Q**) to calculate var(**P**).

The format of Figure 3 is identical to that of Figure 1. Figure 3 presents ten charts, each displaying the percentage differences for the ten simulations with the same M_i. Within charts, the percent differences are plotted by the input standard deviations.

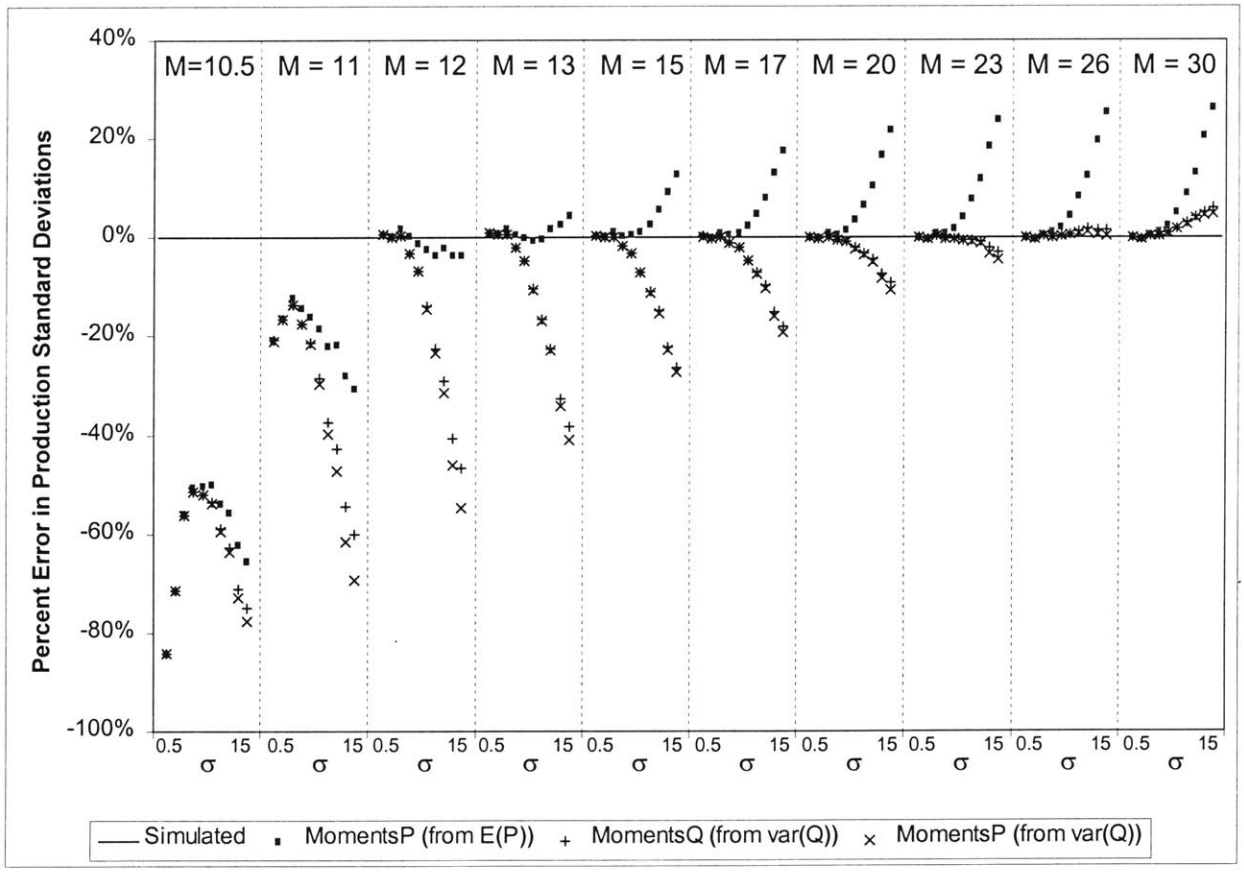


Figure 8 -- Simulated Production Standard Deviations v. Approximate Production Standard Deviations

Again, the errors from production standard deviations are significantly worse than the errors for the average queue lengths, with estimates with low maximum service rates being meaningless. As with the queue length standard deviations, the errors are generally less than 10% if the max service rates are greater than 120% of the arrival rate, and the input standard deviations are less than the arrival rate.

However, with queue length standard deviations, the E(**Q**)-based approximations clearly provided greater accuracy than the var(**P**)-based approximations. It is not possible to make such a clear distinction in this case. For medium maximum service rates (between 120% and 170% of the arrival rate), the E(**P**)-based approximation were more accurate. However, as the maximum service rates increased to 200% of the arrival rate or greater, the var(**Q**)-based approximations were considerably more accurate. This result is not immediately obvious since var(**Q**)-based approximations are first-order approximations of first-order approximations.

The chart suggests an explanation. Note that E(**P**)-based approximations are greater than the var(**Q**)-based approximations. Further, the mean errors for both types of approximations increase with the maximum service rate, and the rate of the increase diminishes with higher maximum service rates. These two factors cause the var(**Q**)-based estimates to become accurate over a fairly wide interval of high maximum service rates. Note that both var(**Q**)-based estimates were virtually identical over this range, so the analytic algorithm may be used when a var(**Q**)-based estimate is desirable.

The chart also explains the pattern of queue length errors shown in Figure 6. Comparing the two charts, we find that the queue length errors are greatest where the production standard deviation errors are greatest. Thus, the analytic algorithm (which uses E(P)-based approximations for var(P)), is less accurate on models with high maximum service rates and input variances. Conversely, the iterative algorithm is less accurate on models with low maximum service rates.

4.2 A Six-Station Assembly Line

In this section, we study the behavior of the approximation algorithms on multi-station networks. In particular, we study whether differences between the simulated and approximated moments increase for stations that are farther downstream, and if so, how pronounced the effects are.

We consider a six-station assembly line, in which each station processes an average of 10 units of work each time periods. We characterize work transfers between successive stations as follows: the work arriving at station i is the work completed at station $i-1$ last period ($\Phi_{i,i-1}=1$), plus a random noise term, ε_{it}. The noise terms have an expectation of 0, and a variance, Σ_{ii}, that varies with the tested scenario. All stations use clearing functions. Figure 9 shows the assembly line.

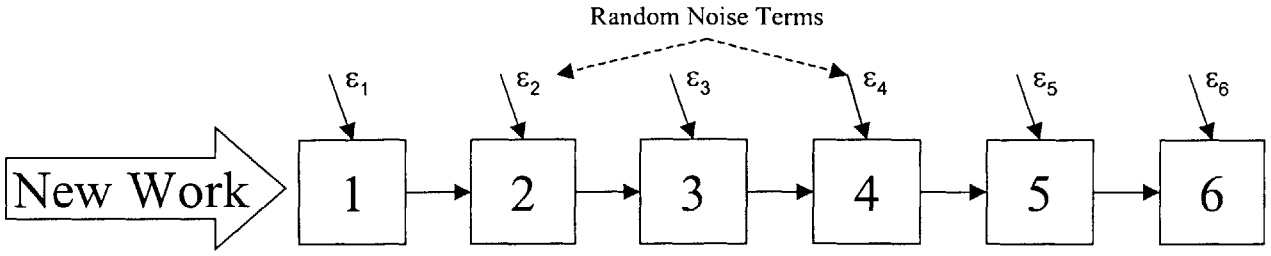


Figure 9 -- A Six-Station Assembly Line

We consider nine scenarios for the shop, varying the maximum service rates and estimated production standard deviation at each station.

We restrict our tests to the MomentsP algorithm, which is usually preferred because it is an analytic algorithm. We perform tests specifically designed to determine whether errors ever increase with station position. To perform these tests, we assume that the stations are as similar as possible, with the exception of line position. Thus, all stations have the same expected arrival rate, the same clearing function parameters (M_i and β_i), and, ideally, the same production standard deviation.

Of course, the production standard deviations cannot be determined exactly. Instead, we use MomentsP to set the input variances (the Σ_{ii}'s) so that all the production variances are equal. For example, suppose we want the production standard deviation at each station to be one unit. We first find the input variance to the first station, Σ_{11}, that makes the production standard deviation at the first station (as estimated by MomentsP) equal one. We next find the input variance to the second station, Σ_{22}, that makes the production standard deviation at the second station equal one. Note that the input variance calculation at the second station accounts for the estimated variance in the work arriving from station 1. We then calculate the input variances for the remaining stations, 3 to 6.

The following table shows the clearing function parameters and the desired production standard deviation for each scenario. (Note that $M_i = \beta_i$ for all the scenarios.) The table also shows the input standard deviations ($\sqrt{\Sigma_{ii}}$) used for each scenario, as calculated by the MomentsP algorithm.

Clearing Function (M_i and β_i)	*Production Standard Deviation*	*Input Standard Deviation, $\sqrt{\Sigma_{ii}}$*					
		1	*2*	*3*	*4*	*5*	*6*
14	1	4.85	3.43	2.99	2.83	2.86	2.97
14	2	9.70	6.85	5.99	5.67	5.72	5.94
14	3	14.54	10.28	8.98	8.50	8.58	8.91
20	1	2.65	1.85	1.61	1.47	1.38	1.31
20	3	7.94	5.55	4.83	4.41	4.13	3.92
20	5	13.23	9.26	8.05	7.35	6.88	6.53
26	1	2.07	1.42	1.24	1.14	1.06	1.01
26	3	6.21	4.27	3.73	3.41	3.19	3.03
26	5	10.35	7.11	6.22	5.69	5.32	5.05

Each scenario was simulated 100 times; each run comprised 2000 periods (plus a 100-period start-up).

The following set of graphs, Figure 10, compares the percentage differences between the approximate and simulated expected queue lengths for each station in the chain, across the nine scenarios. The gray lines plot the hypothesis limits for whether the approximate expected queue lengths equal the true expected queue lengths. (Each hypothesis limit equals $\pm$twice the standard deviation of the simulated expected queue lengths.) Graphs in the same row have equally-sized y-axes. (Note that graphs in the same row have identical production standard deviations.)

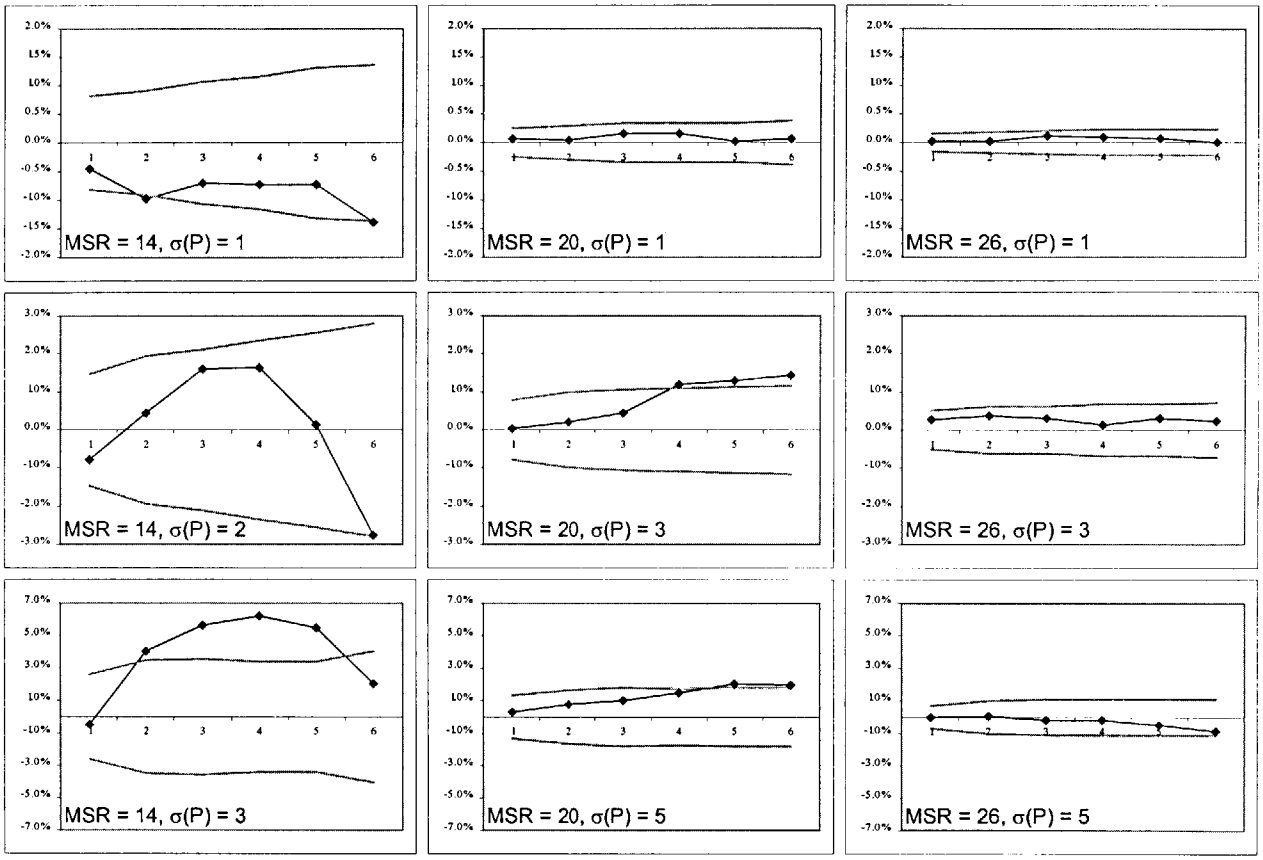


Figure 10 -- Approximate v. Simulated Expected Queue Lengths in a 6-station Chain

The approximate and simulated estimated are within 3% of each other in most of the scenarios, which is consistent with the results in the single-station model. Further, the errors often do not appear to follow any patterns. The estimates for station towards the end of the chain generally do not seem worse than the estimates for stations towards the beginning of the chain. The exceptions are the scenarios with a maximum service rate of 20 and production standard deviations of 3 or 5; there appear to be slight order biases in these scenarios. Later on, we will see that these biases are probably caused by errors in the production standard deviation estimates. Note, however, that the effects are not pronounced.

One scenario has fairly significant differences between the simulated and estimated service lengths (maximum service rate = 14, production standard deviation = 3). Such differences are not surprising, given the scenario's very high input variances. We note that, oddly, station 6's simulated average queue length is closer to its estimated queue length than four the preceding four stations. However, given the wide confidence intervals of this scenario, the result is likely due to random noise.

The next set of graphs, Figure 11, compares the percentage differences between the approximate and estimated queue length standard deviations. The gray lines plot the hypothesis limits for whether the approximate standard deviations equal the true standard deviations. Graphs in the same row have equally-sized *y*-axes.

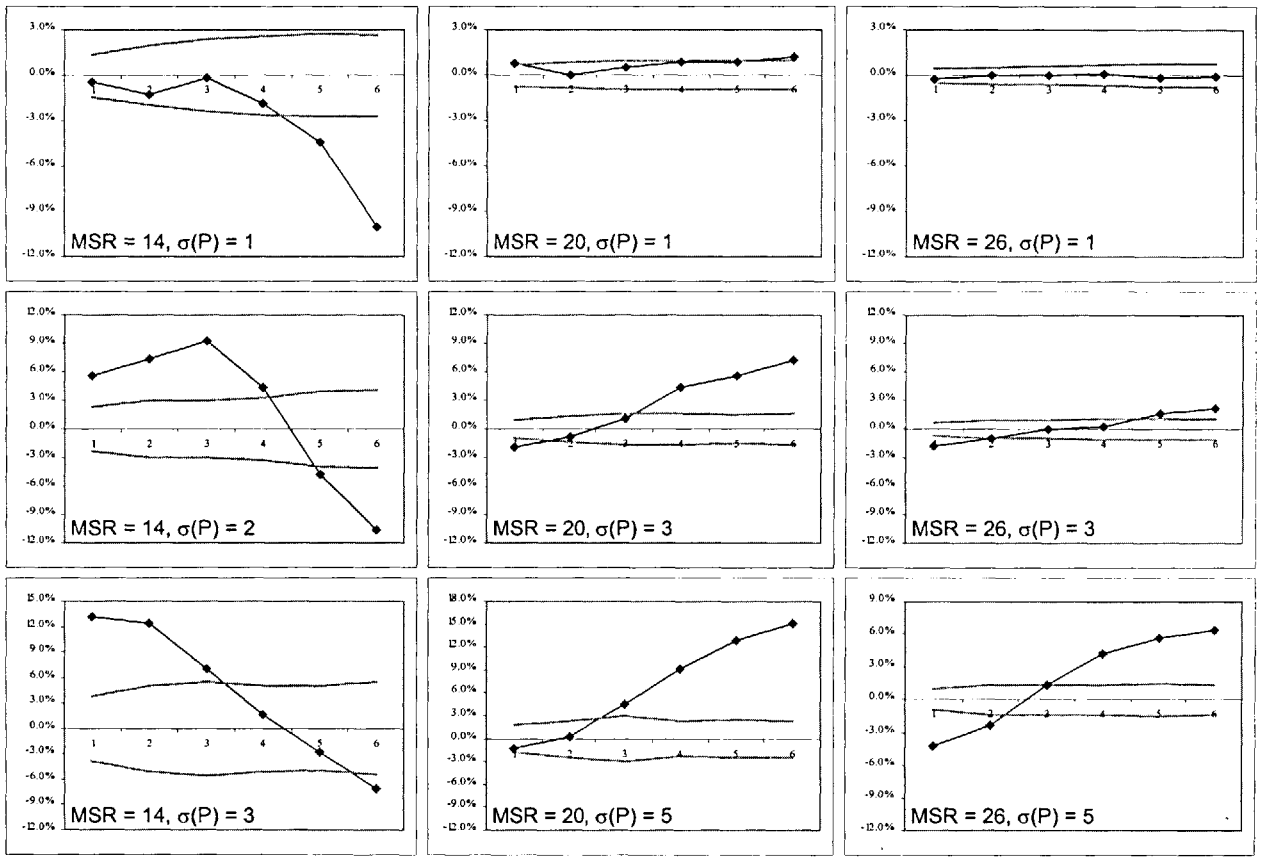


Figure 11 -- Approximate v. Simulated Queue Length Standard Deviations in a 6-station Chain

As in the single station case, the estimates of the queue length standard deviations are generally less accurate than the expected queue length estimates. Here, we do see some location biases. With the higher maximum service rates, downstream overestimate the queue length variances compared with the upstream stations. The effect, however, is only pronounced with high input standard deviations (i.e., the desired production standard deviation equals 5).

The patterns of differences are reversed when the maximum service rate is 14; then, downstream stations underestimate the queue length standard deviations with respect to the upstream stations. Note that in some cases, this underestimating effect corrects for estimation errors at upstream stations, making the estimates more accurate at the downstream station. In addition, in the scenario with the production standard deviation of 1, station 6's estimated and simulated queue length standard deviations are quite different from the rest of the stations. While the underestimating effect holds for this scenario, as well, part of the difference is probably due to random noise (given the size of the confidence intervals).

Finally, the following set of graphs, Figure 12, compares the approximate and simulated production standard deviations. The gray lines plot the hypothesis limits for whether the approximate standard deviations equal the true standard deviations. Graphs in the same row have equally-sized *y*-axes.

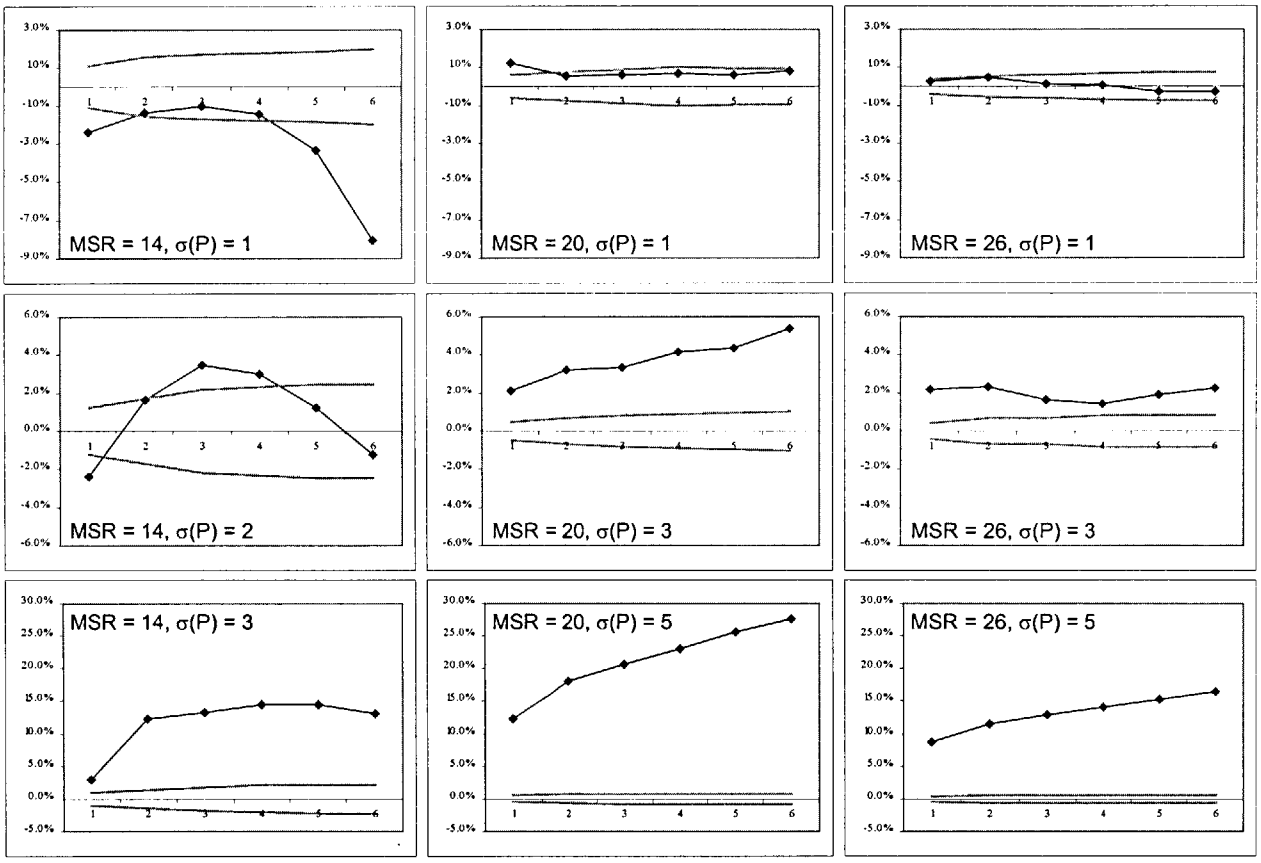


Figure 12 – Approximate v. Simulated Production Standard Deviations in a 6-station Chain

As with production standard deviations, there is some location bias for the higher maximum service rates. As with the queue length standard deviations, downstream stations overestimate production standard deviations. The effect is pronounced when the input standard deviations are high (which occurs when the desired production standard deviation is 5). Note that this pattern of errors likely explains the slight

biases in the expected queue length results, since MomentQ's average queue length estimations depend on the estimated production variances.

Unlike the queue length standard deviations, the production standard deviations do not show any distinct trends when the maximum service rates are low. The exception is that in the scenario with the production standard deviation of 1, station 6's estimated and simulated queue length standard deviations are quite different from the rest of the stations. However, this result just appears to be the same outlier discussed above (for queue length standard deviations).

Thus, in the scenarios tested, the expected queue lengths showed little if any location biases; the queue length and production standard deviations did show location biases, especially with high maximum service rates and high input standard deviations. Figure 3, in the previous section, implies a possible explanation – the analytic algorithm increasingly overestimates the production standard deviations with high maximum service rates and input variances. When this phenomenon is repeated throughout an entire chain of (ideally) identical stations, it can create a reinforcing cycle. Each station overestimates the production variances of the previous station, thus causing the station to overestimate its own production variance and propagate the error downstream. Nonetheless, in most cases likely to be seen in practice (that do not have input standard deviations greater than the expected production quantities, or have chains of identical stations), the effect does not appear significant enough to cause serious problems with the accuracy of the approximations.

4.3 A Job Shop that Manufactures Mainframe Subcomponents

In this section, we consider a job shop that manufactures subcomponents of mainframe computers. Fine and Graves (1989) first considered the job shop, located at an IBM plant in Poughkeepsie, New York, as an application of the original Tactical Planning Model. Here, we consider a variant of the shop that uses the clearing functions as production functions.

Figure 13 shows a diagram of the job shop. The shop contains 13 stations; work enters through the NB Release and OTN Release stations and exits through the Encapsulation station. There is a feedback loop in the flow, related to subcomponents failing tests and requiring repair work. The numbers on each arc (i,j) represent the expected amount of work that arrives at station$j$ given one unit of work at station i.

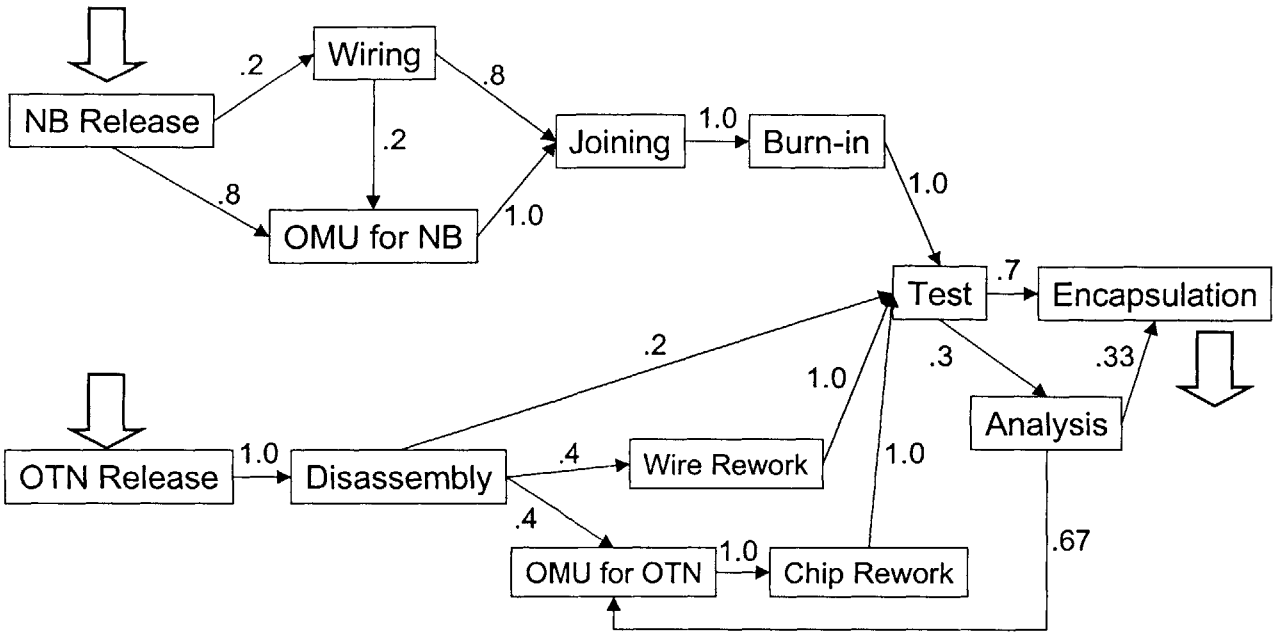
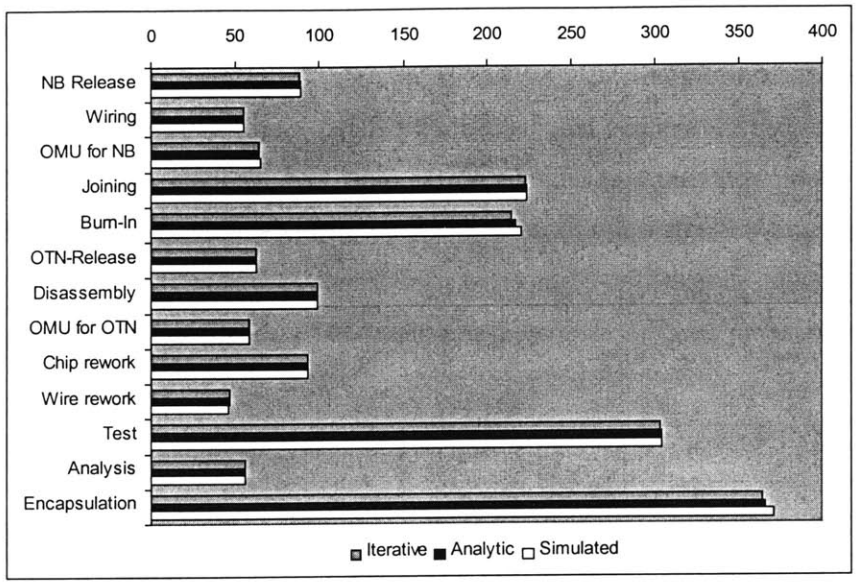


Figure 13 -- A Job Shop that Manufactures Subcomponents of Mainframe Computers

The following table shows the input data for the shop, including the average work that arrives to the NB Release and OTN Release stations, and the input standard deviations, the expected production quantities, and maximum capacities at each station. Note that a station's maximum capacity becomes both the M_i and β_i parameters of the station's clearing function.

Station	*Average Arrivals per Period*	*Input Standard Deviation*	*Maximum Capacity*	*Expected Production*
NB release	40	20	80	40
Wiring		6.3	16.3	12
OMU for NB		0	60	30.4
Joining		0	50	40
Burn-in		10	50	40
OTN release	30	12.5	60	30
Disassembly		12.5	45	30
OMU for OTN		0	60	29.6
Chip rework		12.5	45	29.6
Wire rework		12.5	18.8	12
Test		17.5	125	87.6
Analysis		0	50	26.3
Encapsulation		7.5	87.5	70

The following graphs compare the results of the iterative and analytic algorithms to a 250,000 period simulation of the job shop. The bar chart in Figure 14 compares the simulated and approximated expected queue lengths at each station. The table in Figure 14 presents the percent differences between the approximate and simulated results.



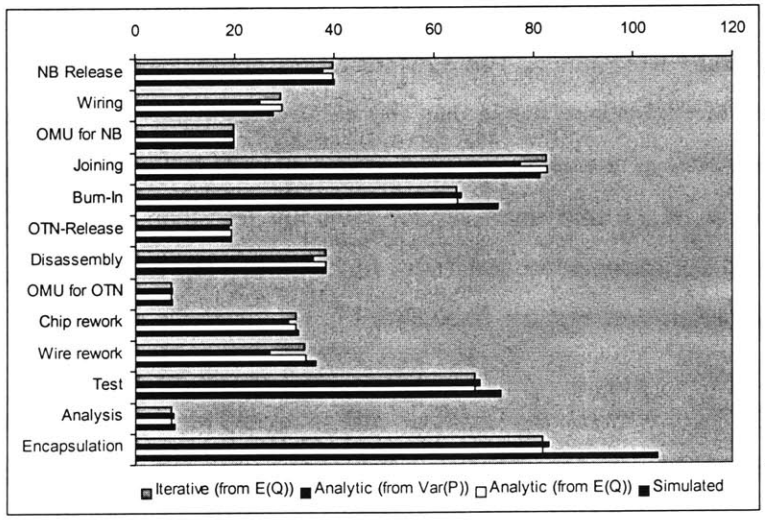
Percent Difference Between Approximations and Simulation Results

Station	*Iterative*	*Analytic*
NB Release	-0.30%	-0.16%
Wiring	0.56%	0.47%
OMU for NB	-0.34%	-0.09%
Joining	-0.65%	-0.15%
Burn-In	-2.82%	-1.77%
OTN-Release	0.10%	0.13%
Disassembly	0.20%	0.18%
OMU for OTN	0.03%	0.11%
Chip rework	0.26%	0.31%
Wire rework	1.08%	1.34%
Test	-0.34%	-0.05%
Analysis	-0.06%	0.05%
Encapsulation	-1.87%	-1.48%

Figure 14 – Expected Queue Lengths for the Mainframe Subcomponents Job Shop

In general, the iterative and analytic approximations are quite close to the simulated results, with no difference being greater than 3%. The larger differences track closely to the differences observed in the single-station case, not to the location of a station in the job shop. For example, the Burn-in and Encapsulation stations, which have the greatest differences, also have very high queue length and production variances (discussed below). They are also heavily loaded (maximum service rate / expected production is 1.25 for the Burn-in station, and 1.17 for the Encapsulation station). In the previous sections, combinations of heavy loading with significant production variances were associated with inaccuracies in estimating the expected queue lengths.

Figure 15 compares the estimated and simulated standard deviations of the queue lengths.



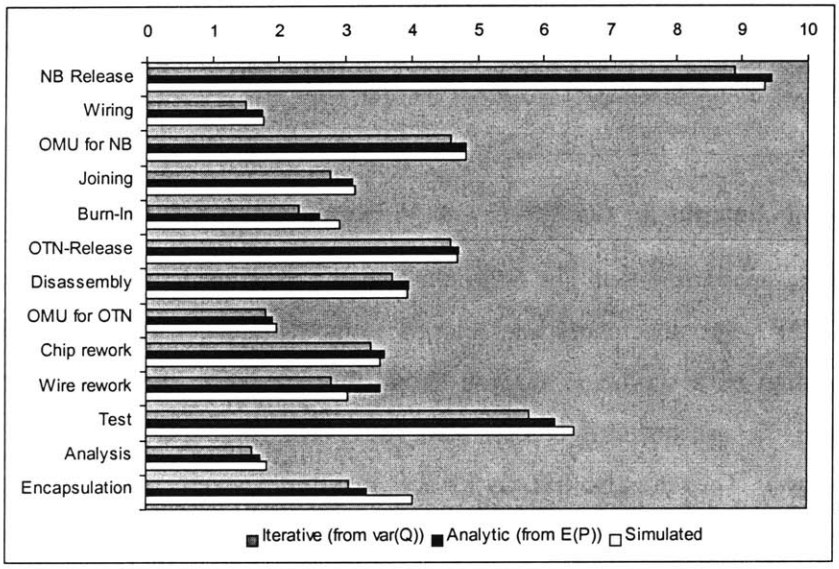
Percent Difference Between Approximations and Simulation Results

Station	*Iterative (from E(Q))*	*Analytic (from Var(P))*	*Analytic (from E(Q))*
NB Release	-1.02%	-5.50%	-0.95%
Wiring	5.37%	-9.85%	5.43%
OMU for NB	-0.50%	-0.46%	-0.37%
Joining	1.62%	-4.62%	2.04%
Burn-In	-11.53%	-10.14%	-10.99%
OTN-Release	-0.38%	-2.45%	-0.37%
Disassembly	-0.06%	-6.71%	-0.07%
OMU for OTN	-5.85%	-1.13%	-5.92%
Chip rework	-1.92%	-6.62%	-1.91%
Wire rework	-6.16%	-25.62%	-5.59%
Test	-7.24%	-5.78%	-7.19%
Analysis	-9.33%	-4.09%	-9.53%
Encapsulation	-22.08%	-20.83%	-22.00%

Figure 15 -- Queue Length Standard Deviations for the Mainframe Subcomponents Job Shop

Again, the estimates track fairly well with the simulated results; most differences were less than ten percent. The biggest differences were for the heaviest-loaded stations, Burn-In and Encapsulation. Usually, the estimates based on the expected queue lengths did a bit better than the estimate based on production variances, although there were some exceptions at the Test and Analysis stations. The estimate based on production variances was especially far off at the Wire rework station; presumably, this is because the estimate of the production standard deviation erred significantly (see below).

The third graph, Figure 16, compares the estimated and simulated standard deviations of the production quantities.



Percent Difference Between Approximations and Simulation Results

Station	*Iterative (from var(Q))*	*Analytic (from E(P))*
NB Release	-4.72%	1.24%
Wiring	-14.48%	-0.75%
OMU for NB	-4.74%	0.19%
Joining	-11.86%	-1.42%
Burn-In	-21.11%	-10.06%
OTN-Release	-2.30%	0.44%
Disassembly	-5.90%	0.58%
OMU for OTN	-7.68%	-2.35%
Chip rework	-4.17%	1.35%
Wire rework	-8.08%	16.06%
Test	-10.37%	-4.29%
Analysis	-11.85%	-5.88%
Encapsulation	-23.94%	-17.57%

Figure 16 -- Production Standard Deviations for the Mainframe Subcomponents Job Shop

The estimates continue to track well with the simulated results, although in general there is much more of a performance gap between the iterative and analytic estimates. Here, the analytic estimates, based on expected production results, generally were better. These results do not conflict with the single-station results; recall that the analytic estimates became worse than the iterative estimates when the (1) maximum service rate was greater than 200% of the expected production, and (2) the input standard deviations were high. In this model, most of the maximum service rates are less than 200% of the expected production, and the input standard deviations never reach very high values. The exception is the Wire rework station; its input standard deviation, not counting fluctuations in work arrivals, is 105% of its expected production.

The other stations with great differences between the simulated and estimated results, Burn-in and Encapsulation, are the stations that previously have had the greatest differences. Again, these differences are associated with the heavy loading at these stations.

Most importantly, the differences between the simulated and approximate results were not noticeably associated with station locations. Thus, for example, estimates for stations near the end of the work flows were not noticeably worse than those for stations at the front of the work flow. Consequently, the approximation algorithms discuss in this chapter should provide useful results for multi-station networks.

5. Steady-State Analysis for Models with Multi-Queue Control Rules

In the previous sections, we assumed that each station used a single-queue control rule, $P_{it} = p_i(Q_{it})$. In this section, we extend our results to multi-queue control rules. We consider rules of the form $P_{it} = p_i(\mathbf{Q}_t)$, where $\mathbf{Q_t}$ is the vector of all queue levels at time t. In matrix-vector form, this rule is still $\mathbf{P}_t = \mathbf{p}(\mathbf{Q}_t)$, but there is no longer a requirement that the individual production values reduce to functions of single queue lengths. Similarly, the inverse production functions are $\mathbf{Q}_t = \mathbf{q}(\mathbf{P}_t)$, with individual queue lengths now being a function of multiple production values. We present extensions to both the MomentsP and MomentsQ algorithms; either approach may be preferable, depending on the nature of production functions.

5.1 The Delta Method for Models with Multi-Queue Control Rules

We first present multi-queue generalizations of the Delta Method formulas developed in Section 2. Suppose that $Y = g(\mathbf{X})$, where $\mathbf{X}$ is a vector. The second-order Taylor-series of expansion of Y around $E(\mathbf{X})$ is:

$$Y \approx g(\overline{\mathbf{X}}) + \sum_i (X_i - \overline{X}_i)\frac{\partial g(\overline{\mathbf{X}})}{\partial X_i} + \frac{1}{2}\sum_i \sum_j (X_i - \overline{X}_i)(X_j - \overline{X}_j)\frac{\partial^2 g(\overline{\mathbf{X}})}{\partial X_i \partial X_j}. \tag{58}$$

Then a second-order approximation for $E(Y)$ is given by:

$$E(Y) \approx \mathbf{g}(\overline{\mathbf{X}}) + \frac{1}{2}\mathbf{e}^T \cdot \left(\text{var}(\mathbf{X}) \bullet \nabla^2 g(\overline{\mathbf{X}})\right) \cdot \mathbf{e}, \tag{59}$$

where the large dot-sign represents array multiplication, $\mathbf{e}$ is a unit vector, and $\nabla^2 g(\overline{\mathbf{X}})$ is a Hessian matrix such that $\nabla^2 g(\overline{\mathbf{X}}) = \left(\partial^2 g(\overline{X})/\partial X_i \partial X_j\right)$.

A first-order approximation for $\text{var}(Y)$ is given by:

$$\text{var}(Y) \approx \nabla g(\overline{\mathbf{X}}) \cdot \mathbf{var}(\mathbf{X}) \cdot \nabla g(\overline{\mathbf{X}})^{\mathrm{T}}, \tag{60}$$

where $\nabla g(\overline{\mathbf{X}})$ is a gradient vector such that $\left(\nabla g(\overline{\mathbf{X}})\right)_i = \partial g(\overline{X})/\partial X_i$.

5.2 The Multi-Queue Form of the MomentsQ Algorithm

The equations of the multi-queue MomentsQ algorithm appear similar to those of the single-queue MomentsQ algorithm. However, the calculations of the queue-length moments are considerably more complicated. Previously, we generated successive estimates for the expected queue lengths by solving single-variable equations for those estimates. Now, we must solve systems of nonlinear equations to find the successive estimates. We find that there is one class of common multi-queue control rules for which the systems solve easily. This is the class of *separable control rules*.

We begin this section by deriving the equations of the multi-queue MomentsQ algorithm. We then show how this algorithm becomes simple to use for models with using separable control rules.

5.2.1 Algorithm Development

Step 1: **Initialization**. As in the single-queue control rule case, we begin by finding E(**P**), and use E(**P**) to find a first-order approximation of E(**Q**). Now, however, solving E(**Q**) requires finding the solution to the system of nonlinear equations

$$p_i(\overline{\mathbf{Q}}) = \overline{P}_i, \text{ for all stations } i. \tag{61}$$

We then use this value of E(**Q**) to calculate a first-order estimate of var(**Q**), again using the power-series equation

$$\begin{gathered}\text{var}(\mathbf{Q}) \approx \sum_{s=0}^{\infty} \mathbf{B}^s \Sigma \mathbf{B}'^s, \\ \text{where } \mathbf{B} = \mathbf{I} + (\Phi - \mathbf{I}) \cdot \Pi.\end{gathered} \tag{62}$$

Now, however, Π is a non-diagonal matrix whose i,j entry is $\partial p_i(\overline{\mathbf{Q}})/\partial Q_j$. This matrix definition follows from the fact that the first-order expansion of $p_i(\mathbf{Q}) \approx p_i(\overline{\mathbf{Q}}) + \sum_j (Q_j - \overline{Q}_j) \cdot \partial p_i(\overline{\mathbf{Q}})/\partial Q_j$, for all stations j. Then, writing all of the first-order expansions for all production functions simultaneously yields $\mathbf{p}(\mathbf{Q}_t) \approx \mathbf{p}(\overline{\mathbf{Q}}) + \mathbf{\Pi} \cdot (\mathbf{Q}_t - \overline{\mathbf{Q}})$, where the elements of Π are those previously described.

Step 2: Iteration. The next step is to use the newly found var(Q) estimate to generate a second-order estimate for E(**Q**). To do so requires solving the system of nonlinear equations

$$p_i(\overline{\mathbf{Q}}) + \frac{1}{2}\mathbf{e}' \cdot \left(\text{var}(\mathbf{Q}) \bullet \nabla^2 p_i(\overline{\mathbf{Q}})\right) \cdot \mathbf{e} = \overline{P}_i, \text{ for all stations } i, \tag{63}$$

where the large dot-sign represents array multiplication, **e** is a vector of ones, and $\nabla^2 p_i(\overline{\mathbf{Q}})$ is a Hessian matrix whose *i,j* element is $\partial^2 p_i(\overline{\mathbf{Q}})/\partial Q_i \partial Q_j$. This system results from applying the second-order equation for E(*Y*) to each of the E(P_i)'s.

Once this nonlinear system is solved, the new estimate for E(**Q**) is used to generate a new estimate for var(**Q**).

Step 3: Convergence. Step 2 is repeated until successive estimates of E(**Q**) and var(**Q**) converge to desired limits.

5.2.2 The MomentsQ Algorithm and Separable Production Functions

The MomentsQ algorithm can be quite useful when paired with *separable* multi-queue production functions. A separable production function is a sum of single-queue production functions, and has the following form:

$$P_{it} = \sum_j k_{ij} p_j(Q_{jt}), \tag{64}$$

where the k_{ij}'s are constants, and each p_j is a single-queue production function. With this form of production function, the systems of nonlinear equations for the expected queue lengths decompose into a set of equations where each equation contains a single expected queue length variable. To show this, we first note that with a separable production function, $\partial^2 p_i(\overline{\mathbf{Q}})/\partial Q_i \partial Q_j = 0$ unless $i = j$. Then, applying (63) to a network with separable production functions yields the following system:

$$\overline{P_i} = \sum_j k_{ij} \cdot \left[p_j(\overline{Q_j}) + \frac{1}{2} p_j''(\overline{Q_j}) \cdot \mathrm{var}(Q_j) \right], \text{ for all stations } i. \tag{65}$$

If the quantities in the brackets are temporarily treated as variables, (65) becomes a linear system. Solving for the quantities in the brackets yields the set of equations:

$$p_j(\overline{Q_j}) + \frac{1}{2} p_j''(\overline{Q_j}) \cdot \mathrm{var}(Q_j) = \left(\mathbf{K}^{-1}\overline{\mathbf{P}}\right)_i, \text{ for all stations } j, \tag{66}$$

where $\left(\mathbf{K}^{-1}\overline{\mathbf{P}}\right)_i$ is the *i*th entry of the vector indicated, and **K** is the matrix of k_{ij} terms. Each equation in this new set contains a single expected queue length variable, as desired. Thus, with separable production functions, using the MomentsQ in the multi-queue case is not much more difficult than using the algorithm in the single-queue case.

Example. Consider a general, but separable form of a clearing function,

$$p_i(\mathbf{Q}_t) = \sum_j k_{ij} \cdot \frac{M_j Q_{jt}}{Q_{jt} + \beta_j}. \tag{67}$$

The system of equations for the expected queue lengths solved at each iteration simplifies to:

$$\frac{M_j \overline{Q}_j}{\overline{Q}_j + \beta_j} - \frac{M_j \beta_j}{(\overline{Q}_j + \beta_j)^3} \cdot \text{var}(Q_j) = \left(\mathbf{K}^{-1}\overline{\mathbf{P}}\right)_i, \text{ for all stations } j. \tag{68}$$

The entries of the Π matrix used to calculate the queue length variances at each iteration are:

$$\Pi_{ij} = \frac{\partial p_i(\overline{\mathbf{Q}})}{\partial Q_j} = \frac{k_{ij} \cdot M\beta}{(\overline{Q}_j + \beta)^2}, \forall i, j. \tag{69}$$

Thus, it is almost as simple to process the generalized clearing function as the original, single-queue function.

5.3 The Multi-Queue Form of the MomentsP Algorithm

The equations of the multi-queue MomentsP algorithm appear similar to those of the single-queue MomentsP algorithm. The computational difficulty, however will be much greater, since finding $\mathbf{q(P_t)}$ and its first and second derivatives no longer involves solving a sequence of one-variable equations. Instead, we have to solve systems of simultaneous nonlinear equations, or estimate the values of $\mathbf{q(P_t)}$ numerically. Consequently, the multi-queue MomentsP algorithm will be preferable to the multi-queue MomentsQ algorithm when the work required to calculate all the first and second derivatives of $\mathbf{q(P_t)}$ is less than the work required to solve a sequence of systems of nonlinear equations.

As with the MomentsQ algorithm, we find a class of common multi-queue control rules for which MomentsP is easy to use. Rules in this class, *rules with separable inverse functions*, have a special structure making it simple to calculate the first and second derivatives of $\mathbf{q(P_t)}$.

We begin this section by deriving the equations of the multi-queue MomentsP algorithm. We then show how this algorithm becomes simple to use for control rules with separable inverse functions.

5.3.1 Algorithm Development

Step 1: Calculation of E(P). By Proposition 1, $\mathbf{E(P)} = \left(\mathbf{I} - \Phi\right)^{-1}\mu$.

Step 2: Estimate of var(P). The power-series equation of the first-order estimates of var(**P**) is the same as it was in the single-station case:

$$\text{var}(\mathbf{P}) \approx \sum_{s=0}^{\infty} \mathbf{B}^s \Psi^{-1} \Sigma \Psi^{-1} \mathbf{B}'^s, \text{where } \mathbf{B} = \mathbf{I} - \Psi^{-1} + \Psi^{-1}\Phi. \tag{70}$$

Note that Ψ is no longer a diagonal matrix; it is now a general matrix such that $\Psi_{ij} = \partial q_i(\overline{P})/\partial P_j$. This expression follows from the fact that the first-order expansion of $q_i(\mathbf{P}) \approx q_i(\overline{\mathbf{P}}) + \sum_j (P_j - \overline{P}_j) \cdot \partial q_i(\overline{\mathbf{P}})/\partial P_j$, for all stations j. Then, writing all of the first-order expansions for all $q_i(\mathbf{P})$ functions simultaneously yields $\mathbf{q}(\mathbf{P_t}) \approx \mathbf{q}(\overline{\mathbf{P}}) + \Psi \cdot (\mathbf{P_t} - \overline{\mathbf{P}})$, where the elements of Ψ are those previously described. Finding Ψ will require calculating up to n^2 partial derivatives of the inverse production functions.

Step 3: Estimate of E(Q). A second-order estimate for the expected queue length at each station i is given by:

$$E(Q_i) = q_i(\overline{\mathbf{P}}) + \frac{1}{2} \cdot \mathbf{e} \cdot \left(\text{var}(\mathbf{P}) \bullet \nabla^2 q_i(\overline{\mathbf{P}})\right) \cdot \mathbf{e}^{\mathrm{T}}, \tag{71}$$

where $\nabla^2 q_i(\overline{\mathbf{P}})$ is the Hessian matrix of $q_i(\overline{\mathbf{P}})$.

This step may be where the general algorithm becomes intractable, as the estimate for E(**Q**) may require the calculation of up to n^3 partial second derivatives (n^2 partial second derivatives for each inverse production function.) Alternately, if a high degree of accuracy is not required, one can use the first-order estimates of E(**Q**), which only requires calculating the inverse production functions for each queue.

Step 4: Estimate of var(Q). We use the procedure of Proposition 4 to calculate var(Q):

$$\text{var}(\mathbf{Q}) \approx \sum_{s=0}^{\infty} \mathbf{B}^s \Sigma \mathbf{B}'^s, \text{where } \mathbf{B} = \mathbf{I} + (\Phi - \mathbf{I}) \cdot \Pi. \tag{72}$$

As with the calculation of var(**P**), the matrix Π must be modified to account for the fact that $p_i(\mathbf{Q})$ is now a function of multiple queue lengths. The modified Π has entries:

$$\Pi_{ij} = \frac{\partial p_i(\overline{\mathbf{Q}})}{\partial Q_j}. \tag{73}$$

These entries come from the fact that, with production functions of more than one queue, the first-order expansion of $\mathbf{p}(\mathbf{Q}_t)$ is $\mathbf{p}(\mathbf{Q}_t) \approx \mathbf{\Pi} \cdot (\mathbf{Q_t} - \overline{\mathbf{Q}})$, where Π has the entries given above.

5.3.2 The MomentsP Algorithm and Separable Inverse Production Functions

The MomentsP algorithm can be quite useful on the class of multi-queue production functions whose *inverse production functions* are separable. This class has the form:

$$P_{it} = p_i\left(\sum_j k_{ij} Q_{it}\right), \text{ for all stations } i. \tag{74}$$

Assume that **K**, the matrix of k_{ij} entries, is invertible. Then when inverted, (74) becomes:

$$\begin{aligned} q_i(P_{it}) &= \sum_j k_{ij} Q_{jt} \\ \Rightarrow Q_{jt} &= \sum_i k_{ij}^{-1} q_i(P_{it}), \end{aligned} \tag{75}$$

where k_{ij}^{-1} is the i,j element of $\mathbf{K}^{-1}$. This formulation makes it easy to calculate the vectors and matrices needed to run the analytic algorithm. Recall that to use MomentsP, we:

1. Calculate $\mathbf{E(P)} = (\mathbf{I} - \Phi)^{-1}\mu$.
2. Estimate var(**P**), which requires calculating all of the first partial derivatives of the inverse production functions.
3. Estimate $E(Q_i)$ for all stations i, which requires calculating the inverse production functions and all of the second partial derivatives of the inverse production functions.
4. Finally, estimate var(**Q**), which requires calculating the first partial derivatives of all of the production functions.

Thus, we need to calculate the inverse production functions, along with their partial first and second derivatives. Equation (75) shows the inverse production functions. The first partial derivatives are:

$$\frac{\partial q_j(\overline{\mathbf{P}})}{\partial P_i} = k_{ij}^{-1} q_i'(\overline{P}_i), \tag{76}$$

and the second partial derivatives are:

$$\begin{aligned} \frac{\partial^2 q_j(\overline{\mathbf{P}})}{\partial P_i \partial P_k} &= k_{ij}^{-1} q_i''(\overline{P}_i), i = k; \\ &= 0, \text{otherwise}. \end{aligned} \tag{77}$$

Finally, we calculate the first partial derivatives of the production functions. These are also simple to calculate:

$$\frac{\partial p_i(\overline{\mathbf{Q}})}{\partial Q_j} = k_{ij} p_i'\left(\sum_j k_{ij}\overline{Q}_i\right) \text{ for all stations } i \tag{78}$$

Thus, it is comparatively easy to use the MomentsP algorithm on production rules with separable inverse functions.

Example. Consider a general form of a clearing function whose inverse production function is separable:

$$P_{it} = p_i(\mathbf{Q}_t) = \frac{M_i\left(\sum_j k_{ij}Q_j\right)}{\beta_i + \left(\sum_j k_{ij}Q_j\right)}. \tag{79}$$

The inverse production functions are:

$$Q_{jt} = q_j(\mathbf{P}_t) = \sum_i k_{ij}^{-1}\frac{\beta_i P_{it}}{M_i - P_{it}}, \forall j, \tag{80}$$

where k_{ij}^{-1} is the i,j element of $\mathbf{K}^{-1}$, and where $\mathbf{K}$ is the matrix of k_{ij} entries. The first derivatives of the inverse production functions are:

$$\frac{\partial q_j(\overline{\mathbf{P}})}{\partial P_i} = \frac{M_i k_{ij}^{-1}\beta_i}{(M_i - \overline{P}_i)^2}, \forall i, j \tag{81}$$

The second partial derivatives are:

$$\begin{aligned}\frac{\partial^2 q_j(P)}{\partial P_i \partial P_k} &= \frac{2M_i k_{ij}^{-1}\beta_i}{(M_i - P_i)^3}, \text{if } i = k; \\ &= 0, \text{otherwise.}\end{aligned} \tag{82}$$

Finally, the first partial derivatives of the production functions are:

$$\frac{\partial p_i(\mathbf{Q})}{\partial Q_j} = \frac{M_i k_{ij}\beta_i}{\left(\beta_i + \sum_j k_{ij}Q_j\right)^2}, \forall i, j. \tag{83}$$

It is almost as simple to process this alternate generalized form a clearing function as the original, single-queue form.

6. An Asymptotic Lower Bound on Expected Queue Lengths

So far, the approximations considered in this chapter have not provided any guarantees on how close the approximations come to the actual moments. We conclude this chapter with an asymptotic lower bound

on the expected queue length at all stations in a network, provided that the production function used at all stations is concave. Thus, the bound will apply to most general production functions seen in practice, such as the Clearing Functions. The bound also becomes tight as the standard deviation of production approaches zero. Finally, the lower bound is an expression we have seen before – it is the first-order estimate for the expected queue length.

Define $\mathbf{p}(\mathbf{Q}_t)$ to be a vector-valued function whose ith returned variable is $p_i(\mathbf{Q}_t)$. Then we have the following proposition to calculate the lower bound.

Proposition 2. Consider a network in which every station's production rule is concave, monotonically increasing, and has partial derivatives less than one when $\mathbf{Q}_t > 0$. Then a lower bound on the steady-state expected queue lengths, $\overline{\mathbf{Q}}$, is the solution to the equation $\mathbf{p}(\mathbf{Q}) = \overline{\mathbf{P}}$.

Proof. Consider the following nonlinear program, *EQ-Opt*:

$$\min \text{ or } \max \left(\lim_{T\to\infty} \sum_{t=1}^{T} \frac{1}{T} \|\mathbf{Q}_t\|_2 \right) \qquad (a)$$

$$\text{subject to } \mathbf{P}_t \le \mathbf{p}(\mathbf{Q}_t), t = 1 \ldots T \qquad (b) \quad (84)$$

$$\mathbf{Q}_t = \mathbf{Q}_{t-1} + (\Phi - \mathbf{I})\mathbf{P}_{t-1} + \mu, t = 2 \ldots T \qquad (c)$$

$$\mathbf{Q}_t \ge 0, t = 1 \ldots T. \qquad (d)$$

$$\mathbf{P}_t \ge 0, t = 1 \ldots T \qquad (e)$$

We show that $\mathbf{Q}_t = E(\mathbf{Q})$, and $\mathbf{P}_t = E(\mathbf{P})$, for all t, is a feasible solution of *EQ-Opt*. We first show that E(**Q**) and E(**P**) obey constraint (*b*). By assumption the production functions are concave, and a concave function f has the property that $E(f(\mathbf{X})) \le f(E(\mathbf{X}))$. This property implies that $\overline{\mathbf{P}} \le \mathbf{p}(\overline{\mathbf{Q}})$, since $\overline{\mathbf{P}} = E(\mathbf{p}(\mathbf{Q}))$.

Next, E(**Q**) and E(**P**) obey constraint (*c*), since (*c*) is just the expectation of the inventory balance equation, $\mathbf{Q}_t = \mathbf{Q}_{t-1} + (\Phi - \mathbf{I})\mathbf{P}_{t-1} + \varepsilon_t$.

Finally, E(**Q**) obeys (*d*) since we assume that negative work-in-queue levels are not allowed, $E(\mathbf{Q}) > \mathbf{0}$. Similarly, E(P) obeys (*e*), since negative production levels are not allowed.

Since E(**Q**) and E(**P**) comprise a feasible solution, the optimal solution of *EQ-Opt* implies a lower bound for the steady-state E(**Q**) if the objective (*a*) is minimized, and an upper bound for E(**Q**) if (*a*) is maximized. The bound created is:

$$E(\mathbf{Q}^*) = \lim_{T\to\infty} \sum_{t=1}^{T} \frac{\mathbf{Q}_t^*}{T}, \tag{85}$$

where $\mathbf{Q}_t^*$, for all t, is the optimal solution to *EQ-Opt*.

The program *EQ-Opt* is unbounded if it is maximized, since we can quickly verify that $\mathbf{P}_t = 0$ and $\mathbf{Q}_t$ arbitrarily large for all t is a feasible solution.

This leaves the minimization of *EQ-Opt*, which we do by term-by-term minimization. Since the objective function (*a*) is a summation of norms of the $\mathbf{Q}_t$'s, it will be minimized by making each $\mathbf{Q}_t$ as small as possible. Thus, we begin by setting $\mathbf{Q}_1 = \mathbf{0}$. We next consider the successive $\mathbf{Q}_t$'s. Consider constraint (*c*), which dictates $\mathbf{Q}_t$ as a function $\mathbf{Q}_{t-1}$ and $\mathbf{P}_{t-1}$. To minimize $\mathbf{Q}_t$, we need to make the quantity added to $\mathbf{Q}_{t-1}$, $(\mathbf{\Phi} - \mathbf{I})\mathbf{P}_{t-1}$, as negative as possible. Now, $\mathbf{\Phi}$ is a MR-model workflow matrix, so we assume that $\mathbf{\Phi}$ is a positive matrix with a spectral radius less than one. Then, $(\mathbf{\Phi} - \mathbf{I})\mathbf{P}_{t-1}$ is most negative when $\mathbf{P}_{t-1}$ is as great as possible. Constraint (*b*) dictates that $\mathbf{P}_t \le \mathbf{p}(\mathbf{Q}_t), \forall t$, so $\mathbf{P}_{t-1}$ is maximized by setting $\mathbf{P}_t = \mathbf{p}(\mathbf{Q}_t), \forall t$. Substituting this value for $\mathbf{P}_t$ into constraint (*c*) yields a recursion equation for every $\mathbf{Q}_t$ that minimizes *EQ-Opt*:

$$\begin{aligned} &\mathbf{Q}_1^* = \mathbf{0}, \\ &\mathbf{Q}_t^* = \mathbf{Q}_{t-1}^* + (\mathbf{\Phi} - \mathbf{I}) \cdot \mathbf{p}(\mathbf{Q}_{t-1}^*) + \mu, t = 2 \ldots T. \end{aligned} \tag{86}$$

In steady-state, this recursion equation becomes:

$$\begin{aligned} &(\mathbf{I} - \mathbf{\Phi}) \cdot \mathbf{p}(\mathbf{Q}^*) = \boldsymbol{\mu} \\ &\Rightarrow \mathbf{p}(\mathbf{Q}^*) = (\mathbf{I} - \mathbf{\Phi})^{-1} \boldsymbol{\mu} = E(\mathbf{P}) \end{aligned} \tag{87}$$

which is the system of equations that must be solved to yield a first-order estimate of E(**Q**). It remains to show that $\mathbf{Q}_t^* \to \mathbf{Q}^*$; if so, $\mathbf{Q}^*$ will be the desired lower bound on E(**Q**). We show a stricter condition in the following lemma, the proof of which is given in the appendix at the end of this section.

Lemma. $\mathbf{Q}_t^* \to \mathbf{Q}^*$, where $\mathbf{Q}^*$ is the solution to the equation $\mathbf{p}(\mathbf{Q}^*) = \overline{\mathbf{P}}$. Further, $\mathbf{0} = \mathbf{Q}_1^* \le \mathbf{Q}_t^* \le \mathbf{Q}^*$ for all t.

Since $\mathbf{Q}_t^* \to \mathbf{Q}^*$, and $\mathbf{0} = \mathbf{Q}_1^* \le \mathbf{Q}_t^* \le \mathbf{Q}^*$, we have that the sequence of $\mathbf{Q}_t^*$'s both converges and is bounded. Thus, we immediately have:

$$\lim_{T\to\infty}\sum_{t=1}^{T}\frac{\mathbf{Q}_t^*}{T}=\mathbf{Q}^*, \tag{88}$$

where $\mathbf{Q}^*$ is the solution to the system of equations $\mathbf{p}(\mathbf{Q})=E(\mathbf{P})$. This proves the proposition. □

The production rules covered by this proposition are likely to be those most often used in practice. Presumably, most rules will feature production increases in response to increasing queue lengths, but that the rate of these increases diminishes; these conditions will create production rules that are monotonically increasing and concave. Further, with a monotonically increasing and concave function, the condition that the partial derivatives be less than one simply ensures that a station will not produce more work than it has in its queue.

Next, we note that this lower bound is tight. If the variance of production at all stations is 0 (implying the arrival streams to all stations have a variance of 0), by definition the (now deterministic) steady-state queue lengths at all stations will be the solution of $\mathbf{p}(\overline{\mathbf{Q}})=\overline{\mathbf{P}}$.

An important open question would be to find a valid upper bound on E(**Q**). Unfortunately, finding such an upper bound would likely be much more difficult than finding the lower bound. At the least, the upper bound would depend on the input and / or production variances, since it can be shown that the an infinite arrival variance corresponds to an infinite expected queue length. Consequently, the approach used in Proposition 2 (which only uses expected queue lengths and production quantities) could not be applied directly.

Appendix: Proof of the Convergence Lemma in Proposition 2

Lemma. $\mathbf{Q}_t^*\to\mathbf{Q}^*$, where $\mathbf{Q}^*$ is the solution to the equation $\mathbf{p}(\mathbf{Q}^*)=\overline{\mathbf{P}}$. Further, $\mathbf{0}=\mathbf{Q}_1^*\le\mathbf{Q}_t^*\le\mathbf{Q}^*$ for all t.

Proof. We compare the difference between $\mathbf{Q}^*-\mathbf{Q}_t^*$ and $\mathbf{Q}^*-\mathbf{Q}_{t-1}^*$:

$$\begin{aligned}\frac{\mathbf{Q}^*-\mathbf{Q}_t^*}{\mathbf{Q}^*-\mathbf{Q}_{t-1}^*}&=\frac{\mathbf{Q}^*-\left(\mathbf{Q}_{t-1}^*+(\Phi-\mathbf{I})\mathbf{p}(\mathbf{Q}_{t-1}^*)+\mu\right)}{\mathbf{Q}^*-\mathbf{Q}_{t-1}^*}\\&=1-\frac{(\Phi-\mathbf{I})\mathbf{p}(\mathbf{Q}_{t-1}^*)+\mu}{\mathbf{Q}^*-\mathbf{Q}_{t-1}^*}.\end{aligned} \tag{89}$$

To show convergence and that $\mathbf{0}=\mathbf{Q}_1^*\le\mathbf{Q}_t^*\le\mathbf{Q}^*$, the right hand side of (89) must be between 0 and 1. This is equivalent to showing:

$$\left(\mathbf{Q}^*-\mathbf{Q}_{t-1}^*\right)>(\Phi-\mathbf{I})\mathbf{p}(\mathbf{Q}_{t-1}^*)+\boldsymbol{\mu},\text{ for all }t\text{, and} \tag{90}$$

$$(\Phi-\mathbf{I})\mathbf{p}(\mathbf{Q}_{t-1}^*)+\boldsymbol{\mu}>\mathbf{0},\text{ for all }t. \tag{91}$$

We show that (90) and (91) hold through an inductive proof.

Invariant: this proof uses the invariant condition $\mathbf{Q}^* - \mathbf{Q}_{t-1}^* > 0$.

Base case: for $\mathbf{Q}_1^* = \mathbf{0}$, we have that $\mathbf{Q}^* - \mathbf{Q}_1^* > 0$. Then (90) holds since:

$$\begin{aligned} &(\mathbf{Q}^* - \mathbf{Q}_1^*) > (\Phi - \mathbf{I})\mathbf{p}(\mathbf{Q}_1^*) + \boldsymbol{\mu} \\ &\Rightarrow \mathbf{Q}^* > \boldsymbol{\mu}. \end{aligned} \tag{92}$$

Further, (91) holds since:

$$(\Phi - I)\mathbf{p}(\mathbf{Q}_1^*) + \mu = \mu > \mathbf{0}. \tag{93}$$

Induction step: assume that $\mathbf{Q}^* - \mathbf{Q}_{t-1}^* > 0$.

We first determine whether (91) holds. The invariant condition, the monoticity of the production functions, and the fact that Φ has a spectral radius less than one requires that :

$$(\Phi - \mathbf{I})\mathbf{p}(\mathbf{Q}_{t-1}^*) > (\Phi - \mathbf{I})\mathbf{p}(\mathbf{Q}^*). \tag{94}$$

Further, from (86), $(\Phi - \mathbf{I})\mathbf{p}(\mathbf{Q}^*) + \mu = \mathbf{0}$. These two facts imply that $(\Phi - \mathbf{I})\mathbf{p}(\mathbf{Q}_{t-1}^*) + \mu > \mathbf{0}$, so (91) holds.

We now consider whether (90) holds. By assumption, $\mathbf{Q}^* - \mathbf{Q}_{t-1}^* > 0$. Define $\boldsymbol{\beta}_{t-1}$ to be the nonnegative vector such that $\mathbf{Q}^* = \mathbf{Q}_{t-1}^* + \boldsymbol{\beta}_{t-1}$. Substituting, we rewrite (90) to be:

$$\begin{aligned} &(\mathbf{Q}^* - (\mathbf{Q}^* - \boldsymbol{\beta}_{t-1})) > (\Phi - \mathbf{I})\mathbf{p}(\mathbf{Q}^* - \boldsymbol{\beta}_{t-1}) + \mu \\ &\Rightarrow \boldsymbol{\beta}_{t-1} > (\Phi - \mathbf{I})\mathbf{p}(\mathbf{Q}^* - \boldsymbol{\beta}_{t-1}) + \mu. \end{aligned} \tag{95}$$

By assumption, the partial derivatives of $\mathbf{p}(\mathbf{Q}_t^*)$ are less than one, which implies that:

$$\mathbf{p}(\mathbf{Q}^*) - \boldsymbol{\beta}_{t-1} < \mathbf{p}(\mathbf{Q}^* - \boldsymbol{\beta}_{t-1}). \tag{96}$$

Now, consider the following:

$$\begin{aligned} &\boldsymbol{\beta}_{t-1} > (\Phi - \mathbf{I})(\mathbf{p}(\mathbf{Q}^*) - \boldsymbol{\beta}_{t-1}) + \mu \\ &\Rightarrow \boldsymbol{\beta}_{t-1} > (\mathbf{I} - \Phi)\boldsymbol{\beta}_{t-1} + (\Phi - \mathbf{I})\mathbf{p}(\mathbf{Q}^*) + \mu \\ &\Rightarrow \boldsymbol{\beta}_{t-1} > (\mathbf{I} - \Phi)\boldsymbol{\beta}_{t-1}. \end{aligned} \tag{97}$$

The transition from the second to third step of (97) comes from the fact that $(\Phi - \mathbf{I})\mathbf{p}(\mathbf{Q}^*) + \mu = \mathbf{0}$. The final inequality of (97) holds since $\Phi > \mathbf{0}$ and $\boldsymbol{\beta}_{t-1} \geq \mathbf{0}$. But then, (97), combined with inequality (96) implies that that (95) holds. Then (90) holds, as desired.

Finally, to complete the induction, we need to show that the invariant will hold for the next iteration, i.e. that $\mathbf{Q}^* - \mathbf{Q}_t^* > 0$:

$$\begin{aligned} &= (\mathbf{Q}^* - \mathbf{Q}_t^*) = (\mathbf{Q}^* - \mathbf{Q}_{t-1}^*) - ((\Phi - \mathbf{I})\mathbf{p}(\mathbf{Q}_{t-1}^*) + \mu) \\ &\Rightarrow (\mathbf{Q}^* - \mathbf{Q}_t^*) \geq 0, \end{aligned} \tag{98}$$

since condition (90) implies that $(\mathbf{Q}^* - \mathbf{Q}_{t-1}^*) > (\Phi - \mathbf{I})\mathbf{p}(\mathbf{Q}_{t-1}^*) + \mu$.

Thus, we have proven that $\mathbf{Q}_t^* \to \mathbf{Q}^*$, and $\mathbf{0} = \mathbf{Q}_1^* \leq \mathbf{Q}_t^* \leq \mathbf{Q}^*$ for all t. □

Chapter 4: Models That Maintain a Constant Inventory (Class LLC-MR)

1. Introduction

In previous chapters, we have assumed that new work arrivals to job shops come from distributions independent from the amount of work in the job shop. In this chapter, we consider *constant inventory* models of job shops. In these models, there is a set of stations whose inputs we control; we adjust these inputs to keep constant the weighted inventory of the shop.

Keeping the inventory in a shop constant is a much-used strategy to control inventory and work-in-progress levels, along with production variability, and has been studied in detail. Recent work on constant work-in-progress rules includes that of Spearman, Hopp and Woodruff (1989), Duenyas, Hopp and Spearman (1993), Duenyas and Hopp (1993), Duenyas, Hopp and Spearman (1993), Hopp and Spearman (1996), Gstettner and Kuhn (1996), and Herer and Masin (1997).

For the sake of simplicity, the models we will study in this chapter will be identical to the Tactical Planning Model (TPM) with the exception of the new-work arrival processes. Thus, we assume that stations process a fixed fraction of the work in their queues each time period, or $P_{it} = a_i Q_{it}$, and that arrivals from upstream stations are linear combinations of the work produced upstream plus a noise term. We will show how to extend the TPM to keep the shop's work-in-progress constant, and how this extension significantly reduces the variability of the shop's production. Section 2 of this paper quickly reviews the original TPM's moment equations. Section 3 expands on the TPM equations to derive the constant-inventory tactical planning model. Section 4 discusses how the analyst can set inventory weights and initial inventories to optimize the performance of the job shop. Finally, section 5 tests the constant-inventory model on an example job shop, which demonstrates the advantages of the model over both the TPM and a constant-release model where we keep the input to a station constant. Further, the example also shows the importance of using the optimal weighting vector – we find that using a simple weight vector (for example, all weights equal one) may result in very poor performance.

2. Review of the Tactical Planning Model

Recall that the Tactical Planning Model (TPM) uses the simplest possible linear control rule to determine the amount of work to perform each period. The rule is:

$$P_{it} = a_i Q_{it}, \tag{1}$$

where P_{it} is the production of work station i in time period t, Q_{it} is the work-in-process or work-in-queue at the start of period t, and the parameter $a_i, 0 < a_i \leq 1$, is a smoothing parameter.

The characterization of work arrivals is also simple. A workstation receives two types of arrivals. The first type comprises new jobs that have their first processing step at the station. The second type comprises jobs in process that have just completed processing at an upstream station.

We model the arrivals to a station from another station by the following equation:

$$A_{ijt} = \phi_{ij} P_{j,t-1} + \varepsilon_{ijt} . \tag{2}$$

In this equation, A_{ijt} is the amount of work arriving to station i from station j at the start of period t, ϕ_{ij} is a positive scalar, and ε_{ijt} is random variable. Thus, we assume that one unit (e.g. hour) of work at station j generates ϕ_{ij} time units of work at station i, on average. The term ε_{ijt} is a noise term that introduces uncertainty into the relationship between production at j and arrivals to i; we assume this term is a serially i. i. d. random variable with zero mean and a known variance.

Then, the arrival stream to station i is given by the following:

$$A_{it} = \sum_j A_{ijt} + N_{it} , \tag{3}$$

where N_{it} is an i. i. d. random variable for the work load from new jobs that enter the shop at station i and at time t. Substituting for A_{ijt}, we find:

$$A_{it} = \sum_j \phi_{ij} P_{j,t-1} + \varepsilon_{it} , \text{ where } \varepsilon_{it} = N_{it} + \sum_j \varepsilon_{ijt} . \tag{4}$$

Note that ε_{it} represents the work arrivals not predictable from the production levels of the previous period, and includes work from new jobs and noise in the flow. By assumption, the time series ε_{it} is independent and identically distributed over time.

Next, substituting the production rules and the characterization of work arrivals into a standard inventory balance equation yields the following recursion equation:

$$\mathbf{P}_t = (\mathbf{I} - \mathbf{D} + \mathbf{D}\Phi)\mathbf{P}_{t-1} + \mathbf{D}\varepsilon_t . \tag{5}$$

Here, $\mathbf{P}_t = \{P_{1t}, \ldots, P_{nt}\}'$ and $\varepsilon_t = \{\varepsilon_{1t}, \ldots, \varepsilon_{nt}\}'$ are column vectors of random variables, n is the number of workstations, $\mathbf{I}$ is the identity matrix, $\mathbf{D}$ is a diagonal matrix with $\{a_1, \ldots, a_n\}$ on the diagonal, and Φ is an n-by-n matrix with elements ϕ_{ij}. By iterating the equation and assuming an infinite history of the system, we rewrite the above equation as the following infinite series:

$$\mathbf{P}_t = \sum_{s=0}^{\infty} (\mathbf{I} - \mathbf{D} + \mathbf{D}\Phi)^s \mathbf{D}\varepsilon_{t-s} . \tag{6}$$

To calculate the moments of the production random vector P_t, we denote the mean and the covariance for the noise vector ε_t by $\mu = \{\mu_1, \ldots, \mu_n\}'$, and by $\Sigma = \{\sigma_{ij}\}$, respectively. Then, the expectation of the production vector is given by:

$$\begin{aligned} E[\mathbf{P}_t] &= \sum_{s=0}^{\infty} (\mathbf{I} - \mathbf{D} + \mathbf{D}\Phi)^s \mathbf{D}\mu, \\ &= (I - \Phi)^{-1}\mu, \end{aligned} \tag{7}$$

provided that the spectral radius of Φ is less than one (see Graves 1986).

The covariance matrix of production, **S**, is given by:

$$\begin{aligned} \mathbf{S} = \mathrm{var}(\mathbf{P}_t) &= \sum_{s=0}^{\infty} \mathbf{B}^s \mathbf{D}\Sigma\mathbf{D}\mathbf{B}'^s, \\ \text{where } \mathbf{B} &= \mathbf{I} - \mathbf{D} + \mathbf{D}\Phi. \end{aligned} \tag{8}$$

Again, this infinite series converges if the spectral radius of Φ is less than one.

3. The Constant Inventory Model

In this section, we extend the TPM equations to create the *Constant-Inventory Tactical Planning Model* (CI-TPM). This model requires that the weighted inventory of the shop is equal to some constant, W. Mathematically, this constraint is:

$$\sum_i w_i Q_{it} = W, \forall t\,. \tag{9}$$

Here, w_i is a non-negative weight for the inventory or queue at station i. As the work-in-queue Q_{it} is work load at station i, the weights would reflect how to combine workloads at different stations into a common measure of inventory or workload in the shop.

The constraint (9) provides a very general mechanism for modeling workload-regulating control polices such as CONWIP (c.f. Hopp and Spearman, 1996) or Drum-Buffer-Rope (c.f. Goldratt, 1986). The constraint assures that the weighted workload in the shop remains constant, where we only require that the weights be nonnegative. We can model different polices by the choice of weights. For instance, if w_i is the number of jobs per unit of work at station i, then we are regulating the number of jobs in the shop. Alternatively, if w_i is the amount of remaining work for the shop bottleneck(s) per unit of work at station i, then we are regulating the work load upstream of the bottleneck(s). Similarly we can regulate the amount of work at or destined to any subset of the stages in the shop.

We can also use the above constraint to set constraints on production. Since we assume a linear control rule (1) for setting production, we can use (9) to model any linear constraint on the production rates, e. g.,

$$\sum_i v_i P_{it} = V, \forall t . \tag{10}$$

For instance, if the shop had a single bottleneck, we might impose a simple constraint that specifies the bottleneck's production rate and then use the model to characterize the work flow throughout the shop.

To adapt the TPM to include the constraint (9), we assume that the initial work-in-queue Q_{i0} satisfies (9). Then to assure (9) for all time periods, we must have that

$$\sum_i w_i \left(Q_{it} - Q_{i,t-1}\right) = 0, \forall t . \tag{11}$$

The difference between Q_{it} and $Q_{i,t-1}$ equals arrivals of new work less completed work, i.e. $Q_{it} - Q_{i,t-1} = A_{it} - P_{it}$. Then we can rewrite the above equation as:

$$\sum_i w_i \left(A_{it} - P_{i,t-1}\right) = 0, \forall t \tag{12}$$

In order to achieve constant inventory, as specified above, we need to assume that we can control the release of work to the shop in some way. In particular, we assume the following:

- In each period we have uncontrolled and controlled releases. As with the TPM, the uncontrolled releases are given by N_{it}, an i. i. d. random variable for the work load from new jobs that enter the shop at station i and time t. In addition, to assure that (9) holds, we release additional jobs; define R_{it} to be the work load from these jobs that enter the shop at station i and time t. We allow R_{it} to be negative, denoting that we would remove work from the uncontrolled releases, or even from the shop floor. Nevertheless, we assume that, on average, each station receives work each time period, so that work removals are infrequent. We also assume that work queues are generally large enough that removing work will not require us to make a station's work-in-progress negative.
- We assume that each period we release r_t units of work to the shop, where work r_t is set to assure that (9) holds. Each station i receives a fixed fraction β_i of the controlled-release work load r_t. For example, one station would always receive 70% of the work load, and another station would always receive 30%. Thus, we have that each station i receives $R_{it} = \beta_i r_t$. Note that there are no controlled releases to stations with $\beta_i = 0$.

- In order to have a controllable system, we require that $\mathbf{w}'\boldsymbol{\beta} > 0$, where $\mathbf{w} = \{w_1, \ldots, w_n\}$, and $\boldsymbol{\beta} = \{\beta_1, \ldots, \beta_n\}$. If this is not true, then we cannot assure that (9) holds because we can only release work to stations that appear in the constraint with zero weights.

Now we write the arrivals to each station as:

$$A_{it} = \sum_j \phi_{ij} P_{j,t-1} + N_{it} + \sum_j \varepsilon_{ijt} + R_{it}, i = 1,\ldots,N \tag{13}$$

Here, R_{it} is the work that arrives to station i at time t to help maintain constant inventory. Substituting (13) into (12), we find:

$$\sum_{i=1}^{N} w_i \left(\sum_j \phi_{ij} P_{j,t-1} + N_{it} + \sum_j \varepsilon_{ijt} + R_{it} - P_{i,t-1} \right) = 0, \forall t\,. \tag{14}$$

Solving for the sum of the R_{it}'s, we find:

$$\sum_{i=1}^{N} w_i R_{it} = \sum_{i=1}^{N} w_i \left(P_{i,t-1} - \sum_j \phi_{ij} P_{j,t-1} - N_{it} - \sum_j \varepsilon_{ijt} \right), \forall t\,. \tag{15}$$

We can rewrite (15) in matrix-vector form as follows:

$$\mathbf{w}'\mathbf{R}_t = \mathbf{w}'\left((\mathbf{I} - \boldsymbol{\Phi})\mathbf{P}_{t-1} - \boldsymbol{\varepsilon}_t\right). \tag{16}$$

Here, $\mathbf{w} = \{w_1, \ldots, w_n\}$, $\mathbf{R}_t = \{R_{1t}, \ldots, R_{nt}\}$, and the other vectors and matrices are the same as in the TPM model. From our previous discussion of inputs to the controlled stations,

$$R_t = r_t \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix} = r_t \boldsymbol{\beta}\,, \tag{17}$$

where r_t, a scalar, is the amount of work that arrives to the shop to maintain constant inventory, and the β_i's are "templates" for the constant-inventory releases. Then,

$$\mathbf{w}'\mathbf{R}_t = r_t(\mathbf{w}'\boldsymbol{\beta}) = \mathbf{w}'\left((\mathbf{I} - \boldsymbol{\Phi})\mathbf{P}_{t-1} - \boldsymbol{\varepsilon}_t\right). \tag{18}$$

Without loss of generality, suppose that we choose $\mathbf{w}$ and β such that $\mathbf{w}'\boldsymbol{\beta} = 1$. We can do this since $\mathbf{w}'\boldsymbol{\beta} > 0$. Then,

$$\begin{aligned} r_t &= \mathbf{w}'\big((\mathbf{I}-\boldsymbol{\Phi})\mathbf{P}_{t-1} - \boldsymbol{\varepsilon}_t\big), \text{ and} \\ \mathbf{R}_t &= \boldsymbol{\beta}\mathbf{w}'\big((\mathbf{I}-\boldsymbol{\Phi})\mathbf{P}_{t-1} - \boldsymbol{\varepsilon}_t\big). \end{aligned} \tag{19}$$

Now, if we write (13) in matrix form, we find:

$$\begin{aligned} \mathbf{A}_t &= \boldsymbol{\Phi}\mathbf{P}_{t-1} + \boldsymbol{\varepsilon}_t + \mathbf{R}_t, \\ &= \boldsymbol{\Phi}\mathbf{P}_{t-1} + \boldsymbol{\varepsilon}_t + \boldsymbol{\beta}\mathbf{w}'\big((\mathbf{I}-\boldsymbol{\Phi})\mathbf{P}_{t-1} - \boldsymbol{\varepsilon}_t\big). \end{aligned} \tag{20}$$

As in the TPM, the station productions at time t are given by the following equation:

$$\begin{aligned} \mathbf{P}_t &= (\mathbf{I}-\mathbf{D})\mathbf{P}_{t-1} + \mathbf{D}\mathbf{A}_t, \\ &= (\mathbf{I}-\mathbf{D})\mathbf{P}_{t-1} + \mathbf{D}\big(\boldsymbol{\Phi}\mathbf{P}_{t-1} + \boldsymbol{\varepsilon}_t + \boldsymbol{\beta}\mathbf{w}'\big((\mathbf{I}-\boldsymbol{\Phi})\mathbf{P}_{t-1} - \boldsymbol{\varepsilon}_t\big)\big), \end{aligned} \tag{21}$$

which can be simplified to the following form:

$$\begin{aligned} &\mathbf{P}_t = (\mathbf{I}-\mathbf{D}+\mathbf{D}\mathbf{F})\mathbf{P}_{t-1} + \mathbf{D}\mathbf{G}\boldsymbol{\varepsilon}_t, \text{ where} \\ &\mathbf{F} = \boldsymbol{\Phi} + \boldsymbol{\beta}\mathbf{w}'\big(\mathbf{I}-\boldsymbol{\Phi}\big), \text{ and} \\ &\mathbf{G} = \mathbf{I} - \boldsymbol{\beta}\mathbf{w}'. \end{aligned} \tag{22}$$

Note that (22) has the same form as (5), the recursion equation for the TPM. In comparing (25) to (5), we have replaced Φ by $\mathbf{F}$, and $\boldsymbol{\varepsilon}_t$ by $\mathbf{G}\boldsymbol{\varepsilon}_t$.

We now prove three propositions about the constant inventory model in order to characterize whether the moments of $\mathbf{P}_t$ converge as $t \to \infty$, and to what.

Proposition 1. *If the spectral radius of the work flow matrix Φ is less than one and if the vector of inventory weights are such that $\mathbf{F} \geq \mathbf{0}$, then the spectral radius of the matrix $\mathbf{F}$ is equal to 1; furthermore, the left eigenvector of $\mathbf{F}$ is the vector of inventory weights $\mathbf{w}$.*

Proof. The proof that $\mathbf{w}'\mathbf{F} = \mathbf{w}'$ is immediate from (22) and the assumption that $\mathbf{w}'\boldsymbol{\beta} = 1$.

Since $\mathbf{F}$ is a positive matrix, a result from the Frobenius theory of reducible nonnegative matrices (c.f. Gantmacher, 1959, p. 66) guarantees that $\mathbf{F}$ has a maximum eigenvalue $\lambda_0 \geq \mathbf{0}$, and associated left eigenvector $\mathbf{y}'$ such that $\mathbf{y}'\mathbf{F} = \lambda_0\mathbf{y}'$. From (22), we can rewrite $\mathbf{F}$ as: $\mathbf{F} = \boldsymbol{\Phi} + \boldsymbol{\beta}(\mathbf{w}' - \mathbf{w}'\boldsymbol{\Phi})$. Suppose $\lambda_0 > 1$. We consider three cases:

(i) $\mathbf{y}'\boldsymbol{\beta} = 0$. Then $\mathbf{y}'\mathbf{F} = \mathbf{y}'\boldsymbol{\Phi} + \mathbf{y}'\boldsymbol{\beta}(\mathbf{w}' - \mathbf{w}'\boldsymbol{\Phi}) = \mathbf{y}'\boldsymbol{\Phi} = \lambda_o\mathbf{y}'$. This contradicts the assumption that the spectral radius of the work flow matrix Φ is less than one.

(ii) $\mathbf{y}'\boldsymbol{\beta} > 0$. Without loss of generality, we can re-scale $\mathbf{y}'$ so that $\mathbf{y}'\boldsymbol{\beta} = 1$. Then $\mathbf{y}'\mathbf{F} = \mathbf{y}'\boldsymbol{\Phi} + \mathbf{y}'\boldsymbol{\beta}(\mathbf{w}' - \mathbf{w}'\boldsymbol{\Phi}) = \mathbf{y}'\boldsymbol{\Phi} + (\mathbf{w}' - \mathbf{w}'\boldsymbol{\Phi}) = \lambda_0\mathbf{y}'$, and $\|\lambda_0\mathbf{y}'\| > \|\mathbf{y}'\|$ since $\lambda_0 > 1$. Thus we have $\|(\mathbf{y}' - \mathbf{w}')\boldsymbol{\Phi}\| > \|(\mathbf{y}' - \mathbf{w}')\|$. This contradicts the assumption that the spectral radius of the work flow matrix Φ is less than one.

(iii) $\mathbf{y'\beta} < 0$. Without loss of generality, we can re-scale $\mathbf{y'}$ so that $\mathbf{y'\beta} = -1$. Then $\mathbf{y'F} = \mathbf{y'\Phi} + \mathbf{y'\beta(w'-w'\Phi)} = \mathbf{y'\Phi} - (\mathbf{w'-w'\Phi}) = \lambda_0\mathbf{y'}$, and $\|\lambda_0\mathbf{y'}\| > \|\mathbf{y'}\|$ since $\lambda_0 > 1$. Thus we have $\|(\mathbf{y'+w'})\mathbf{\Phi}\| > \|(\mathbf{y'+w'})\|$. This contradicts the assumption that the spectral radius of the work flow matrix Φ is less than one.

Thus, we cannot have $\lambda_0 > 1$. Since $\mathbf{w'F} = \mathbf{w'}$, we have that the maximal eigenvalue $\lambda_0 = 1$ with associated left eigenvector $\mathbf{w'}$. □

To interpret Proposition 1, we note that $\mathbf{F} \geq \mathbf{0}$ is equivalent to the following:

$$\phi_{ij} + \beta_i\left(w_j - \sum_k w_k\phi_{kj}\right) \geq 0, \forall i, j \tag{23}$$

The left-hand-side of (23) is the expected amount of work that will arrive next period at station i per unit of work completed at station j in the current period. The first component, ϕ_{ij}, represents the direct flow from station j to station i. The second part of (23) corresponds to the amount of work to be released to station i per unit of work completed at station j. A unit of work at station j will reduce the work load constraint directly by w_j. But this unit of work at station j will increase the workload at downstream stations, and thus increase the workload constraint by $\sum_k w_k\phi_{kj}$. Thus, the difference is the impact on the workload constraint; the fraction β_i of this impact will result in new releases to station i.

Thus, the condition that $\mathbf{F} \geq \mathbf{0}$ assumes that for every pair of stations (i,j), the expected amount of work that arrives at station i per unit of work completed at station j is non-negative. In other words, production at station j would not on average reduce the arrivals at station i in the next period. As such, this is seems to be a fairly weak restriction on the choice of weights.

To show that $\mathbf{P_t}$ converges if $\mathbf{F}$'s spectral radius equals one, we rewrite (22) as

$$\mathbf{P}_t = (\mathbf{I} - \mathbf{D} + \mathbf{DF})^t\mathbf{DQ}_0 + \sum_{s=0}^{t}(\mathbf{I} - \mathbf{D} + \mathbf{DF})^s\mathbf{DG}\varepsilon_{t-s} \tag{24}$$

where we substitute $\mathbf{P_0} = \mathbf{DQ_0}$, for $\mathbf{Q_0}$ being the initial work-in-queue. We now prove that the expectation and covariance matrix of $\mathbf{P_t}$ both converge.

Proposition 2. *Suppose we have a job-shop model in which the expression for the steady-state production vector,* $\mathbf{P}$*, is given by the formula:*

$$\mathbf{P} = (\mathbf{I} - \mathbf{D} + \mathbf{DF})^\infty\mathbf{DQ}_0 + \sum_{s=0}^{\infty}(\mathbf{I} - \mathbf{D} + \mathbf{DF})^s\mathbf{DG}\varepsilon_{t-s}\,. \tag{25}$$

Assume that the spectral radius of $\mathbf{F} = 1$*. Then the expectation of* $\mathbf{P}$ *converges to a finite value.*

Proof. Define $\mathbf{B} = (\mathbf{I} - \mathbf{D} + \mathbf{DF})$. **B**'s largest eigenvalue equals the largest eigenvalue of **F** (the argument is identical to that given in Graves, 1986), so the spectral radius of **B** is 1. Further, we see by substitution that $\mathbf{w'D^{-1}}$ is the left eigenvector of **B** corresponding to **B**'s maximal eigenvalue:

$$\begin{aligned} \mathbf{w'D^{-1}B} &= \mathbf{w'D^{-1}(I - D + DF)}, \\ &= \mathbf{w'D^{-1} - w' + w'F}, \\ &= \mathbf{w'D^{-1}}. \end{aligned} \tag{26}$$

Here, we use Proposition 1 for the last step. We then have the following decomposition of $\mathbf{B}^s$:

$$\mathbf{B}^s = 1^s \mathbf{v}_1 \mathbf{w'D^{-1}} + \mathbf{E}^s. \tag{27}$$

Here, 1 is the largest eigenvalue of **B**, $\mathbf{v}_1$ is the corresponding right eigenvector of **B**, $\mathbf{w'D^{-1}}$ is the corresponding left eigenvector of **B**, and **E** is an orthogonal matrix such that $\mathbf{v}_1\mathbf{w'D^{-1}E}^s = \mathbf{0}$. Further, **E** has a spectral radius strictly less than one. By substituting the expression for $\mathbf{B}^s$ into the formula for **P**, and multiplying through terms, we find:

$$\mathbf{P} = \mathbf{B}^\infty \mathbf{DQ}_0 + \sum_{s=0}^{\infty} \left(1^s \mathbf{v}_1 \mathbf{w'D^{-1}DG}\varepsilon_{t-s} + \mathbf{E}^s \mathbf{DG}\varepsilon_{t-s} \right). \tag{28}$$

We know that $\mathbf{B}^\infty\mathbf{DQ}_0$ converges to a finite value, since the spectral radius of **B** is one. This leaves the infinite summation. From (22), we immediately find that $\mathbf{w'G} = \mathbf{0}$, so the first term in the summation is zero. But then, since the spectral radius of **E** is less than one, the second term must converge to a finite value. Consequently, the expectation of **P** converges to the following:

$$\begin{aligned} E[\mathbf{P}] &= \mathbf{B}^\infty \mathbf{DQ}_0 + \sum_{s=0}^{\infty} \mathbf{E}^s \mathbf{DG}\boldsymbol{\mu} \\ &= \mathbf{v}_1 \mathbf{w'Q}_0 + (\mathbf{I} - \mathbf{E})^{-1} \mathbf{DG}\boldsymbol{\mu}, \end{aligned} \tag{29}$$

where $\boldsymbol{\mu} = \{\mu_1, \ldots, \mu_n\}'$ is the expectation of the noise vector ε_t. □

Proposition 3. *Suppose we have a job-shop model in which the expression for the steady-state production vector,* **P**, *is given by the formula:*

$$\mathbf{P} = (\mathbf{I} - \mathbf{D} + \mathbf{DF})^\infty \mathbf{DQ}_0 + \sum_{s=0}^{\infty} (\mathbf{I} - \mathbf{D} + \mathbf{DF})^s \mathbf{DG}\varepsilon_{t-s}. \tag{30}$$

Assume the spectral radius of $\mathbf{F} = 1$. *Then the covariance matrix of* **P** *converges to a finite value.*

Proof. Define $\mathbf{B} = (\mathbf{I} - \mathbf{D} + \mathbf{DF})$. The covariance matrix for **P** is denoted as **S** and given by:

$$\begin{aligned} \mathbf{S} &= \text{var}\left[\sum_{s=0}^{\infty} \mathbf{B}^s \mathbf{DG}\varepsilon_{t-s} \right], \\ &= \text{var}\left[\sum_{s=0}^{\infty} \mathbf{E}^s \mathbf{DG}\varepsilon_{t-s} \right], \text{ as shown in Proposition 2,} \\ &= \sum_{s=0}^{\infty} \mathbf{E}^s \mathbf{DG}\Sigma\mathbf{G'DE'}^s. \end{aligned} \tag{31}$$

Here, Σ is the covariance matrix of the noise vector. This summation converges, as the maximal eigenvalue of **E** is less than one, as shown in Proposition 2. □

Note that it is possible to generate a set of weights where the resulting **F** is not a positive matrix, and where E(**P**) and var(**P**) do not converge. Indeed, some natural choices for weight vectors (such as **w** is a vector of ones) can create instabilities. For example, consider a two-station Tactical Planning Model. Station one receives an average of one unit of work each period, and each unit of work at station one results in an average of ten units of work at station two ($\Phi_{21} = 10$, all other Φ entries are zero). We convert station one into a constant-inventory station, so that $\beta_1 = 1$, and $\beta_2 = 0$. The vector of weights is just $\mathbf{w} = (1,1)'$. Using the relationship (22) for **F**, we find:

$$F = \begin{pmatrix} -9 & 1 \\ 10 & 0 \end{pmatrix}, \text{ and } \lambda_0 = -10, \lambda_1 = -1.$$

Assume that the lead time at each station, $\mathbf{D}_{ii} = 1$. Then, if we attempt to use the results of Propositions 2 and 3, we will find that the resulting expressions do not converge.

Intuitively, what is happening is that a small fluctuation of δ at station 2 immediately produces a $-\delta$ change to the arrivals at station 1 to keep inventory constant. In the following period, however, said $-\delta$ change at station 1 induces a -10δ fluctuation at station 2. This large fluctuation sets off a reinforcing loop between stations one and two, causing the "inventory" and "production" quantities to oscillate in ever-widening levels.

4. Setting the Vector of Inventory Weights and the Initial Inventories

4.1 Making Initial Inventories Large Enough to Address Production Fluctuations

Equation (18) defines the constant-inventory releases, which tell the shop to increase or decrease work at the controlled stations depending on the most recent inventory fluctuations at the other stations. Increasing work presents little difficulty. However, decreasing work requires that the typical work-in-progress at the controlled stations be great enough that the typical production less any decreases be greater than zero. In theory, this is not possible, since we assume that the fluctuations are normally distributed. In practice all fluctuations will be bounded, so we expect the model to work well if the prescribed input is greater than zero "most of the time".

For example, if the fluctuations are approximately normally distributed, and we let "most of the time" be more than 97% of the time, we can develop a formula to find acceptable initial inventory levels. We want to have Q_{it} for controlled stations positive more than 97% of the time, which, in steady state, and recalling properties of the normal distribution, is equivalent to having $E[Q_i] \geq 2\sqrt{\text{var}[Q_i]}$. Since

$P_{it} = a_i Q_{it}$, the previous inequality is equivalent to $a_i^{-1} E[P_i] \geq 2a_i^{-1}\sqrt{\text{var}[P_i]}$, which simplifies to $E[P_i] \geq 2\sqrt{\text{var}[P_i]}$.

Substituting in the formulas for the expectation and variance of $\mathbf{P}$ that we found in the previous section, we require that we set $\mathbf{Q}_0$ large enough that:

$$E[P_i] \geq 2\sqrt{\text{var}[P_i]} = 2\sqrt{\mathbf{S}_{ii}}, \tag{32}$$

for all controlled stations *i*.

In addition, depending on the job shop, we might ensure that work arrivals to stations are almost always positive, as well. (The latter is important in shops where work cannot simply be removed from queues.) By definition, we have that arrivals $\mathbf{A}_t = \mathbf{\Phi P}_t + \boldsymbol{\varepsilon}_t$. Again assuming that arrival fluctuations are approximately normally distributed, we want the expected arrivals to be greater than or equal to twice the standard deviation of the arrivals. Mathematically, we want:

$$E[A_i] \geq 2\sqrt{\text{var}[A_i]}, \tag{33}$$

for all controlled stations *i*. By taking moments of $\mathbf{A}_t = \mathbf{\Phi P}_t + \boldsymbol{\varepsilon}_t$, we find that $E[\mathbf{A}] = \mathbf{\Phi} E(\mathbf{P}) + \boldsymbol{\mu}$, and $\text{var}[\mathbf{A}] = \mathbf{\Phi}' \text{var}(\mathbf{P})\mathbf{\Phi} + \mathbf{\Sigma}$.

4.2 Creating Equivalent TPM and Constant Inventory Models

In this section, we create constant inventory models that are *equivalent* to TPM models of the same job shop. By equivalent we mean that all stations have the same expected production, expected queue levels, and expected exogenous inputs in both models.

In this section, we will compare similar variables from the TPM and the constant inventory (CIM) models. We denote TPM variables with the subscript *TPM*, and CIM variables with subscript *CI*. We first show how to create a CIM model from a TPM model.

Proposition 4. *(TPM $\Rightarrow$ CIM). Suppose that we have a CIM model and a TPM model with identical flow matrices Φ, and matrix of smoothing factors $\mathbf{D}$. Suppose that the TPM model has vector of expected inputs $\boldsymbol{\mu}_{TPM}$. Assume that $\mathbf{E}[\mathbf{P}_{CI,0}] = \mathbf{E}[\mathbf{P}_{TPM}]$, where $\mathbf{E}[\mathbf{P}_{CI,0}]$ is the vector of initial production quantities in the CIM model. Also assume that $\boldsymbol{\mu}_{TPM} \neq \boldsymbol{\mu}_{CI}$. Then, given any CIM inventory vector $\mathbf{w}$ that satisfies $\mathbf{w}'(\boldsymbol{\mu}_{TPM} - \boldsymbol{\mu}_{CI}) \neq 0$, we can find a CIM template vector $\boldsymbol{\beta}$ such that $\mathbf{w}'\boldsymbol{\beta} > 0$, and:*

(i) *The models have the same expected production vectors, or* $\mathbf{E}[\mathbf{P}_{CI}] = \mathbf{E}[\mathbf{P}_{TPM}]$.

(ii) *The models expect to receive the same work arrivals from outside the shop each time period, or* $\mathbf{E}[\mathbf{R}] + \boldsymbol{\mu}_{CI} = \boldsymbol{\mu}_{TPM}$.

(iii) *The expected weighted inventory in the shop is* $\mathbf{W} = \mathbf{w}'\mathbf{D}^{-1}\mathbf{E}(\mathbf{P}_{TPM})$.

Proof. For the CIM model, the expected production at time t satisfies the following recursion equation:

$$\mathbf{E}[\mathbf{P}_{\mathbf{CI},t}] = (\mathbf{I} - \mathbf{D} + \mathbf{DF})\mathbf{E}[\mathbf{P}_{\mathbf{CI},t-1}] + \mathbf{DG}\boldsymbol{\mu}_{\mathbf{CI}}. \tag{34}$$

We are interested in finding a fixed-point solution to (a), such that $\mathbf{E}[\mathbf{P}_{\mathbf{CI},t}] = \mathbf{E}[\mathbf{P}_{\mathbf{CI},t-1}]$. In particular, suppose we can find a template vector $\boldsymbol{\beta}$ such that $\mathbf{E}[\mathbf{P}_{\mathbf{CI},t}] = \mathbf{E}[\mathbf{P}_{\mathbf{CI},t-1}] = \mathbf{E}[\mathbf{P}_{\mathbf{TPM}}]$. Then, if $\mathbf{E}[\mathbf{P}_{\mathbf{CI},0}] = \mathbf{E}[\mathbf{P}_{\mathbf{TPM}}]$, it follows that $\mathbf{E}[\mathbf{P}_{\mathbf{CI}}] = \mathbf{E}[\mathbf{P}_{\mathbf{TPM}}]$. Mathematically, we want to solve the following equation for $\boldsymbol{\beta}$:

$$\begin{aligned} &\mathbf{E}[\mathbf{P}_{\mathbf{TPM}}] = (\mathbf{I} - \mathbf{D} + \mathbf{DF})\mathbf{E}[\mathbf{P}_{\mathbf{TPM}}] + \mathbf{DG}\boldsymbol{\mu}_{\mathbf{CI}}, \\ \Rightarrow\ &\mathbf{E}[\mathbf{P}_{\mathbf{TPM}}] = \left(\mathbf{I} - \mathbf{D} + \mathbf{D}\left[\boldsymbol{\Phi} + \boldsymbol{\beta}\mathbf{w}'(\mathbf{I} - \boldsymbol{\Phi})\right]\right)\mathbf{E}[\mathbf{P}_{\mathbf{TPM}}] + \mathbf{D}(\mathbf{I} - \boldsymbol{\beta}\mathbf{w}')\boldsymbol{\mu}_{\mathbf{CI}}. \end{aligned} \tag{35}$$

The solution to (35) is:

$$\boldsymbol{\beta} = \frac{(\mathbf{I} - \boldsymbol{\Phi})E[\mathbf{P}_{\mathbf{TPM}}] - \boldsymbol{\mu}_{\mathbf{CI}}}{\mathbf{w}'\left((\mathbf{I} - \boldsymbol{\Phi})E[\mathbf{P}_{\mathbf{TPM}}] - \boldsymbol{\mu}_{\mathbf{CI}}\right)}, \tag{36}$$

which is well-defined provided that $(\mathbf{I} - \boldsymbol{\Phi})\mathbf{E}[\mathbf{P}_{\mathbf{TPM}}] \neq \boldsymbol{\mu}_{\mathbf{CI}}$ and $\mathbf{w}'\boldsymbol{\beta} > 0$. Note that $(\mathbf{I} - \boldsymbol{\Phi})\mathbf{E}[\mathbf{P}_{\mathbf{TPM}}] = \boldsymbol{\mu}_{\mathbf{TPM}}$ follows immediately from the TPM formula for $\mathbf{E}[\mathbf{P}_{\mathbf{TPM}}]$. Then we can rewrite (36) as:

$$\boldsymbol{\beta} = \frac{\boldsymbol{\mu}_{\mathbf{TPM}} - \boldsymbol{\mu}_{\mathbf{CI}}}{\mathbf{w}'\left(\boldsymbol{\mu}_{\mathbf{TPM}} - \boldsymbol{\mu}_{\mathbf{CI}}\right)}. \tag{37}$$

Thus, we have found a $\boldsymbol{\beta}$ such that $\mathbf{E}[\mathbf{P}_{\mathbf{CI}}] = \mathbf{E}[\mathbf{P}_{\mathbf{TPM}}]$ and $\mathbf{w}'\boldsymbol{\beta} = 1$, provided that $\boldsymbol{\mu}_{\mathbf{TPM}} \neq \boldsymbol{\mu}_{\mathbf{CI}}$ and $\mathbf{w}'\left(\boldsymbol{\mu}_{\mathbf{TPM}} - \boldsymbol{\mu}_{\mathbf{CI}}\right) \neq 0$.

Further, since $E[\mathbf{Q}_{\mathbf{CI}}] = \mathbf{D}^{-1}E[\mathbf{P}_{\mathbf{CI}}]$, we immediately have that $E[\mathbf{Q}_{\mathbf{CI}}] = E[\mathbf{Q}_{\mathbf{TPM}}]$, and that the total expected weighted inventory in the shop is $\mathbf{W} = \mathbf{w}'\mathbf{D}^{-1}\mathbf{E}(\mathbf{P}_{\mathbf{TPM}})$.

It remains to show that with this $\boldsymbol{\beta}$, $\mathbf{E}[\mathbf{R}] + \boldsymbol{\mu}_{\mathbf{CI}} = \boldsymbol{\mu}_{\mathbf{TPM}}$. We set $\boldsymbol{\beta}$ so that $\mathbf{E}[\mathbf{P}_{\mathbf{CI}}] = \mathbf{E}[\mathbf{P}_{\mathbf{TPM}}]$. By the inventory balance equations, (2), we have that:

$$\mathbf{E}[\mathbf{Q}_{\mathbf{CI}}] = \mathbf{E}[\mathbf{Q}_{\mathbf{CI}}] - \mathbf{E}[\mathbf{P}_{\mathbf{CI}}] + \mathbf{E}[\mathbf{A}_{\mathbf{CI}}], \text{ and } \mathbf{E}[\mathbf{Q}_{\mathbf{TPM}}] = \mathbf{E}[\mathbf{Q}_{\mathbf{TPM}}] - \mathbf{E}[\mathbf{P}_{\mathbf{TPM}}] + \mathbf{E}[\mathbf{A}_{\mathbf{TPM}}]. \tag{38}$$

We have that $\mathbf{E}[\mathbf{P}_{\mathbf{CI}}] = \mathbf{E}[\mathbf{P}_{\mathbf{TPM}}]$, and we have shown that $E[\mathbf{Q}_{\mathbf{CI}}] = E[\mathbf{Q}_{\mathbf{TPM}}]$. Then $\mathbf{E}[\mathbf{A}_{\mathbf{CI}}] = \mathbf{E}[\mathbf{A}_{\mathbf{TPM}}]$. Substituting yields:

$$\boldsymbol{\Phi}\mathbf{E}[\mathbf{P}_{\mathbf{CI}}] + \mathbf{E}[\mathbf{R}] + \boldsymbol{\mu}_{\mathbf{CI}} = \mathbf{E}[\mathbf{A}_{\mathbf{CI}}] = \mathbf{E}[\mathbf{A}_{\mathbf{TPM}}] = \boldsymbol{\Phi}\mathbf{E}[\mathbf{P}_{\mathbf{TPM}}] + \boldsymbol{\mu}_{\mathbf{TPM}}. \tag{39}$$

It immediately follows that $\mathbf{E}[\mathbf{R}] + \boldsymbol{\mu}_{\mathbf{CI}} = \boldsymbol{\mu}_{\mathbf{TPM}}$. □

The requirement that $\mathbf{w}'\left(\boldsymbol{\mu}_{\mathbf{TPM}} - \boldsymbol{\mu}_{\mathbf{CI}}\right) \neq 0$ says that the weighted inventory being kept constant must include at least one of the constant inventory stations. Consequently, it is easy to create a CIM model from a TPM model, as follows:

1. Compute the expected production vector for the TPM model, $\mathbf{E}[\mathbf{P}_{\mathbf{TPM}}]$.

2. Select the stations that receive constant-inventory rule inputs by giving them non-zero β_i's in accordance with formula (37), so that $\boldsymbol{\beta} = (\boldsymbol{\mu}_{TPM} - \boldsymbol{\mu}_{CI})(\mathbf{w}'(\boldsymbol{\mu}_{TPM} - \boldsymbol{\mu}_{CI}))^{-1}$.
3. Create the equivalent constant inventory model by setting the CIM's initial inventory levels to $\mathbf{Q}_0 = \mathbf{D}^{-1}\mathbf{E}[\mathbf{P}_{TPM}]$, and noting that the total expected weighted inventory is $\mathbf{W} = \mathbf{w}'\mathbf{D}^{-1}(\mathbf{E}_{TPM})$.

We now show how to create a TPM model from a CIM model.

Proposition 5. *(CIM $\Rightarrow$ TPM) Suppose that we have a CIM model and a TPM model with identical flow matrices* Φ*, and matrix of smoothing factors* $\mathbf{D}$*. Suppose that the CIM model has vector of expected inputs* $\boldsymbol{\mu}_{CI}$*, weight vector* $\mathbf{w}$*, and template vector* $\boldsymbol{\beta}$*. Then we can find a* $\boldsymbol{\mu}_{TPM}$ *such that* $\mathbf{E}[\mathbf{P}_{CI}] = \mathbf{E}[\mathbf{P}_{TPM}]$ *and* $\mathbf{E}[\mathbf{R}] + \boldsymbol{\mu}_{CI} = \boldsymbol{\mu}_{TPM}$.

Proof. Note that $\mathbf{E}[\mathbf{P}_{CI}]$ solves the steady-state recursion equation:

$$\begin{aligned}\mathbf{E}[\mathbf{P}_{CI}] &= (\mathbf{I} - \mathbf{D} + \mathbf{DF})\mathbf{E}[\mathbf{P}_{CI}] + \mathbf{DG}\boldsymbol{\mu}_{CI} \\ &= (\mathbf{I} - \mathbf{D} + \mathbf{D}\Phi)\mathbf{E}[\mathbf{P}_{CI}] + (\mathbf{D}\boldsymbol{\beta}\,\mathbf{w}'(\mathbf{I} - \Phi)\mathbf{E}[\mathbf{P}_{CI}] + \mathbf{D}(\mathbf{I} - \boldsymbol{\beta}\mathbf{w}')\boldsymbol{\mu}_{CI}).\end{aligned} \tag{40}$$

Set $\boldsymbol{\mu}_{TPM} = \boldsymbol{\beta}\,\mathbf{w}'(\mathbf{I} - \Phi)\mathbf{E}[\mathbf{P}_{CI}] + (\mathbf{I} - \boldsymbol{\beta}\mathbf{w}')\boldsymbol{\mu}_{CI}$. Substituting this expression into (40) yields:

$$\mathbf{E}[\mathbf{P}_{CI}] = (\mathbf{I} - \mathbf{D} + \mathbf{D}\Phi)\mathbf{E}[\mathbf{P}_{CI}] + \mathbf{D}\boldsymbol{\mu}_{TPM}. \tag{41}$$

Further, note that $\mathbf{E}[\mathbf{P}_{TPM}]$ is the unique solution to the TPM recursion equation:

$$\mathbf{E}[\mathbf{P}_{TPM}] = (\mathbf{I} - \mathbf{D} + \mathbf{D}\Phi)\mathbf{E}[\mathbf{P}_{TPM}] + \mathbf{D}\boldsymbol{\mu}_{TPM}. \tag{42}$$

But, since (41) and (42) are the same equations, we immediately have that $\mathbf{E}[\mathbf{P}_{CI}] = \mathbf{E}[\mathbf{P}_{TPM}]$ provided that $\boldsymbol{\mu}_{TPM} = \boldsymbol{\beta}\,\mathbf{w}'(\mathbf{I} - \Phi)\mathbf{E}[\mathbf{P}_{CI}] + (\mathbf{I} - \boldsymbol{\beta}\mathbf{w}')\boldsymbol{\mu}_{CI}$. Then, by applying (19) to the latter equation, we have that $\mathbf{E}[\mathbf{R}] + \boldsymbol{\mu}_{CI} = \boldsymbol{\mu}_{TPM}$. □

4.3 Minimizing Production Standard Deviations

In this subsection we show how to minimize production standard deviations under the constant inventory rule by optimizing the inventory weights and the choice of β_i's. In general, we will have two motivations for doing so. First, minimizing production standard deviations may often be useful in just-in-time environments, or in minimizing wear on machines caused by large production fluctuations. Second, a common way to set a station's capacity is to set the capacity equal to the expected production plus some constant times the standard deviation of production; doing so will allow the machine to process the desired amount most of the time. For example, if production fluctuations are normally distributed, setting the capacity at the station to be the expected production plus twice the standard deviation of production will allow the station to process the desired amount ~97% of the time. Thus, minimizing standard deviations directly translates to reductions in required capacity and cost savings.

We assume that we have an existing TPM job shop that we would like to convert to a constant inventory job shop. Thus, in the new CIM shop we want to have $E[\mathbf{Q}_{CI}] = E[\mathbf{Q}_{TPM}]$ and $\mathbf{E}[\mathbf{P}_{CI}] = \mathbf{E}[\mathbf{P}_{TPM}]$. In Proposition 4, we found that meeting these requirements is equivalent to ensuring that $\mathbf{w}'(\boldsymbol{\mu}_{TPM} - \boldsymbol{\mu}_{CI}) \neq 0$ and setting $\boldsymbol{\beta} = (\boldsymbol{\mu}_{TPM} - \boldsymbol{\mu}_{CI})(\mathbf{w}'(\boldsymbol{\mu}_{TPM} - \boldsymbol{\mu}_{CI}))^{-1}$. Consequently, finding the CIM shop that minimizes production standard deviations is a nonlinear programming problem with $\mathbf{w}$ and $\boldsymbol{\mu}_{CI}$ as decision variables. We have the following nonlinear program, *S-MIN*:

$$\begin{array}{lll} & \min & f(\mathbf{w},\boldsymbol{\beta}) \\ \textit{(S-MIN)} & \text{s.t.} & \mathbf{w} \geq \mathbf{0}, \\ & & \boldsymbol{\beta} = ((\boldsymbol{\mu}_{TPM} - \boldsymbol{\mu}_{CI})(\mathbf{w}'(\boldsymbol{\mu}_{TPM} - \boldsymbol{\mu}_{CI}))^{-1}, \\ & & \mathbf{0} \leq \boldsymbol{\mu}_{CI} \leq \boldsymbol{\mu}_{TPM}, \end{array} \qquad (43)$$

where $f(\mathbf{w},\boldsymbol{\beta})$ is given by the following algorithm:

1. Generate the corresponding $\mathbf{F}$ and $\mathbf{G}$ matrices. Calculate the resulting covariance matrix of production, var($\mathbf{P}$), using Proposition (3).
2. Let $f(\mathbf{w},\boldsymbol{\beta})$ be the sum of the square roots of the diagonal entries of var($\mathbf{P}$) (the diagonal entries of var($\mathbf{P}$) are the station variances).

Despite its complicated appearance, $f(\mathbf{w},\boldsymbol{\beta})$ is a continuous and continuously differentiable function that has been fairly easy to minimize in practice. First, we move the constraint on $\boldsymbol{\beta}$ directly to the objective function, yielding a new nonlinear program with an objective function $f(\mathbf{w},\boldsymbol{\mu}_{CI})$ and without the third constraint. This new nonlinear program has only orthant constraints, and there are special methods that solve nonlinear programs with orthant constraints quickly.

In particular, to find the optimal solution for the examples in the next section, we employ the Two-Metric Projection Method (c.f. Bertsekas, 1995c, 224-229) with diagonally scaled steepest descent iterations (taking numerical approximations of derivatives). Using this approach, an SGI workstation found the optimal solution in well under a minute of computing time.

4.4 Comparison of Constant Inventory and Constant Release Models

In this subsection we compare the constant inventory model with a constant release (CR) model. In a constant release model, we add the same amount of work to the "controlled" stations each time period, regardless of queue levels at other stations. We can use the nonlinear program in section 4.3 to prove that, under certain conditions, constant inventory models always achieve lower variances than constant release models.

As in section 4.2, we will compare similar variables from the constant release and constant inventory models. We denote constant release variables with the subscript *CR*.

Proposition 6. *Suppose that we have a constant inventory (CI) model and a constant release (CR) model with the following properties:*

- *The CI and CR models have identical flow matrices* Φ, *matrix of smoothing factors* **D**, *vector of expected inputs* μ, *and input covariance matrix* Σ.
- *Define all stations that receive input from outside the system to be* controlled stations. *Controlled stations behave according to the constant inventory control rule for the CI model, and receive a constant amount of work each period in the CR model. Let the controlled stations be numbered* 1 *to* K. *Then the exogenous variance at each controlled stations in either model is 0, or* $\text{var}[N_{i,CI}] = \text{var}[N_{i,CR}] = 0$, *for stations* 1 *to* K.
- $\mathbf{E[Q_{CR}]} = \mathbf{E[Q_{CI}]}$ *and* $E[R_{i,CI}] + \mu_{i,CI} = \mu_{TPM}$ *for all controlled stations* i.

Define $g(\mathbf{S})$ *to be* $\sum_{i=1}^{N} \sqrt{\mathbf{S_{ii}}}$, *where* **S** *is a production covariance matrix. Further, for the constant inventory model,* $g(\mathbf{S_{CI}})$ *is found as the optimal value of the program* S-MIN. *Then:*

- *(Case 1) If none of the controlled stations receive input from another station,* $g(\mathbf{S_{CI}}) \le g(\mathbf{S_{CR}})$.
- *(Case 2) If there is only one controlled station,* $g(\mathbf{S_{CI}}) \le g(\mathbf{S_{CR}})$.

Proof. *Case 1*: Consider the following weight vector:

$$\mathbf{w_c} : \begin{cases} w_i = 1, i \text{ a controlled station} \\ w_i = 0, \text{otherwise} \end{cases} . \tag{44}$$

Choose $\boldsymbol{\mu_{CI}} = \mathbf{0}$, so that all arrivals to the controlled stations come only from constant-inventory arrivals. Clearly, $\mathbf{w_c}$ and μ_{CI} is a feasible solution to the nonlinear program. We now determine whether $\mathbf{w_c}$ determines a constant release policy identical to that of the CR model.

This weight vector requires that $\sum_{i=1}^{K} Q_{i,t} = \sum_{i=1}^{K} Q_{i,t-1}$. By assumption, none of the controlled stations receives input from other stations, so they never see arrival variances resulting from fluctuations at other stations. Further, the exogenous variances at all controlled stations are assumed to be zero. Then the controlled stations process exactly the same amount of work each time period. Then, satisfying the constant inventory constraints forces the controlled stations to receive exactly the same amount of work each period, as in the constant release model. Further, since we require that $\mathbf{E[R]} + \boldsymbol{\mu_{CI}} = \boldsymbol{\mu_{TPM}}$, $\mathbf{w_c}$ will cause the constant inventory model to behave just like the CR model.

Consequently, for this case we have found a feasible solution to the nonlinear program that duplicates the CR model. Then $g(\mathbf{S_{CI}}) \le g(\mathbf{S_{CR}})$. □

Case 2: Consider the same weight vector $\mathbf{w_c}$ presented in case 1, and set $\mu_{CI} = \mathbf{0}$. As in Case 1, $\mathbf{w_c}$ and μ_{CI} is a feasible solution to the nonlinear program.

Note that a constant inventory model with this feasible solution does not correspond directly to a constant release model, since the controlled station adjusts for fluctuations in the arrivals from the other stations. Thus, the inventory in the controlled station will remain constant, whereas the inventory in the constant release station will fluctuate. Then, for controlled station i, $0 = \text{var}(P_{i,CI}) \le \text{var}(P_{i,CR})$.

By definition, the work that arrives at any station j from station i is given by $A_{jit} = \Phi_{ji} P_{it} + \varepsilon_{jit}$. Then $\text{var}(A_{ji}) = (\Phi_{ji})^2 \text{var}(P_{it}) + \text{var}(\varepsilon_{ji})$. But then, the fact that $\text{var}(P_{i,CI}) \le \text{var}(P_{i,CR})$ implies that $\text{var}(A_{ji,CI}) \le \text{var}(A_{ji,CR})$. The latter fact directly implies that $\text{var}(P_{j,CI}) \le \text{var}(P_{j,CR})$, for all stations j,

since all other model parameters are the same for the CIM and TPM models. But then, since g(**S**) directly increases with respect to the var(P_j)'s, $g(\mathbf{S} \mid \mathbf{w_C}, \mathbf{0}) \leq g(\mathbf{S_{CR}})$. Since we have found a feasible solution to S-MIN whose objective value is less than $g(\mathbf{S_{CR}})$, we immediately have that $g(\mathbf{S_{CI}}) \leq g(\mathbf{S_{CR}})$.□

5. An Example

We illustrate the use of the constant inventory model with the same example given in Graves (1986). This example corresponds to a job shop that produces spindle components for grinding machines. The job shop consists of ten stations, and the work flow is described by the matrix Φ given in Table I. The only station that receives work from outside the network is station 1 (the lathe); it receives 4 hours of work, on average, each time period.

Recall that Φ is not a probability matrix; its elements show the expected amount of work generated at a subsequent station by a fixed amount of work at the current station. For example, one hour of work at station 8 creates, on average, 3.43 hours of work at station 9.

Table 1 -- Work Flow Matrix for the Example

	From Work Station									
To Work Station	1	2	3	4	5	6	7	8	9	10
1 (lathe)			0.11		0.68					
2 (copy lathe)	0.15									
3 (drill press)	0.04	0.01		0.71		0.6			0.07	
4 (milling)	0.01	0.41								
5 (rough grinder)	0.03	0.37	1.36							
6 (internal grinder)	0.24				0.15				0.13	
7 (thread cutting)					0.10					
8 (hole abrading)	0.01					0.22	1.00			
9 (precision grider)								3.43		
10 (ultra-precision grinder)									1.16	

To test the performance of the constant inventory model, we perform a 2^k design test, as follows. We consider pairs of stations (1 and 2, 3 and 4, etc.), and we alternate between giving the pairs "high" input variances and "low" input variances. Each "high" station has an input variance of 4, and a lead time of 4. Each "low" station has an input variance of 0.05, and a lead time of 2. There are 32 test cases formed by choosing all ways to assign "high" or "low" to the station pairs. Table 2 shows the input variances and lead times for each test.

Table 2 -- Tested Input Variances and Lead Times

	Station Input Variances					Station Lead Times						Station Input Variances					Station Lead Times				
Test	1,2	3,4	5,6	7,8	9,10	1,2	3,4	5,6	7,8	9,10	Test	1,2	3,4	5,6	7,8	9,10	1,2	3,4	5,6	7,8	9,10
1	0.05	0.05	0.05	0.05	0.05	2	2	2	2	2	17	4.0	0.05	0.05	0.05	0.05	4	2	2	2	2
2	0.05	0.05	0.05	0.05	4.0	2	2	2	2	4	18	4.0	0.05	0.05	0.05	4.0	4	2	2	2	4
3	0.05	0.05	0.05	4.0	0.05	2	2	2	4	2	19	4.0	0.05	0.05	4.0	0.05	4	2	2	4	2
4	0.05	0.05	0.05	4.0	4.0	2	2	2	4	4	20	4.0	0.05	0.05	4.0	4.0	4	2	2	4	4
5	0.05	0.05	4.0	0.05	0.05	2	2	4	2	2	21	4.0	0.05	4.0	0.05	0.05	4	2	4	2	2
6	0.05	0.05	4.0	0.05	4.0	2	2	4	2	4	22	4.0	0.05	4.0	0.05	4.0	4	2	4	2	4

	Station Input Variances					Station Lead Times						Station Input Variances					Station Lead Times				
Test	1,2	3,4	5,6	7,8	9,10	1,2	3,4	5,6	7,8	9,10	Test	1,2	3,4	5,6	7,8	9,10	1,2	3,4	5,6	7,8	9,10
7	0.05	0.05	4.0	4.0	0.05	2	2	4	4	2	23	4.0	0.05	4.0	4.0	0.05	4	2	4	4	2
8	0.05	0.05	4.0	4.0	4.0	2	2	4	4	4	24	4.0	0.05	4.0	4.0	4.0	4	2	4	4	4
9	0.05	4.0	0.05	0.05	0.05	2	4	2	2	2	25	4.0	4.0	0.05	0.05	0.05	4	4	2	2	2
10	0.05	4.0	0.05	0.05	4.0	2	4	2	2	4	26	4.0	4.0	0.05	0.05	4.0	4	4	2	2	4
11	0.05	4.0	0.05	4.0	0.05	2	4	2	4	2	27	4.0	4.0	0.05	4.0	0.05	4	4	2	4	2
12	0.05	4.0	0.05	4.0	4.0	2	4	2	4	4	28	4.0	4.0	0.05	4.0	4.0	4	4	2	4	4
13	0.05	4.0	4.0	0.05	0.05	2	4	4	2	2	29	4.0	4.0	4.0	0.05	0.05	4	4	4	2	2
14	0.05	4.0	4.0	0.05	4.0	2	4	4	2	4	30	4.0	4.0	4.0	0.05	4.0	4	4	4	2	4
15	0.05	4.0	4.0	4.0	0.05	2	4	4	4	2	31	4.0	4.0	4.0	4.0	0.05	4	4	4	4	2
16	0.05	4.0	4.0	4.0	4.0	2	4	4	4	4	32	4.0	4.0	4.0	4.0	4.0	4	4	4	4	4

For each test, we evaluate the sum of the production standard deviations, $g(\mathbf{S})$, across all stations for three rules: original TPM, constant release, and constant inventory rules.

- For the original model, we simply let the input variance at station 1 be 0.05 or 4, depending on whether the station is set to "low" or "high".
- For the constant release model, the input variance at station 1 is always 0.
- For the constant inventory model, station 1's input is completely controlled to maintain a constant weighted-inventory throughout the job shop (i.e. $\mu_{CI} = 0$, so that $\beta_1 = 1$). We consider two CIM models. First, we evaluate a shop with **w** = a vector of ones. Second, we evaluate a shop with the **w** that optimizes the nonlinear program *S-MIN*.

Table 3 gives the results of the 32 tests. For each test, we give the sum of the standard deviations of production across all station for each of the three models. We also present percentage comparisons between the models ("% change" is the percent difference between the listed model's $g(\mathbf{S})$ and the original TPM's $g(\mathbf{S})$.)

Table 3 -- Test Results (Sums of Production Standard Deviations)

	Original TPM	Constant Release		Constant Inventory (**w** = ones)		Constant Inventory (**w** = optimal **w**)	
Test	g(S)	g(S)	% Change	g(S)	% Change	g(S)	% Change
1	2.28	2.22	0.03	2.70	-18.7%	2.05	10.2%
2	3.27	3.21	0.02	5.63	-72.4%	3.03	7.3%
3	9.67	9.63	0.00	13.97	-44.6%	9.24	4.4%
4	8.91	8.88	0.00	15.61	-75.1%	8.51	4.6%
5	4.57	4.55	0.00	5.35	-17.1%	3.87	15.2%
6	5.18	5.16	0.00	7.24	-39.8%	4.53	12.6%
7	10.84	10.82	0.00	14.67	-35.3%	10.15	6.4%
8	10.12	10.10	0.00	15.83	-56.4%	9.44	6.7%
9	6.21	6.19	0.00	6.59	-6.1%	4.71	24.1%
10	6.80	6.79	0.00	8.47	-24.5%	5.47	19.6%
11	12.45	12.43	0.00	15.87	-27.5%	11.25	9.6%
12	11.71	11.70	0.00	17.00	-45.2%	10.53	10.0%
13	7.18	7.17	0.00	8.02	-11.7%	5.83	18.9%

	Original TPM	Constant Release		Constant Inventory (**w** = ones)		Constant Inventory (**w** = optimal **w**)	
Test	g(**S**)	g(**S**)	% Change	g(**S**)	% Change	g(**S**)	% Change
14	7.64	7.63	0.00	9.53	-24.8%	6.39	16.3%
15	12.96	12.95	0.00	16.45	-26.9%	11.79	9.0%
16	12.23	12.22	0.00	17.25	-41.0%	11.10	9.3%
17	4.53	3.83	0.15	3.87	14.6%	3.31	26.9%
18	5.36	4.71	0.12	5.49	-2.3%	4.24	21.0%
19	11.40	10.84	0.05	12.38	-8.6%	10.30	9.6%
20	10.66	10.09	0.05	13.10	-22.9%	9.57	10.2%
21	6.10	5.59	0.08	5.68	6.9%	4.89	19.8%
22	6.67	6.18	0.07	6.89	-3.4%	5.52	17.2%
23	12.21	11.76	0.04	13.35	-9.3%	11.09	9.2%
24	11.49	11.04	0.04	13.77	-19.8%	10.39	9.6%
25	7.30	6.87	0.06	6.45	11.6%	5.51	24.5%
26	7.87	7.45	0.05	7.77	1.3%	6.25	20.6%
27	13.44	13.07	0.03	14.29	-6.3%	12.00	10.7%
28	12.70	12.33	0.03	14.72	-15.9%	11.28	11.2%
29	8.17	7.80	0.05	7.71	5.7%	6.59	19.4%
30	8.62	8.25	0.04	8.78	-1.8%	7.14	17.1%
31	13.90	13.55	0.02	15.01	-8.0%	12.52	9.9%
32	13.17	12.83	0.03	15.25	-15.7%	11.83	10.2%

The table shows that the constant inventory rule (with the optimal **w**) offers substantial decreases in production variability over the original model. Decreases in the sum of production standard deviations ranged from about 5% to over 25%, with the average being about 13%. This performance is much better than the constant release rule, which only manages decreases from 0.2% to 12%, with the average being about 3%. Note that the constant release rule was particularly ineffective for cases in which most of the variability was far downstream from the first station (cases 1 to 16). The constant inventory rule usually produced a significant decrease, even for cases in which most of the variability was far downstream from the first station. As noted, these decreases in production standard deviations may reduce capacity requirements, lowering costs.

However, the table also shows that a great deal of care must be used in determining **w**. In this case, the natural choice of **w** (**w** = ones), resulted in large increases in production variability over the original model. In some cases, the increase to $g(\mathbf{S})$ was as great as 75%. This is an important effect, so we discuss it in some detail.

Consider experiment 8, in which the input standard deviations at the first four stations are 0.5, and the input stations at the last six stations are 4. Thus, in this experiment, the major production fluctuations are relatively far downstream from the controlled station, station 1.Table 3 lists g(**S**) for the optimal weights as being a 6.7% improvement over the original model, but g(**S**) for the unit weights as

being a 56.4% worsening over the original model. The following table compares the optimal and unit weights, along with the production standard deviations produced by each weight vector.

Table 4 -- Comparison of Optimal and Unit Weights

		Optimal Weights		Unit Weights	
Station	Input Standard Deviation	Weights	Production Standard Deviation	Weights	Production Standard Deviation
1	0.5	9.7104	0.0070	1	5.8906
2	0.5	0.0995	0.1290	1	0.8236
3	0.5	0.0183	0.2518	1	0.3845
4	0.5	0.1284	0.1350	1	0.3901
5	4.0	0.0118	0.8084	1	1.0585
6	4.0	0.0225	0.8388	1	1.1394
7	4.0	0.0000	0.7588	1	0.7561
8	4.0	0.0057	0.9571	1	0.8734
9	4.0	0.0021	2.6901	1	2.2507
10	4.0	0.0013	2.8663	1	2.2628

We see that the optimal weight vector massively overweights the first station's queue. As a result, the production standard deviation at the first station is near zero, and the production standard deviations at the station that receive significant work from station 1 (stations 2-6) are lowered as well, largely because the work coming from station 1 is smoothed. Conversely, with the unit weights, the production standard deviation is extremely high – greater than expected production at station 1. The standard deviations at stations 2-6 are elevated as well.

Intuitively, what is happening is similar to the example in Section 3, in which we demonstrated the existence of weight vector which cause the resulting vectors to be unstable. With the unit weights, the network responds to large weighted-queue fluctuations at the downstream stations by making large controlled releases to station 1. Unfortunately, doing so results in large fluctuations at the first station. Further, by the time the controlled releases reach the downstream stations, most of the downstream fluctuations causing the releases will have vanished. Thus, the controlled releases cause more fluctuations downstream that are then addressed through more controlled releases, establishing a reinforcing cycle. In extreme cases, as in the Section 3 example, this reinforcing cycle may cause the system to become unstable. In this instance, the network is stable – indeed, the standard deviations downstream are slightly lower with the unit weights than with the optimal weights. Nonetheless, the benefit downstream is very small in comparison to the increased production variability upstream.

Conversely, the optimal weight vector recognizes that one cannot counter downstream fluctuations with controlled releases that will not reach the downstream stations until after the fluctuations have largely dissipated. This explains the overweighting of the first station's queue, and the small

weights placed on the queues of the stations that receive work from station 1. Controlled releases will counterbalance fluctuations upstream.

The optimal weights are not always as pronounced as they are for experiment 8. Nonetheless, in general, optimal weight vectors will substantially overweight the queues of the controlled release stations and place some weight on the queues of the stations receiving work from the controlled release stations. Weights on queues of stations farther downstream will be small if not zero.

In summary, we have shown how to add constant weighted-inventory constraints to TPM models, and how to set vectors of inventory weights to minimize production standard deviations. The resulting models appear to significantly reduce fluctuations in production for fixed work-in-process levels, which may lead to reductions in capacity requirements for fixed work-in-process levels. Alternately, we have discussed in Chapter 2 that longer lead times and higher inventory levels are associated with smoother production flows; by applying constant-inventory rules we should be able to lower lead times and work-in process levels while maintaining the same production variances.

We note three opportunities for future research on constant-inventory TPM models. First, the stability of these models should be explored further. Proposition 2, the condition on the inventory weights that guarantees convergence, is a sufficient but not necessary condition. Experimental tests suggest that models whose weights violate the proposition are able to converge if (1) the resulting total weighted arrivals are less than the total weighted production, or (2) if station lead times are lengthened sufficiently. These phenomena warrant additional study.

Second, constant inventory models with multiple constraints should be explored. Multiple constraints would allow us to control constant inventory stations independently. Controlling stations independently would allow these stations to reduce production variations more effectively. Therefore, multiple-constraint constant inventory models would be the next logical step in this research area.

Finally, we note that although the discussion in this chapter was restricted to TPM models, this was done so solely for simplicity. It should be fairly simple to develop a constant inventory models for a job shop with general linear control rules. The derivation would be similar to that in Section 2, except that **D** would be a non-diagonal matrix, and additional γ_{it} terms representing production fluctuations would be carried through the equations. Similarly, developing an approximate constant inventory model for a job shop with general control rules should be tractable, as well.

Chapter 5: Models That Process Discrete Jobs (Class LRS-MR)

1. Introduction

In previous chapters, we have treated work in the job shops as if they were fluid flows. As such, fluid work completed at one station is multiplied, and becomes fluid work at a downstream station. In practice, this relationship often does not hold. Instead, rather than process "work", stations process jobs, and it is the completion of jobs that triggers work at downstream stations. Further, a job completed at one station need not become a job at a downstream station; it can trigger multiple jobs (or zero jobs) downstream.

For example, consider a data-processing environment in which the completion of one computing job initiates other computer jobs downstream. The number of new jobs is a random variable, which may be zero. The downstream computing station then receives all the work required to complete the new jobs. Note that the new work at the downstream station depends on the new jobs, not the computing work completed at the upstream station. Further, the work required to complete one of the upstream jobs may vary by orders of magnitude from the work required to complete one of the downstream jobs. In this environment, a direct relationship between work completed at upstream and downstream relationships is no longer suitable.

In this chapter, we study the LRS-MR model, which incorporates work transitions that are major generalizations of the fluid-flow transfers. In particular, the LRS-MR model implements a flexible, *request-based* relationship between the work completed at an upstream station and work arrivals at downstream stations. In the new relationship, the work completed at an upstream station is expressed as some number of completed jobs. These jobs become requests at a downstream station. The downstream station converts the requests into a number of jobs to perform, and then converts the jobs into some amount of work (measured in instructions). The downstream station processes the work and separates the completed work into completed jobs. The completed jobs are sent to stations farther downstream, and the process continues. The resulting flexibility of the LRS-MR model should make it a useful addition to the existing scheduling and planning manufacturing models.

The new relationship gives rise to two new control rules. For the sake of simplicity, we will assume that stations process a fixed fraction of their work-in-queue each period, as with the Tactical Planning Model. However, we allow "work-in-queue" to have two definitions. A station may process either a fixed fraction of the total work in its queue, measured in instructions (the *instruction-control rule*), or a fixed fraction of the jobs in its queue (the *job-control rule*). We will be able to model networks using either or both of these rules.

We also assume, for the sake of simplicity, that new work arrivals to the network come from independent distributions. However, note that these distributions now generate work requests, not the fluid amounts of work seen in previous models.

However, it is far from obvious that request-based relationships can be modeled using the approach used to analyze other MR models. No longer are work arrivals simple functions of work completed upstream. Instead, as will be seen, work arrivals become complicated random sums of random variables. While we can and will derive a formula for production, this equation cannot be written solely in terms of previous production quantities because the work arrivals are now functions of completed jobs, new jobs, and instructions per job. The resulting formula is not a recursion equation, and cannot be analyzed directly.

The trick we will use is that we can, with some difficulty, derive linear recursion equations for the expectation and variances of production. (Note that we will not be able to calculate the variances exactly, but we will be able to find closed bounds on the variances.) Consequently, the moment recursion equations are amenable to the infinite-iteration techniques we have used in previous chapters, and yield the steady-state expected production quantities and bounds on the steady-state production variances.

Section 2 gives an overview of the derivation of the LRS-MR model. Section 3 describes an example model of a data-processing network. The example illustrates ways to analyze job shops (including data-processing networks) with the LRS-MR model, and shows how the LRS-MR model's results may be used to improve job shop operations. Section 4 further discusses the LRS-MR model and its assumptions, and describes possible generalizations to the model. Finally, Section 5 derives the LRS-MR model in detail. (Section 6 is an appendix that presents the lemmas used in Section 5.)

2. An Overview of the Model Development

2.1 Model Assumptions

The LRS-MR model's input data are:

- The expectation and variance of the number of *requests* entering each station from outside the job shop.
- The expectation and variance of the number of *jobs* each station j will process per request from station k, for all pairs of stations (j,k).
- The expectation and variance of the amount of work per job at each station. The work per job is measured in *instructions*.

The model produces the following outputs:

- The expectation and variance of the workload at each station in the system, measured in instructions per time period.

To calculate these outputs, the LRS-MR model makes the following assumptions about station behavior.

- Every station operates in discrete time. All incoming work arrives at the start of a period. A set fraction of the station's work-in-progress is processed during that time period.
- Incoming work is characterized as follows:
  - Stations receive a random number of *requests* at the start of each period. Each incoming request is converted into a random number of *jobs*. (Note that a request can be converted into zero jobs.)
  - Each incoming job becomes a random number of *instructions*.
  - The total work in instructions is added to the station's *work queue*.
- Job requests are characterized as follows:
  - Whenever a station completes a job, it sends a request to all the stations immediately downstream of it. Formally, for each job that station k completes in period t, station k sends one request to each immediately-downstream station at the start of period $t+1$. These transfers are called *internal requests*.
  - Stations also receive requests from outside the system. These are called *external requests*. External requests come from a random distribution with an expectation and variance that do not vary over time.
  - Incoming requests may be treated differently depending upon their source. Mathematically, the distribution that converts requests sent from station k to station j into jobs may differ from the distribution that converts requests sent from station l to station j. The random distributions that convert requests into jobs do not vary over time. Further, these random distributions do not depend on the past history of the requests received.

The following figure shows how the LRS-MR model calculates the work done at a single station in a single time period.

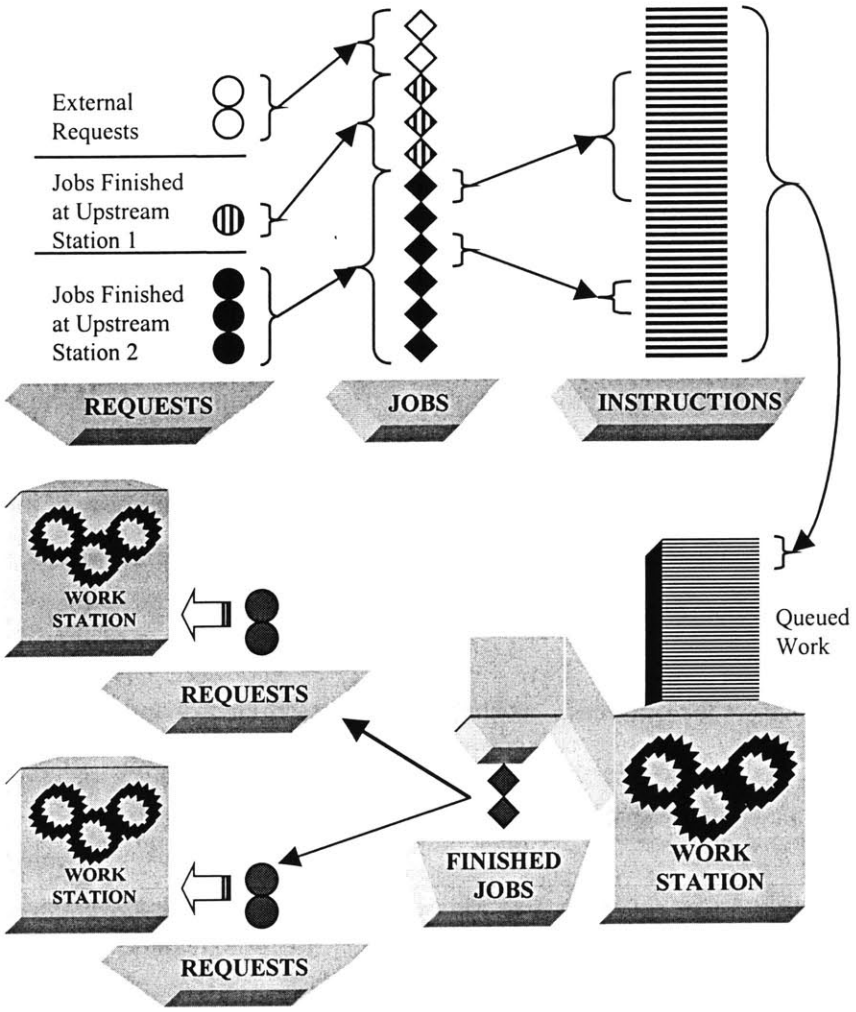


Figure 17 -- How the LRS-MR Model Determines a Station's Workload

At the start of a time period, the station receives requests from two sources. First, it receives two requests from outside the network. It also receives one request from upstream station 1 and three requests from upstream station 2. This means that in the previous period station 1 completed one job and station 2 completed three jobs.

Next, the station converts the requests into a random number of jobs. The distribution that converts requests into jobs varies with the source of the request. Here, the station turned every external request into one job, every request from station 1 into three jobs, and every request from station 2 into two jobs.

The station then converts each job into a random number of instructions. The station uses the same distribution to convert jobs into instructions for all jobs, regardless of where the jobs came from. However, the number of instructions per job can vary substantially, as shown in the figure. The resulting block of instructions is added to the station's work queue.

Finally, the station completes some specified amount of the work in its queue during the time period. Each time the station completes enough work to finish a job, it sends a request to all of its downstream stations. The downstream stations then convert these requests into work at the start of the next time period.

- Each station in the network obeys one of the following control rules.
  - *Instruction control*: each time period, station k processes $1 / L_k$ of the total work (in instructions) at its queue, for some constant L_k. The constant L_k is called the *leadtime* at station k.
  - *Job control*: each time period, station k processes $1 / L_k$ of the number of jobs in its queue.
  - Note: we assume that stations always process their specified workload (no production errors). A station's workload always equals its *production*.

The following figure shows the difference between the instruction-control and job-control rule.

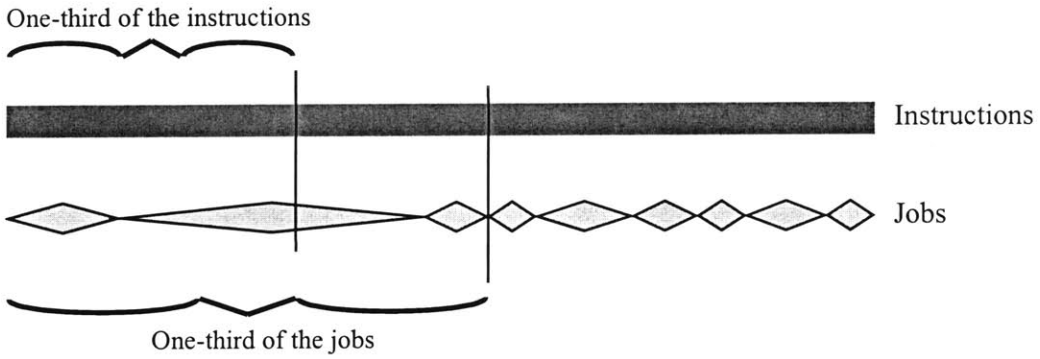


Figure 18 -- Comparison of the Two Control Rules

Here, we see how the station can process very different amounts of work under the two rules, even using the same parameter (lead time set to three). We also see the advantages and disadvantages of the rules. With the instruction-control rule, we smooth out the work done at the station. However, the number of jobs produced each period can vary substantially, which will lead to greater fluctuations in the work at the downstream stations. With the job-control rule, the amount of work done each period varies more, but we smooth out the number of jobs the station produces each period. This will lead to lower fluctuations in the work at the downstream stations. Clearly, we will be interested in determining where to use each type of rule.

Further, we will be interested in two variations of the instruction-control rule. In the first case, we simply process the required fraction of instructions each time period, regardless of where the breakpoints in the jobs are. In second case, we reorder the jobs to process the required fraction of both jobs and instructions in the same period. We expect that the latter rule will be quite complicated to implement and will impose heavy costs on the system, but will smooth work both at the current station and at downstream stations. We will be interested in knowing where to install this "double-control" rule.

2.2 Variable Declarations and Assumptions

The LRS-MR model uses a series of equations to calculate statistics for the station workloads. These equations use the following input variables.

- $R^e_{j,t}$ is a random variable for the number of external requests entering station j from outside the network at the start of period t. This variable is independent from the time period, the number of internal requests, and the work levels at all other stations. $E[R^e_{j,t}]$ and $Var[R^e_{j,t}]$ are inputs to the model.
- $J_{j,k}$ is a random variable for the number of jobs per request sent from station k to station j. It is independent from the time period, the number of requests, and the work levels at all other stations. $E[J_{j,k}]$ and $Var[J_{j,k}]$ are inputs to the model.
- $J_{j,R}$ is a random variable for the number jobs per request entering station j from outside the system. It is independent from the time period, the number of requests, and the work levels at all other stations. $E[J_{j,R}]$ and $Var[J_{j,R}]$ are inputs to the model.
- X_j is a random variable for the number of instructions per job. It is independent from the time period, the number of requests, the number of jobs, and the work levels at all other stations. $E[X_j]$ and $Var[X_j]$ are inputs to the model.

The equations of the LRS-MR model use the following "intermediate" variables as well.

- $P_{j,t}$ is the number of instructions processed by station j in period t.
- $Q_{j,t}$ is the length of station j's work queue (in instructions) at the start of period t.
- $A_{j,k,t}$ is the work, in instructions, that station k sends to station j at the start of period t.
- $A_{j,t}$ is the total work, in instructions, that arrives at station j at the start of period t.
- $N_{j,t}$ is the number of jobs completed by station j during period t.
- $q_{j,t}$ is the number of jobs in station j's work queue at the start of period t.

The model estimates the following statistics for each station:

- $E[P_j]$, the expected work per period (in instructions) in steady state.
- $Var[P_j]$, the variance of the work per period in steady state.

2.3 Model Mechanics

The strategy behind the LRS-MR model's derivation is as follows: We assume we know the expected workload, workload variances, and covariances for all stations in time *t-1*. We then find an equation for the expected workload and variance at each station in time *t*. We then write all of these "*t-1* to *t*" equations simultaneously in matrix form, and analyze infinite recursions of the resulting matrix equations. The result is a set of matrix formulas that calculate the expected workloads and estimate the workload variances for all stations.

We develop the equations differently for stations using the instruction-control rule or the job-control rule.

- Under the instruction-control rule, a station processes a fixed fraction of its queued instructions each time period. Consequently, the model equations keep track of the work done in instructions at the station. This rule's advantage is that the equations' results are the desired results, $E[P_j]$ and $Var[P_j]$. The disadvantage, however, is that we must find ways to convert results measured in instructions to results measured in jobs to properly calculate the work requests sent to downstream stations. Consequently, we will be unable to precisely calculate the variances at the stations.
- Under the job-control rule, a station processes a fixed fraction of its queued jobs each time period. Consequently, the model equations keep track of the work done in jobs at each station. The advantage of this rule is that we can represent work requests sent to downstream stations directly in terms of the jobs completed at the station. The disadvantage is that we must convert results measured in jobs into results measured in instructions to calculate $E[P_j]$ and $Var[P_j]$. However, we can use fairly simple formulas to do this.
- Finally, we will need sets of equations to relate work done at stations using the instruction-control rule to stations using the job-control rule, and vice versa.

We first describe the mechanics of stations using the instruction-control rule, and then describe the mechanics of stations using the job-control rule.

2.3.1 Instruction-Rule Mechanics

Work Arrivals. Let station k be upstream of station j. Station k sends $N_{k,t-1}$ requests to station j at the start of period t. Further, recall that each request is converted into a random number of jobs. Then, each job is

converted into a random number of instructions. Then the work, in instructions, that station k sends to station j at the start of period t is:

$$A_{j,k,t} = \sum_{l=1}^{N_{k,t-1}} \sum_{m=1}^{J^l_{jkt}} X^{lm}_{jt} \tag{1}$$

(As stated, $J_{j,k}$, $J_{j,R}$, and X_j do not depend on t. The t-indices are included to clarify the summation.) In words, (1) says the number of work instructions arriving to station k is the number of instructions per job, X_j, summed over the number of jobs per request, $J_{j,k}$, and summed over the total number of requests, $N_{k,t-1}$.

Now, let station j receive input from a subset of stations, K. (Note that K can include j, since a station can send requests to itself.) The total arrival (in instructions) to j at the start of period t, as a function of all the requests received from the stations in K plus the external requests, is:

$$A_{j,t} = \sum_{k \in K} \sum_{l=1}^{N_{k,t-1}} \sum_{m=1}^{J^l_{j,k,t}} X^{klm}_{jt} + \sum_{l=1}^{R^e_{j,t}} \sum_{m=1}^{J^l_{j,R,t}} X^{Rlm}_{jt} \tag{2}$$

Control rule. Each period, station j processes $1/L_j$ of the work in its queue. Mathematically, this is $P_{j,t} = Q_{j,t} / L_j$. We relate the queue length at period t to the queue length at t-1. We have:

$$Q_{j,t} = \underbrace{Q_{j,t-1}}_{\text{Queue last period}} - \underbrace{P_{j,t-1}}_{\substack{\text{Instructions} \\ \text{processed last period}}} + \underbrace{A_{j,t}}_{\substack{\text{Arrivals (in instructions)} \\ \text{this period}}} \tag{3}$$

Now, substitute $Q_{j,t} = L_j P_{j,t}$ into the preceding equation, and solve for $P_{j,t}$.

$$\begin{aligned} &\Rightarrow L_j P_{j,t} = L_j P_{j,t-1} - P_{j,t-1} + A_{j,t} \\ &\Rightarrow P_{j,t} = \left(1 - \frac{1}{L_j}\right) P_{j,t-1} + \frac{1}{L_j} A_{j,t}. \end{aligned} \tag{4}$$

Finally, substitute in for $A_{j,t}$. This gives us a recursive formula for $P_{j,t}$:

$$P_{j,t} = \left(1 - \frac{1}{L_j}\right) P_{j,t-1} + \frac{1}{L_j} \left(\sum_{k \in K} \sum_{l=1}^{N_{k,t-1}} \sum_{m=1}^{J^l_{j,k,t}} X^{klm}_{jt} + \sum_{l=1}^{R^e_{j,t}} \sum_{m=1}^{J^l_{j,R,t}} X^{Rlm}_{jt} \right). \tag{5}$$

Equation 1-- Recursive Formula for the Work at Station j

2.3.2 Job-Rule Mechanics

Measurement of Work. Under the job-shop rule, the model equations track the work at all of the stations in terms of jobs, not instructions. Consequently, the equations under the job-shop rule make no reference to instructions. In calculating the results of the model, we will first compute the expectation and variances of the workloads in jobs per period. Then, we will apply formulas for the expectation and variance of a random sum of random variables to calculate the expectation and variance of the workloads in instructions.

Work Arrivals. Let station k be upstream of station j. Station k sends $N_{k,t-1}$ requests to station j at the start of period t, each of which is then converted into a random number of jobs. Then the work, in jobs, that station k sends to station j at the start of period t is:

$$a_{j,k,t} = \sum_{l=1}^{N_{k,t-1}} J_{jkt}^{l} \tag{6}$$

Now, let station j receive input from a subset of stations, K. (Note that K can include j, since a station can send requests to itself.) The total arrival (in instructions) to k at the start of period t, as a function of all the requests received from the stations in K plus the external requests, is:

$$a_{j,t} = \sum_{k \in K} \sum_{l=1}^{N_{k,t-1}} J_{j,k,t}^{l} + \sum_{l=1}^{R_{j,t}^{e}} J_{j,R,t}^{l} \,. \tag{7}$$

Control rule. Each period, station j processes $1/L_j$ of the work (in jobs) in its queue. Mathematically, this is $N_{j,t} = q_{j,t} / L_j$. We relate the queue length at period t to the queue length at t-1. We have:

$$q_{j,t} = \underbrace{q_{j,t-1}}_{\substack{\text{Queue in jobs} \\ \text{last period}}} - \underbrace{N_{j,t-1}}_{\substack{\text{Jobs processed} \\ \text{last period}}} + \underbrace{a_{j,t}}_{\substack{\text{Arrivals (in jobs)} \\ \text{this period}}} \,. \tag{8}$$

Now, substitute $q_{j,t} = L_j N_{j,t}$ into the preceding equation, and solve for $N_{j,t}$.

$$\begin{aligned} &\Rightarrow L_j N_{j,t} = L_j N_{j,t-1} - N_{j,t-1} + a_{j,t} \\ &\Rightarrow N_{j,t} = \left(1 - \frac{1}{L_j}\right) N_{j,t-1} + \frac{1}{L_j} a_{j,t} \,. \end{aligned} \tag{9}$$

Finally, substitute in for $a_{j,t}$. This gives us a recursive formula for $P_{j,t}$:

$$N_{j,t} = \left(1 - \frac{1}{L_j}\right) N_{j,t-1} + \frac{1}{L_j}\left(\sum_{k \in K} \sum_{l=1}^{N_{k,t-1}} J^l_{j,k,t} + \sum_{l=1}^{R^e_{j,t}} J^l_{j,R,t}\right). \tag{10}$$

Equation 2-- Recursive Formula for the Jobs Processed at Station *j*

2.3.3 Discussion of the Mathematical Derivations

Objective. We want to calculate the following statistics for all stations j in the job shop:

- $E[P_j]$, the expected work per period (in instructions) in steady state.
- $Var[P_j]$, the variance of the work per period in steady state.
- $Cov[P_i, P_j]$, the covariance of the work between two stations in steady state.

The calculations of these statistics are complicated, and are listed in their entirety in section 5. However, an overview follows. We first assume a model in which all stations use only the instruction-control rule. To calculate these values, we use Equation 1 to find recursive relationships for $E[P_{j,t}]$ and $Var[P_{j,t}]$, at time t, provided that we know these values (along with the covariance values) at time $t - 1$.

Calculating the relationship for expected workloads (Section 5.2) uses results concerning the linearity of expectations and the expectation of a random sum of random variables, and is straightforward. (The appendix presents formulas for the expectation and variance of a random sum of random variables.) Calculating the relationship for the variances of the workloads (Section 5.3) uses results concerning the linearity of variances, and the variance of a random sum of random variables. Calculating the variance is complicated, since the applicable formulas themselves are complicated. Further, it is difficult to convert the variance of the station's workload (measured in instructions) to the variance of the number of jobs produced. We will assume that the number of instructions per job is exponentially distributed, which implies that the breakpoints between jobs are uniformly distributed throughout the station's work queue (see appendix). We will see how the uniform-distribution assumption allows us to convert, approximately, the workload variances (in instructions) to the variance of the completed jobs.

Once we derive the recursive relationships for $E[P_{j,t}]$ and $Var[P_{j,t}]$, we will first show how to write all of the relationships for all stations simultaneously in matrix forms. We will then iterate the matrix relationships to derive steady-state values of $E[P_j]$, $Var[P_j]$ and $Cov[P_i, P_j]$. The expectations are calculated in section 5.2, and the variances are calculated in section 5.3. The covariances are calculated in section 5.10. (The covariances derivation is a general proof that applies to networks with stations using either or both control rules.) The results of these derivations are summarized below.

2.4 *Input Matrices and Vectors*

The following matrices and variables are used to calculate the steady-state expected workloads and workload variances.

2.4.1 The Workflow Matrix

Each entry of the workflow matrix, Φ_{jk}, shows the expected amount of work that arrives at station j for each unit of work at station k. It has the following entries, depending on the control rules used at stations j and k:

- If station k sends no work to station j, $\Phi_{jk} = 0$.
- If stations j and k both use instruction control, $\Phi_{jk} = E[J_{j,k}]E[X_j]/E[X_k]$. By definition, this term presents the expected number of instructions appearing at station j, given one instruction completed at station k. This term comes from taking the expectation of equation (1), the formula for $A_{j,k,t}$, and writing the resulting expression in terms of the instructions produced at station k. (Details are given in Section 5.)
- If stations j and k both use job control, $\Phi_{jk} = E[J_{j,k}]$.
- If station j uses job control, and station k uses instruction control, $\Phi_{jk} = E[J_{j,k}]/E[X_k]$.
- If station j uses instruction control, and station k uses job control, $\Phi_{jk} = E[J_{j,k}]E[X_j]$

2.4.2 The Input Covariance Matrix

The input covariance matrix is a square diagonal matrix. Each entry, Σ_{jj}, shows terms used to calculate the workload variance at station j that do not depend on the variances of other stations.

The Σ_{jj} entries are calculated as follows:

- If station j uses instruction control,

$$\Sigma_{jj} = \sum_{k \in K} \frac{E[P_k]}{E[X_k]} \left(E[J_{j,k}]Var[X_j] + Var[J_{j,k}](E[X_j])^2 \right)$$
$$+ E[R_j^e]\left(E[J_{jR}]Var[X_j] + Var[J_{jR}](E[X_j])^2 \right) + Var[R_j^e]\left(E[J_{jR}]E[X_j] \right)^2 .$$

- If station j uses job control,

$$\Sigma_{jj} = \sum_{k \in K} \left(\frac{E[P_k]Var[J_{j,k}]}{E[X_k]} \right) + E[R_j^e]Var[J_{jR}] + (E[J_{jR}])^2 Var[R_j^e] .$$

Note that these expressions use expected work quantities. We can calculate the expected work, before we calculate the work variances, so we can use the expected work as inputs to the variance equations.

The above expressions are quite complicated, and do not have as simple explanation. As noted the Σ_{jj}'s present the terms needed to calculate the work variances that do not depend on work variances at other stations. The instruction control Σ_{jj} contains many terms of the variance of the right hand side of (5), which presents $P_{j,t}$ in terms of $P_{j,t-1}$ and the incoming work (measured in instructions). The job control Σ_{jj} contains many terms of the variance of the of the right hand side of (7) which presents $P_{j,t}$ in terms of $P_{j,t-1}$ and the incoming work (measured in jobs). For both Σ_{jj}'s, the first summation represents the variance resulting from work coming from other stations, and the second and third terms represent the variance from work coming from outside the network. (Section 5 presents detailed derivations of the Σ_{jj}'s.)

2.4.3 The Flow Correction Matrix

The Flow Correction Matrix, **U**, is a square diagonal matrix. Its entries, $\mathbf{U}_{jj}$, show "fudge factors" that convert workload variances given in terms of instructions to workload variances given in terms of jobs. The formulas are as follows:

- If station j uses the job-control rule, $\mathbf{U}_{jj} = 0$.

If station j uses the instruction-control rule, there are two possible cases.

- A lower bound on the correction term is:

$$U_{jj} = -\frac{E[P_j]Var[X_j]}{L_j E[X_j]}.$$

This term approximates using the "double-control" rule at station j, where we try to process the same fraction of jobs and instructions each period.

- An upper bound on the correction term is:

$$U_{jj} = \left(\frac{1}{L_j}\right)\left(1-\frac{1}{L_j}\right)\left(L_j E[P_j]\right) - \frac{E[P_j]Var[X_j]}{L_j E[X_j]}.$$

This term approximates using a rule in which we ignore the placement of jobs in the work queue, and for which the instructions per job is exponentially distributed.

2.4.4 The Lead Time Matrix

The Lead Time Matrix, **D**, is a square diagonal matrix with the inverses of the lead times of all the stations on the diagonal. Consequently, $\mathbf{D}_{jj} = 1 / L_j$.

2.4.5 The Vector of External Inputs

The entries of this vector, μ_j, show the amount of work (either in instructions of jobs, depending on the control rule) that enter station j each period.

- If station j uses job control, $\mu_j = E[R_j^e]E[J_{jR}]$.
- If station j uses instruction control, $\mu_j = E[R_j^e]E[J_{jR}]E[X_j]$.

2.5 Model Equations and Results

The following formulas calculate the steady-state expected workloads and workload variances.

2.5.1 Primary Matrices

Compute the following vector ρ and matrix **S**:

- $\rho = (\mathbf{I} - \Phi)^{-1}\mu$, where **I** is the identity matrix.
- $\mathbf{S} = \sum_{s=0}^{\infty} \mathbf{B}^s((\mathbf{D}\Phi)\mathbf{U}(\mathbf{D}\Phi)'+\mathbf{D}\Sigma\mathbf{D})\mathbf{B}'^s$, where $\mathbf{B} = \mathbf{I} - \mathbf{D} + \mathbf{D}\Phi$.

We use ρ and **S** to calculate the expected workloads and workload variances. Section 5 derives these results.

In practice, ρ is calculated directly, while **S** is approximated by a finite series. Graves [1986] approximates a matrix very similar to **S** using the following technique. Define $\mathbf{S}_n$ to be the sum of the first n terms. Then:

(11) $$\mathbf{S}_{2n} = \mathbf{B}^n\mathbf{S}_n\mathbf{B}'^n + \mathbf{S}_n.$$

By repeated application of the above expression, we get a very good estimate of **S**; for example, seven iterations gives the sum of the first 128 terms in the series.

2.5.2 Expected Workloads

- If station j uses instruction control, $E[P_j] = \rho_j$.
- If station j uses job control, $E[P_j] = E[X_j]\rho_j$.

2.5.3 Workload Variances

Unlike the expected workloads, these are estimates.

- If station j uses instruction control, the estimate of $Var[P_j] = \mathbf{S}_{jj}$. This estimate is a lower bound if the lower bound $\mathbf{U}_{jj}$'s are used for all instruction control stations, and an upper bound if the upper bound $\mathbf{U}_{jj}$'s are used for all instruction control stations.
- If station j uses job control, the estimate of $Var[P_j] = \rho_j Var[X_j] + (E[X_j])^2 \mathbf{S}_{jj}$.

2.5.4 Workload Covariances

Again, these are estimates, not exact calculations.

- If stations j and k both use instruction control, the estimate of $Cov[P_j, P_k] = \mathbf{S}_{jk}$.
- If station j uses instruction control and station k uses job control, the estimate of $Cov[P_j, P_k] = \mathbf{S}_{jk} E[X_k]$. This estimate is a lower bound if the lower bound $\mathbf{U}_{jj}$'s are used for all instruction control stations, and an upper bound if the upper bound $\mathbf{U}_{jj}$'s are used for all instruction control stations.
- If stations j and k both use job control, the estimate of $Cov[P_j, P_k] = \mathbf{S}_{jk} E[X_j] E[X_k]$.

2.6 Some Useful Modeling Techniques

This subsection discusses two useful modeling techniques used in the example model. They are: randomly sending a job to one station in a group of stations, and processing a "follow-up" job.

2.6.1 Randomly Sending a Job to a One Station Out of a Group of Stations

In some job shop environments, a "broker" station assigns jobs to one (or more) stations out of a group of stations. (For example, callers to a technical support line will be directed to one technician out of a group of technicians.) We need to describe this distribution of jobs to the servers in analytic-model terms. This means characterizing the jobs per request, or J_{jk} distribution, for requests sent from broker k to server j. We model the arrivals to each service provider as Bernoulli. Each request that exits the "Broker" workstation k will have some probability, p_{jk}, of being sent to provider j.

This formulation is not quite accurate. At first, we might assume that each request that exits the Broker would be sent to one service provider. The Bernoulli model allows the same request to be sent to multiple stations or no stations. Further, the formulation also assumes that requests are distributed randomly to the providers. This will not be the case if "broker" stations send jobs to the least busy servers. Consequently, the Bernoulli model overstates the variance at the servers somewhat. Generally, however, the overstatement should be acceptable.

Under the Bernoulli model, the number of jobs per broker request is a random sum of independent random variables. It has the following form: $J_{jk} = \sum_{i=1}^{b_{jk}} j_{jk}$, where b_{jk} is a Bernoulli variable with possible values 0 and 1. Further, b_{jk} has parameter p_{jk}, the probability that a request leaving station k is sent to service provider j (namely, the probability that $b_{jk} = 1$. Finally, j_{jk} is a random variable for the number of jobs per request that arrives at station j. In words, this formulation says that station j receives jobs j_{jk} with probability p_{jk}, and nothing with probability $1 - p_{jk}$.

To find the moments of the jobs per request, E[J_{jk}] and Var[J_{jk}], we apply formulas for the expectation and variance of a random sum of random variables (see Appendix) to find:

$$E[J_{jk}] = p_{jk} E[j_{jk}], \text{ and}$$
$$Var[J_{jk}] = p_{jk} Var[j_{jk}] + p_{jk}(1 - p_{jk})(E[J_{jk}])^2. \quad (12)$$

2.6.2 Follow-Up Jobs

In a job-shop, the completion of a job can randomly generate a new "follow-up" request at an upstream station. Here, we want to model, J_{jk}, the distribution of the requests at the upstream station j per job completed at provider k.

Again, we use a Bernoulli model. This time, p_{jk} represents the probability that a service request leaving station k will generate a new request at upstream station j. Since we are using the same model, we get the same formulas for the input statistics for E[J_{jk}] and Var[J_{jk}]:

$$E[J_{jk}] = p_{jk} E[j_{jk}], \text{ and}$$
$$Var[J_{jk}] = p_{jk} Var[j_{jk}] + p_{jk}(1 - p_{jk})(E[J_{jk}])^2. \quad (13)$$

3. An Example Job Shop

3.1 An Example Model – A Data-Processing Network

Figure 19 shows an example data processing network. In this network, computing jobs first go through one of two chains of *control stations*. The control stations determine the work that needs to be done to complete a job, and then assign *service providers* (which can be thought of as computation boxes) to perform the work. When completed, a fraction of the jobs immediately create new computational jobs. These control stations process the new jobs, and the cycle continues.

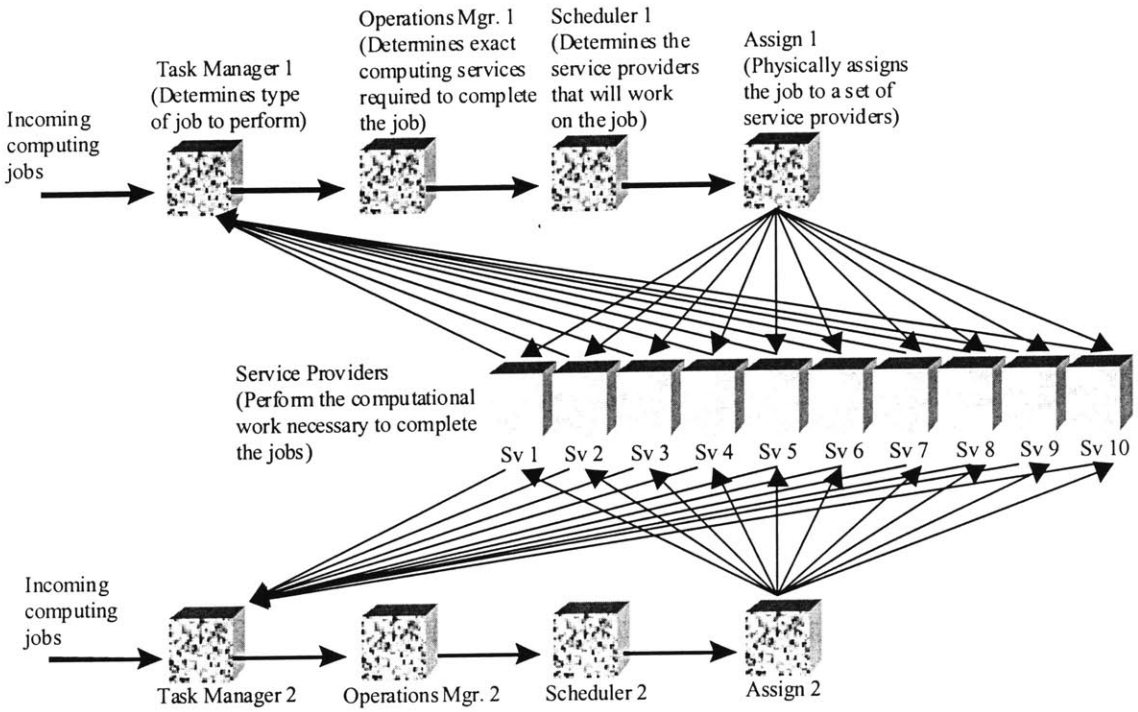


Figure 19 -- Diagram of Example Data-Processing Network

This is a fairly simple network, although it closely resembles an actual data-processing network design studied by the author. In the following calculations, we assume that we already know the overall probability that a request is sent to any service provider, as well as the expectation and variance for all requests sent to that provider. Further, we assume that no service provider ever rejects a job sent to it. (Note that the techniques of the previous section model the distribution of jobs to the service providers, and the flow of jobs back to the Task Manager stations.)

Table 5 shows the input statistics for this job shop, including the number of requests entering each control chain per second, the number of jobs per request at each station, and the number of instructions per job. There are two "Jobs per Request" values for each service station; each value is the probability that a request from a control chain is sent to that service station. (Note that there is a 10% probability that jobs sent through chain 2 will be canceled at the end of the chain.) It is assumed that service stations process one job per received request. Note the high values for the instructions per job, as compared with the number of jobs moving through the network.

Table 5 -- Input Statistics for the Example Model

Station	Jobs per Request		Instructions per Job	
	Expected	Variance	Expected	Variance
First Control Chain (C1)				
Service Requests Entering System per Second	N/A	N/A	10 requests per second	4
1. Task Manager 1	1	0	1,000	100

Station	Jobs per Request		Instructions per Job	
	Expected	Variance	Expected	Variance
2. Operations Mgr. 1	1.5	0.2	1,000	64
3. Scheduler 1	1	0	5,000	40,000
4. Assign 1	1	0	10,000	4,000,000
Second Control Chain (C2)				
Service Requests Entering System per Second	N/A	N/A	20 requests per second	16
5. Initialize 2	1	0	1,000	100
6. Operations Mgr. 2	1.1	0.1	1,000	64
7. Scheduler 2	1	0	5,000	40,000
8. Assign 2	1	0	10,000	4,000,000
Service Providers				
9. Service Station 1	C1: 0.20 C2: 0.25	C1: 0.16 C2: 0.1876	5,000	4,000,000
10. Sv 2	C1: 0.05 C2: 0.07	C1: 0.0475 C2: 0.0651	50,000	36,000,000
11. Sv 3	C1: 0.05 C2: 0.03	C1: 0.0475 C2: 0.0291	100,000	100,000,000
12. Sv 4	C1: 0.10 C2: 0.05	C1: 0.09 C2: 0.0475	10,000	25,000,000
13. Sv 5	C1: 0.30 C2: 0.20	C1: 0.21 C2: 0.16	3,000	10,000
14. Sv 6	C1: 0.05 C2: 0.04	C1: 0.0475 C2: 0.0384	100,000	400,000,000
15. Sv 7	C1: 0.05 C2: 0.06	C1: 0.0475 C2: 0.0564	150,000	2,500,000,000
16. Sv 8	C1: 0.05 C2: 0.06	C1: 0.0475 C2: 0.0564	75,000	144,000,000
17. Sv 9	C1: 0.05 C2: 0.04	C1: 0.0475 C2: 0.0384	250,000	6,400,000,000
18. Sv 10	C1: 0.10 C2: 0.10	C1: 0.09 C2: 0.09	70,000	400,000,000

Using these input statistics and the LRS-MR model equations, we next derive some useful results about this network.

3.1.1 The Effect of Increasing Lead Times

Table 6 shows some basic results from the model. The first column shows the expected workload at each station. The second column shows the standard deviation of the workload at each station when all the lead times are set to one. The third column shows the standard deviations of workload when all the lead times are set to three, and all the stations use the "double control rule," in which the stations process one-third of both the instructions and jobs in their queues each period. The fourth column compares the standard deviations for the two lead times.

Three notes about the following table:

- The expected workloads do not depend on the lead times.
- When the lead times are set to one, all stations process all of the work in their queues every time period. In this case, there is no difference between the control rules.

- The double-control rule generally produces lower standard deviations than the other two rules, so the comparison between the one-period and three-period results is the most favorable possible. Other rule choices will lead to somewhat inferior comparisons.

Station	Expected Workload (Instructions / second)	Standard Deviation with Lead Times of One	Standard Deviation with Lead Times of Three	% Improvement by Setting Lead Times to Three
First Control Chain (C1)				
1. Task Manager 1	17,720	3,610	1,520	57.9%
2. Operations Mgr. 1	26,580	5,740	1,900	66.9%
3. Scheduler 1	132,910	28,710	8,110	71.8%
4. Assign 1	265,830	58,350	15,620	73.2%
Second Control Chain (C2)				
5. Initialize 2	25,150	4,660	2,060	55.8%
6. Operations Mgr. 2	27,660	5,370	1,790	66.7%
7. Scheduler 2	138,310	26,870	7,610	71.7%
8. Assign 2	276,630	54,750	14,650	73.2%
Service Providers				
9. Service Station 1	61,160	19,300	7,910	59.0%
10. Sv 2	163,280	91,680	39,940	56.4%
11. Sv 3	215,900	148,650	65,240	56.1%
12. Sv 4	40,410	22,710	9,860	56.6%
13. Sv 5	40,520	11,470	4,550	60.3%
14. Sv 6	243,570	160,200	70,270	56.1%
15. Sv 7	448,340	275,140	120,420	56.2%
16. Sv 8	224,170	132,240	57,770	56.3%
17. Sv 9	608,910	412,190	181,000	56.1%
18. Sv 10	379,720	171,750	73,740	57.1%

Table 6 – Example Results v. Lead Times

In this "best case" scenario, we see that increasing the lead times to three at all stations lowers the standard deviations by fifty-five percent to seventy-three percent. The price of this reduction, however, is a three-fold increase in the expected time to complete a service request. (Recall that the lead time at a station is the expected time to complete a job at the station). Further, these reductions come from using the most sophisticated control rule at each station, the double-control rule. There may be substantial system costs for using the double-control rule.

These lead time / standard deviation tradeoffs suggest a use of the LRS-MR model. We know from queuing theory that the lower the variance of the work flows, the lower the stations' service rates will need to be to meet a given level of performance (as measured in expected waiting times). We can input a set of desired lead times into the LRS-MR model, and use the model results in simple simulations to determine the service capacities needed at each station. This use will be examined in Chapter 6, on Optimization.

3.1.2 The Effects of Using Different Control Rules

The variances of demand vary with the choice of the control rule. Table 3 compares the standard deviations of demand under four different scenarios:

- *Double Rule*: Each station uses the double-control rule.
- *Instruction Rule*: Each station uses the instruction-control rule.
- *Job Rule*: Each station uses the job-control-rule.
- *Combined Rule*: All of the control stations use the job-control rule. All of the service stations use the instruction-control rule. (The motivation for this combined rule is explained below.)

The lead times are set to three for all rules at all stations. For each rule, the table lists the numerical values for the standard deviations, and percentage differences between the rule and the double-control rule. As with the lead times, the stations will have the same expected workloads under all control rules, so the expected workloads are not reprinted.

Table 7 -- Standard Deviations v. Control Rules

Station	Double Rule	Instruction Rule		Job Rule		Combined Rule	
	Standard Deviation	Standard Deviation	% Increase	Standard Deviation	% Increase	Standard Deviation	% Increase
First Control Chain (C1)							
1. Task Manager 1	1,520	1,660	9.2%	1,520	0.0%	1,570	3.3%
2. Operations Mgr. 1	1,900	3,110	63.7%	1,890	-0.5%	1,950	2.6%
3. Scheduler 1	8,100	15,520	91.5%	8,130	0.2%	8,370	3.3%
4. Assign 1	15,580	32,530	108.4%	18,060	15.6%	18,450	18.2%
Second Control Chain (C2)							
5. Initialize 2	2,050	2,100	1.9%	2,050	-0.5%	2,070	0.5%
6. Operations Mgr. 2	1,790	2,730	52.5%	1,790	0.0%	1,810	1.1%
7. Scheduler 2	7,600	14,390	89.2%	7,660	0.7%	7,720	1.4%
8. Assign 2	14,630	30,830	110.4%	17,380	18.6%	17,470	19.3%
Service Providers							
9. Service Station 1	7,900	9,270	17.3%	10,070	27.3%	7,910	0.1%
10. Sv 2	39,920	42,000	5.2%	41,070	2.8%	39,940	0.1%
11. Sv 3	65,220	67,620	3.7%	66,510	1.9%	65,240	0.0%
12. Sv 4	9,850	10,430	5.9%	13,340	35.3%	9,860	0.1%
13. Sv 5	4,540	5,620	23.7%	4,540	-0.2%	4,550	0.2%
14. Sv 6	70,240	72,920	3.8%	75,570	7.5%	70,270	0.0%
15. Sv 7	120,370	125,520	4.3%	143,030	18.8%	120,410	0.0%
16. Sv 8	57,750	60,420	4.6%	60,640	5.0%	57,770	0.0%
17. Sv 9	180,940	187,450	3.6%	212,580	17.4%	181,000	0.0%
18. Sv 10	73,680	79,630	8.1%	84,610	14.7%	73,740	0.1%

The instruction-control rule creates much greater variances at the control stations than does the double-control rule (more than doubling the standard deviation at the brokering stations), and significantly greater variances at the service providers. The job-control rule performs substantially better than the

instruction-control rule for the initial-processing stations (even matching the double-control rule in some cases), but somewhat worse at the service stations.

These results suggest an alternate mixed-rule model, in which the job-control rule is used at the initial-processing stations and the instruction-control rule is used at the service providers. The results are shown in the last two columns. This mixed model does almost as well as the double-control rule for the control stations, and virtually matches its performance at the service stations. By using a combination of simpler control rules, we have almost matched the performance of much more complicated rules with much greater system costs. Determining where to make these control-rule assignments will help save system costs.

3.1.3 The Diminishing Returns of Increased Lead Times

As previously noted, increasing the lead times decreases the standard deviations of the workload at the stations. However, these decreases diminish as we continue to increase the lead times.

Table 8 shows the effects of these decreases. The table shows sets of integer lead times needed to lower the standard deviations of the workloads to be under 10% of the expected workloads, using the control rules discussed in the previous section. The table also compares the lead times under each rule to those of the double-control rule.

For reference, we used the following heuristic to find the control rules:

- We first found the integer lead times needed to reduce each control station's workload standard deviation sufficiently. We did so one station at a time, starting with the first station in each control chain, and working through the fourth station in each control chain.
- Keeping the controls stations' lead times fixed, we then found the integer lead times needed to reduce each service provider station's standard deviation sufficiently, again proceeding one station at a time.

Obviously, the heuristic is not guaranteed to find the smallest lead times that meet the standard deviation conditions. Nonetheless, the heuristic does take into account the facts that lowering workload standard deviations at one station will immediately lower workload standard deviations at downstream stations, and that the feedback requests from the service providers to the control elements are not great. The resulting lead times are likely quite close to the optimal set of lead times.

Table 8 – Required Lead Times v. Control Rules

Station	Double Rule	Instruction Rule		Job Rule		Combined Rule	
	Lead Time Needed	Lead Time	% Increase	Lead Time	% Increase	Lead Time	% Increase
First Control Chain (C1)							
1. Task Manager 1	3	3	0.0	3	0.0	3	0.0

Station	Double Rule	Instruction Rule		Job Rule		Combined Rule	
2. Operations Mgr. 1	2	4	100.0	2	0.0	2	0.0
3. Scheduler 1	1	5	500.0	1	0.0	1	0.0
4. Assign 1	1	5	500.0	1	0.0	1	0.0
Second Control Chain (C2)							
5. Initialize 2	3	3	0.0	3	0.0	3	0.0
6. Operations Mgr. 2	2	4	100.0	2	0.0	2	0.0
7. Scheduler 2	1	4	400.0	1	0.0	1	0.0
8. Assign 2	1	4	400.0	1	0.0	1	0.0
Service Providers							
9. Service Station 1	6	8	33.3	Infinite	N/A	6	0.0
10. Sv 2	17	20	17.6	29	70.5	17	0.0
11. Sv 3	25	28	12.0	43	72.0	25	0.0
12. Sv 4	17	20	17.6	Infinite	N/A	17	0.0
13. Sv 5	4	6	50.0	5	25.0	5	25.0
14. Sv 6	23	26	13.0	Infinite	N/A	23	0.0
15. Sv 7	20	23	15.0	Infinite	N/A	20	0.0
16. Sv 8	17	19	11.7	95	458.8	17	0.0
17. Sv 9	24	27	12.5	Infinite	N/A	24	0.0
18. Sv 10	11	14	27.3	Infinite	N/A	11	0.0

Note that the lead times at the service stations are much higher than we might expect given the big variance decreases we made by increasing the lead times from one to three. (The initial-processing station variances are low in comparison to their expected demands to begin with, so their required lead times are small.)

Also, choice of control rules can greatly affect the required lead times. As with the previous table, the "combined rule" (job-control rule at initial-processing stations, instruction-control rule at service stations) virtually matches the performance of using the double-control rule at all stations. The instruction-control rule by itself performs substantially worse than the double-control rule, especially at the control stations.

Further, note that the job-control rule by itself performs unacceptably in this case. Under the job-control rule, stations process fixed fractions of queued jobs, ignoring the variance of the instructions per jobs. There will always be some residual variance in the work processed each period, regardless of the lead times. Consequently, stations using the job-control rule have strictly positive lower bounds on their variances, which can make the job-control rule unsuitable. Here, the residual variances prevented us from reducing the workload standard deviation below ten percent of the expected workload at six of the ten service stations.

4. Discussion of the Model

In this section, we discuss the assumptions and limitations of the LRS-MR model, and suggest possible areas of future research. We also suggest possible uses for the LRS-MR model.

4.1 Fractional Control Rules and Resource Allocation Studies

As with the original Tactical Planning Model, one of the biggest limitations of the model appears to be its fractional control rules, which assume that there are no fixed constraints on the production levels each time period. An alternate control rule might require a station to process the minimum of the work proscribed by its fractional control rule and a maximum capacity. This rule appears more appropriate than the plain linear control rule for stations with well-defined maximum processing speeds, such as certain types of computers. Analyzing this *bounded control rule* analytically seems difficult.

However, we can ensure that stations have maximum capacities capable of processing their required workloads "most" of the time, so that stations will "usually" behave like they are modeled. For example, assume that all of the workload variances are normally distributed. Then, setting the maximum capacity of a station to be its expected workload plus twice its workload standard deviation implies that the station will be able to process its proscribed amount about 97% of the time. More generally, we may want to set the capacity of a station j to be:

$$P_j^{\max} = E[P_j] + k_j Var[P_j],$$

where k_j is a safety factor.

This relationship suggests a use of the LRS-MR model: varying lead times and capacity allocations so that the performance of a job shop (as measured by its lead times) meets both performance requirements and budget constraints on station capacities. One may develop mathematical programs, using the LRS-MR model as a subroutine, that approximately optimize capacity allocations to meet budget and / or performance constraints. We present mathematical programs using LRS-MR models in Chapter 6, on Optimization.

We may need to perform simulation studies, however, to find useful relationships between station capacities and the expected workloads and workload variances estimated by the LRS-MR model. Such simulations will be especially important for general cases in which job shops have large numbers of stations, and the distributions of the stations' workloads are not normally distributed. We present an example of these simulations in Chapter 6, as well, and show how these simulations generate functions yielding the required safety factors.

4.2 Markov Assumptions

The LRS-MR model also has the disadvantage that it is Markovian. We assume that work flows in the job shop can be modeled such that accounting for the history of a work flow is not necessary. In the general case, it does not seem possible to address this assumption without having models with

combinatorial complexity. However, there are some limited generalizations that may help address history-dependent work flows.

- Assume the work flows through the job shop can be decomposed into a few distinct job types, each with their own routings and production requirements. Assume further that the stations can be configured to process jobs of each type in each time period. Then we can create one LRS-MR model for each job type. Provided that these job-type flows are independent, the total expected loads and load variances on each station will be the sum of the statistics from the LRS-MR models.

- If the history dependencies are defined by a small number of sequential states (for example, work goes from station A to station B, then back to station A for reprocessing), we can create an LRS-MR model in which each state has one model node. Assuming that the stations can handle the work in each state through separate queues, the expected workload at a station is the sum of the expected workloads at the corresponding state nodes. The workload variance is the sum of the variances of the corresponding state nodes, plus the sum of the covariances between the state nodes.

Using these generalizations, we might study ways to allocate a station's capacity between different types of jobs and work states.

4.3 Model Decomposition

Finally, we consider instances in which we wish to model very large job shops. The overall complexity of the LRS-MR model is $O(n^3)$, where n is the number of stations modeled. This complexity results from inverting or multiplying n-by-n matrices to generate the model's results. While this complexity is excellent for small to medium-sized networks, it becomes impractical for stations with many thousands of nodes – especially if we attempt to use the LRS-MR model as an optimization subroutine.

Clearly, then, we might want to study ways to decompose the calculations of the LRS-MR model. Such decomposition techniques are especially important for "sparse" job shops (job shops with lots of nodes, but comparatively few work flows between them). We note that the LRS-MR model is a discrete-time model; the recursion equations produce the expected workloads and workload covariances in one period, given the expected workloads and workload covariances of the previous workload. Further, these calculations appear to converge quickly. Computational experience suggests that the model's covariance matrix converges to more than four significant digits between 32 and 64 periods; we expect results for expected workloads are similar. These facts suggest iterating the recursion equations directly, or in small

matrices covering sections of the job shop, to converge to the steady-state expected workloads and workload covariances.

5. The Derivation of the LRS-MR model

In this section, we present the derivation of the LRS-MR model, done in several steps.

- Sections 5.1 to 5.3 derive most of the model equations for networks in which every station uses the instruction-control rule, under which a station processes a fixed fraction of the number of instructions in its queue each time period. The sections derive the station mechanics under the instruction-control rule, the expected workloads, and the variances. The covariances between stations under the instruction-control rule are not be defined in these sections. Instead, the covariance equations under all rules are derived in section 5.10.
- Sections 5.4 to 5.6 derive most of the model equations for networks in which every station uses the job-control rule, under which a station processes a fixed fraction of the number of jobs in its queue each time period. The sections derive the station mechanics under the job-control rule, the expected workloads, and the variances.
- Sections 5.7 to 5.9 derive most of the model equations for mixed networks in which some stations use the instruction-control rule and others use the job-control rule. Section 5.7 explains the mechanics of sending work between stations using different rules. Section 5.8 derives the resulting expected workloads equations, and section 5.9 defines the variance equations.
- Section 5.10 derives formulas for the covariances between stations. It shows that, without any modifications, the matrix equations that calculate station variances also calculate the covariances correctly.

5.1 Instruction-Rule Mechanics

The following equations mathematically describe the behavior of the workstations under the instruction-control rule, as declared in Section 4.2.

Work Arrivals. Let station k be upstream of station j. Station k sends $N_{k,t-l}$ requests to station j at the start of period t. Further, recall that each request is converted into a random number of jobs, which is then converted into a random number of instructions. Then the work, in instructions, that station k sends to station j at the start of period t is:

(14)
$$A_{j,k,t} = \sum_{l=1}^{N_{k,t-1}} \sum_{m=1}^{J_{jkt}^{l}} X_{jt}^{lm}$$

(As stated, $J_{j,k}$, $J_{j,R}$, and X_j do not depend on t. The t-indices are included to clarify the summation.)

Now, let station j receive input from a subset of stations, K. (Note that K can include j, since a station can send requests to itself.) The total arrival (in instructions) to k at the start of period t, as a function of all the requests received from the stations in K plus the external requests, is:

$$A_{j,t} = \sum_{k \in K} \sum_{l=1}^{N_{k,t-1}} \sum_{m=1}^{J^l_{j,k,t}} X^{klm}_{jt} + \sum_{l=1}^{R^e_{j,t}} \sum_{m=1}^{J^l_{j,R,t}} X^{Rlm}_{jt} \tag{15}$$

Control rule. Each period, station j processes $1/L_j$ of the work in its queue. Mathematically, this is $P_{j,t} = Q_{j,t} / L_j$. We relate the queue length at period t to the queue length at t-1. We have:

$$Q_{j,t} = \underbrace{Q_{j,t-1}}_{\text{Queue last period}} - \underbrace{P_{j,t-1}}_{\substack{\text{Instructions} \\ \text{processed last period}}} + \underbrace{A_{j,t}}_{\substack{\text{Arrivals (in instructions)} \\ \text{this period}}} \tag{16}$$

Now, substitute $Q_{j,t} = L_j P_{j,t}$ into the preceding equation, and solve for $P_{j,t}$.

$$\begin{aligned} &\Rightarrow L_j P_{j,t} = L_j P_{j,t-1} - P_{j,t-1} + A_{j,t} \\ &\Rightarrow P_{j,t} = \left(1 - \frac{1}{L_j}\right) P_{j,t-1} + \frac{1}{L_j} A_{j,t}. \end{aligned} \tag{17}$$

Finally, substitute in for $A_{j,t}$. This gives us a recursive formula for $P_{j,t}$:

$$P_{j,t} = \left(1 - \frac{1}{L_j}\right) P_{j,t-1} + \frac{1}{L_j} \left(\sum_{k \in K} \sum_{l=1}^{N_{k,t-1}} \sum_{m=1}^{J^l_{j,k,t}} X^{klm}_{jt} + \sum_{l=1}^{R^e_{j,t}} \sum_{m=1}^{J^l_{j,R,t}} X^{Rlm}_{jt} \right). \tag{18}$$

Equation 3 -- Recursive Formula for the Work at Station *j*

Objective. We want to calculate the following statistics for all stations j in the network:

- $E[P_j]$, the expected work per period (in instructions) in steady state.
- $Var[P_j]$, the variance of the work per period in steady state.
- $Cov[P_i, P_j]$, the covariance of the work between two stations in steady state.

To calculate these values, we will use (18) to find recursive relationships for $E[P_{j,t}]$, $Var[P_{j,t}]$, and $Cov[P_i, P_j]$.. Then, we will iterate the recursions to find the steady-state values of $E[P_j]$, $Var[P_j]$ and $Cov[P_i, P_j]$. The expectations are calculated in section 5.2, and the variances are calculated in section 5.3. The

covariances are calculated in section 5.10. (The covariances derviation is a general proof that applies to networks running both the job-control rule and the instruction-control rule.)

5.2 Instruction-Rule Expectations

5.2.1 The Recursion Equation

Previously, we saw that $P_{j,t} = \left(1 - 1/L_j\right)P_{j,t-1} + \left(1/L_j\right)A_{j,t}$. Using the linearity of expectation, we find that

$$E[P_{j,t}] = \left(1 - \frac{1}{L_j}\right)E[P_{j,t-1}] + \frac{1}{L_j}E[A_{j,t}]. \tag{19}$$

We consider $E[A_{j,t}]$. Recall that

$$A_{j,t} = \sum_{k \in K}\sum_{l=1}^{N_{k,t-1}}\sum_{m=1}^{J^l_{j,k,t}} X^{klm}_{jt} + \sum_{l=1}^{R^e_{j,t}}\sum_{m=1}^{J^l_{j,R,t}} X^{Rlm}_{jt}. \tag{20}$$

We see that $A_{j,t}$ is made up of random summations of random variables. Then we can apply the following useful elementary probability formula.

Lemma 1. *Let Y be an independent random variables with a finite expectation and a finite variance, and let Z_i be a set of identically distributed random variables, independent of Y, that have a common mean E[Z].*

Then $E\left(\sum_{i=1}^{Y} Z_i\right) = E[Y]E[Z].$

Proof. See Appendix.

Iterating this expression, we find that:

$$E[A_{j,t}] = \sum_{k \in K}\left(E[N_k]E[J_{j,k}]E[X_j]\right) + E[R^e_j]E[J_{j,R}]E[X_j]. \tag{21}$$

There is one problem with this equation: we want to find the work at time t in terms of the work (in instructions) completed at time *t-1*. This equation relates work at time$t$ to the number of jobs completed at *t-1*. We need to get the expected number of jobs completed in terms of the expected work completed. However, this is not complicated. Again using the linearity of expectation, we see the expected number of jobs completed in a period, $E[N_{k,t-1}]$, is the expected production (in instructions) divided by the

expected number of instructions per job. Then we have $E[N_{k,t-1}] = E[P_{k,t-1}] / E[X_{k,t-1}]$. Substituting this expression into the equation for $E[A_{j,t}]$, we find:

$$E[A_{j,t}] = \sum_{k \in K} \underbrace{\left(\frac{E[J_{j,k}]E[X_j]}{E[X_k]} \right)}_{\substack{\text{This term comprises} \\ \text{constants that are given.} \\ \text{Replace it with } \Phi_{jk}.}} E[P_{k,t-1}] + \underbrace{E[R_j^e]E[J_{j,R}]E[X_j]}_{\substack{\text{This term comprises} \\ \text{constants that are given..} \\ \text{Replace it with } \mu_j.}}. \tag{22}$$

$$\Rightarrow E[A_{j,t}] = \sum_{k \in K} \Phi_{jk} E[P_{k,t-1}] + \mu_j.$$

We plug this expression into the equation for $E[P_{j,t}]$:

$$\boxed{E[P_{j,t}] = \left(1 - \frac{1}{L_j}\right) E[P_{j,t-1}] + \frac{1}{L_j} \left(\sum_{k \in K} \Phi_{jk} E[P_{k,t-1}] + \mu_j \right).} \tag{23}$$

Equation 4 -- Recursive Formula for the Expected Work at Station *j*

This equation applies to all stations in the model.

5.2.2 Calculating the Recursion in Matrix Form

We write the above equation for all stations simultaneously in matrix form.

- Let Φ be a square matrix whose (j,k) entry is Φ_{jk}.
- Let $\mathbf{D}$ be a diagonal matrix whose (j,j) entry is $1 / L_k$.
- Let $\mathbf{E[P_t]}$ be a vector whose jth entry is $E[P_{j,t}]$.
- Let μ be a vector whose jth entry is μ_j.

Then, we can write all of the $E[P_{j,t}]$ equations simultaneously as:

$$\begin{aligned} \mathbf{E[P_t]} &= \mathbf{(I - D)E[P_{t-1}] + D\Phi E[P_{t-1}] + D\mu}, \\ &= \underbrace{\mathbf{(I - D + D\Phi)}}_{\text{Call this } \mathbf{B}} \mathbf{E[P_{t-1}] + D\mu}, \\ &= \mathbf{BE[P_{t-1}] + D\mu}. \end{aligned} \tag{24}$$

The proof of this is term-by-term multiplication. Iterating this recursion yields:

$$
\begin{aligned}
\mathbf{E[P_t]} &= \mathbf{BE[P_{t-1}]} + \mathbf{D}\mu, \\
&= \mathbf{B(BE[P_{t-1}]} + \mathbf{D}\mu) + \mathbf{D}\mu, \\
&= \mathbf{B(B(B\ldots(BE[P_{t-n}]} + \mathbf{D}\mu) + \ldots + \mathbf{D}\mu) + \mathbf{D}\mu) + \mathbf{D}\mu,
\end{aligned} \tag{25}
$$

which, when iterated to infinity, becomes

$$\mathbf{E[P_t]} = \sum_{s=0}^{\infty} \mathbf{B^s D}\mu. \tag{26}$$

Using linear algebra theory, and assuming that Φ has a spectral radius less than one, we can evaluate this summation in closed form:

$$\boxed{\mathbf{E[P_t]} = (\mathbf{I} - \Phi)^{-1}\mu.} \tag{27}$$

Equation 5 -- Formula for the Expected Workload at All Stations

This gives us a formula for the steady-state expected workload at all the stations, as desired.□

5.3 Instruction-Rule Variances

5.3.1 The Top-Level of the Recursion Equation

Previously, we saw that $P_{j,t} = \left(1 - 1/L_j\right)P_{j,t-1} + \left(1/L_j\right)A_{j,t}$. If we take the variance of this sum, we find that

$$Var[P_{j,t}] = \left(1 - \frac{1}{L_j}\right)^2 Var[P_{j,t-1}] + \left(\frac{1}{L_j}\right)^2 Var[A_{j,t}] + 2\left(1 - \frac{1}{L_j}\right)\left(\frac{1}{L_j}\right)Cov[P_{j,t-1}, A_{j,t}]. \tag{28}$$

To use this equation, we will need to calculate $Var[A_{j,t}]$ and $Cov[P_{j,t-1}, A_{j,t}]$.

5.3.2 The Variance of Arrivals

We begin by looking at $Var[A_{j,t}]$. Recall that

$$A_{j,t} = \sum_{k \in K} \sum_{l=1}^{N_{k,t-1}} \sum_{m=1}^{J^l_{j,k,t}} X_{jt}^{klm} + \sum_{l=1}^{R^e_{j,t}} \sum_{m=1}^{J^l_{j,R,t}} X_{jt}^{Rlm}. \tag{29}$$

Since $A_{j,t}$ comprises random sums of random variables, we apply the following lemma for the variance of a random sum of random variables.

Lemma 2. *Let N be a random variable with finite expectation and a finite variance. Let X_i be a set of independent, identically distributed random variables, independent of N, that have a common mean E[X], and a common variance, Var[X].*

Define $Q = \sum_{i=1}^{N} X_i$. *Then* $Var[Q] = E[N]Var[X] + (E[X])^2 Var[N]$.

Proof. See appendix.

5.3.2.1 The Variance of External Arrivals

We begin the study of $Var[A_{j,t}]$ by looking at the second term of the expression for $A_{j,t}$, which represents arrivals from outside the system. (Call the arrivals from outside the system $A^e_{j,t}$.) We have:

$$A^e_{j,t} = \sum_{l=1}^{R^e_{j,t}} \sum_{m=1}^{J^l_{j,R,t}} X^{Rlm}_{jt}. \tag{30}$$

Then Var[$A^e_{j,t}$] is the variance of a random double summation of random variables. We calculate this value by applying the above "useful formula", iterated once. We find:

$$\begin{aligned} Var[A^e_{j,t}] = {} & E[R^e_j]E[J_{j,R}]Var[X_j] \\ & + E[R^e_j]Var[J_{j,R}](E[X_j])^2 \\ & + Var[R^e_j](E[J_{j,R}])^2(E[X_j])^2. \end{aligned} \tag{31}$$

We see that Var[$A^e_{j,t}$] comprises given constants. In future equations, we will simply summarize Var[$A^e_{j,t}$] to be σ^2_{ej}.

5.3.2.2 The Variance of Internal Arrivals

We now look at the first term of the expression for $A_{j,t}$, which represents arrivals from other stations in the network. (Call this term the *internal arrivals*, or $A^K_{j,t}$.) We have that

$$A^K_{j,t} = \sum_{k \in K} \sum_{l=1}^{N_{k,t-1}} \sum_{m=1}^{J^l_{j,k,t}} X^{klm}_{jt}. \tag{32}$$

Now, define a new variable. Let

$$A_{j,k,t} = \sum_{l=1}^{N_{k,t-1}} \sum_{m=1}^{J^l_{j,k,t}} X^{klm}_{jt}. \tag{33}$$

In words, $A_{j,k,t}$ is the work (in instructions) station k sends to station j at the start of period t. We then have that $A_{j,t}^{K} = \sum_{k \in K} A_{j,k,t}$. We take the variance of this sum using the "Bilinearity Property" of covariance.

Bilinearity Property (Form for a Sum of Random Variables). *Let X_i be a set of random variables with finite mean and variance. Then,*

$$\mathrm{var}\left(\sum_i b_i X_i\right) = \sum_i \sum_j b_i b_j \,\mathrm{cov}(X_i, X_j).$$

(This is a generally known result, and so is not proven here.)

Using the bilinearity property, we find that $\mathrm{Var}[A_{j,t}^{K}]$ is:

$$Var[A_{j,t}^{K}] = \sum_{k \in K} Var[A_{jkt}] + \sum_{k \in K} \sum_{\substack{l \in K \\ l \neq k}} Cov[A_{jkt}, A_{jlt}] \quad (34)$$

5.3.2.2.1 The Variance of the Workflow Between Two Stations

To evaluate this expression, we first characterize $\mathrm{Var}[A_{j,k,t}]$.

5.3.2.2.1.1 Initial Development

Since $A_{j,k,t} = \sum_{l=1}^{N_{k,t-1}} \sum_{m=1}^{J_{j,k,t}^{l}} X_{jt}^{klm}$. we reapply the formula for the variance of a random sum of random variables to find:

$$\begin{aligned} Var[A_{j,k,t}] = {} & E[N_{k,t-1}]E[J_{j,k}]Var[X_j] \\ & + E[N_{k,t-1}]Var[J_{j,k}](E[X_j])^2 \\ & + Var[N_{k,t-1}](E[J_{j,k}])^2(E[X_j])^2. \end{aligned} \quad (35)$$

As with the expectation calculations, we must write $\mathrm{Var}[N_{k,t-1}]$ as a function of $\mathrm{Var}[P_{k,t-1}]$ and other given constants. To do so, we first look at the variance of the number of instruction in station k's work queue, Q_k. The number of instructions queued at station k is itself a random sum of random variables. In particular, $Q_{k,t-1} = \sum_{l=1}^{q_{k,t-1}} X_{k,t-1}^{l}$, where $q_{k,t-1}$ is the number of jobs queued at station k at time $t-1$. We therefore have that

$$Var[Q_{k,t-1}] = E[q_{k,t-1}]Var[X_k] + (E[X_k])^2 Var[q_{k,t-1}]. \quad (36)$$

Next, recall that $P_{k,t-1} = Q_{k,t-1} / L_k$. This implies that $L_k^2 Var[P_{k,t-1}] = Var[Q_{k,t-1}]$, and $E[Q_{k,t-1}] = L_k E[P_{k,t-1}]$. If we substitute these expressions into the equation for Var[$Q_{k,t-1}$], and solve for Var[$q_{k,t-1}$], we find:

$$Var[q_{k,t-1}] = \frac{L_k^2 Var[P_{k,t-1}]}{(E[X_k])^2} - \frac{L_k E[N_{k,t-1}] Var[X_k]}{(E[X_k])^2}. \tag{37}$$

Next, we attempt to find $Var[N_{k,t-1}]$ in terms of $Var[Q_{k,t-1}]$. This involves answering the following question: given that there are $q_{k,t-1}$ jobs in the queue, how many jobs will be in the $Q_{k,t-1} / L_k$ instructions processed in period *t-1*?

The answer depends on how we control the work at station k and what the distribution of X_k is. We compute $Var[N_{k,t-1}]$ exactly for two "simple" cases that provide upper and lower bounds for $Var[N_{k,t-1}]$ for most networks.

5.3.2.2.1.2 A Lower Bound – "Double Control"

Case 1 (Lower Bound). We control the order in which station k performs instructions so that the number of jobs processed each period is approximately equal to $q_{k,t-1} / L_k$. (This is feasible if we expect lots of jobs in the queue each period, and if X_k is a reasonably "nice" distribution.) But then, $N_{k,t-1} \approx q_{k,t-1} / L_k \Rightarrow Var[N_{k,t-1}] \geq Var[q_{k,t-1}] / L_k^2$. Substituting this expression into the equation for $Var[q_{k,t-1}]$, we find that:

$$Var[N_{k,t-1}] \geq \frac{Var[P_{k,t-1}]}{(E[X_k])^2} - \frac{E[N_{k,t-1}] Var[X_k]}{L_k (E[X_k])^2}. \tag{38}$$

Now, recall from the previous section that E[$N_{k,t-1}$] = E[$P_{k,t-1}$] / E[X_k]. Substituting this into the previous equation, we find that, for Case 1,

$$Var[N_{k,t-1}] \geq \frac{Var[P_{k,t-1}]}{(E[X_k])^2} - \frac{E[P_{k,t-1}] Var[X_k]}{L_k (E[X_k])^3}. \tag{39}$$

5.3.2.2.1.3 An Upper Bound – "Simple Control"

Case 2 (Upper Bound). In this case, jobs are processed on a first-come, first-served basis, and that the X_k's are exponential. Then we can apply the following lemma:

Lemma 3. *Let X_i be a set of independent, identically distributed random variables that come from an exponential distribution with parameter λ. Consider a queue containing N jobs, where N is some positive integer, and where each job has length X_i. Assume the total length of the queue is Q. Then the endpoints of the first $N-1$ jobs in the queue will be uniformly distributed over $(0,Q)$.*

Proof. See appendix.

Then, each of the first q_k - *1* jobs in the queue has a *1* / L_k chance of being processed in period *t-1*. Equivalently, the chance that each of these jobs is processed in period *t-1* has a Bernoulli distribution with parameter *1* / L_k.

Then, we can derive an upper bound for the total number of jobs processed in period *t-1* from a random sum of random variables. We have that $N_{k,t-1} = \sum_{i=1}^{q_{k,t-1}-1} p_i$, where p_i is a Bernoulli random variable with parameter *1* / L_k. This summation cannot be used directly, because it can produce negative numbers when the work queue is empty. In particular, the summation will create negative numbers when $E[q_k] \leq 1$. We can, however, use an upper bound for $N_{k,t-1}$ by assuming that the last job in the queue, the q_kth job, also has a *1* / L_k chance of being processed. This assumption gives us an upper bound of $N_{k,t-1} < \sum_{i=1}^{q_{k,t-1}} p_i$. Since this upper bound consists of adding a positive random variable to the previous summation of random variables for $N_{k,t-1}$, we have that $Var\left(N_{k,t-1}\right) < Var\left(\sum_{i=1}^{q_{k,t-1}} p_i\right)$. Reapplying the formula for the random sum of random variables, we find that

$$Var[N_{k,t-1}] < E[q_{k,t-1}]Var[p] + (E[p_i])^2 Var[q_{k,t-1}]. \tag{40}$$

The expectation of a Bernoulli random variable with parameter *1* / L_k is just *1* / L_k; the variance is $(1/L_k - 1/L_k^2)$. $E[q_{k,t-1}]$ is the expected number of instructions in the queue divided by the number of instructions per job, $E[Q_{k,t-1}]$ / $E[X_{k,t-1}]$. Noting that $E[Q_{k,t-1}] = L_k E[P_{k,t-1}]$, we see that $E[q_{k,t-1}]$ = $L_k E[P_{k,t-1}]$ / $E[X_{k,t-1}]$. $Var[q_{k,t-1}]$ was calculated above. Substituting these expressions into the above equation, we find that:

$$Var[N_{k,t-1}] < \left(1 - \frac{1}{L_k}\right)\frac{E[P_{k,t-1}]}{E[X_k]} + \frac{Var[P_{k,t-1}]}{E[X_k]^2} - \frac{E[P_{k,t-1}]Var[X_k]}{L_k(E[X_k])^3}. \tag{41}$$

Note that this is the same as the variance in Case 1, except for the addition of the first term, which is on the order of $E[P_{k,t-1}] / E[X_k] = E[N_{k,t-1}]$.

As stated, the variance found in Case 1 and Case 2 are lower and upper bounds for $Var[N_{k,t-1}]$, respectively. The lower bound always holds; the upper bound will hold provided that X_k is a "nice" distribution. The following lemma provides a mathematical description of "nice".

Lemma 4. *Suppose that X_k, a nonnegative random variable for the number of instructions per job at station k, satisfies the following condition: for all values of a constant $t_0 \geq 0$,*

$$E[X_k - t_0 \mid X_k > t_0] \leq E[X_k].$$

Then the following result holds:

$$Var[N_k] \leq E[q_k]\left(\frac{1}{L_k} - \frac{1}{L_k^2}\right) + \frac{1}{L_k^2} Var[q_k],$$

where N_k is the number of jobs contained in the first $1 / L_k$ fraction of instructions in the queue at station k, and q_k is the total number of jobs in the queue.

Proof. See appendix.

In words, the condition $E[X_k - t_0 \mid X_k > t_0] \leq E[X_k]$ means the following. Suppose we arrive at station k at a random time, and wait for the current job in the queue to finish. Further, suppose we know that station k has been working on the current job for at least t_0 time units. The condition says that the expected time remaining until the job is finished is less than the unconditional expected time to process a job at station k.

5.3.2.2.1.4 Completing the Derivation

For both cases, $Var[N_{k,t-1}]$ has the following form:

$$Var[N_{k,t-1}] = \frac{Var[P_{k,t-1}]}{(E[X_k])^2} + u_k^0, \tag{42}$$

where u_k^0 is a set of constant terms equal to $-\left(E[P_{k,t-1}]Var[X_k]\right) / L_k (E[X_k])^3$ if station k uses the double-control rule, and

$$u_k^0 = \left(1 - \frac{1}{L_k}\right)\frac{E[P_{k,t-1}]}{E[X_k]} - \frac{E[P_{k,t-1}]Var[X_k]}{L_k(E[X_k])^3}, \tag{43}$$

if station k uses the simple instruction control rule. Note that the "constant terms" contain $E[P_{k,t-1}]$. We can treat this term as a constant if we solve for the expected workloads for the stations first, which yields $E[P_k]$, and note that $E[P_{k,t-1}]$ equals $E[P_k]$ in steady state.

Now, we substitute the above expression for $Var[N_{k,t-1}]$ back into our expression for $Var[A_{jkt}]$. This yields:

$$\begin{aligned} Var[A_{j,k,t}] &= E[N_{k,t-1}]E[J_{j,k}]Var[X_j] \\ &\quad + E[N_{k,t-1}]Var[J_{j,k}](E[X_j])^2 \\ &\quad + \left(\frac{Var(P_{k,t-1})}{(E[X_k])^2} + u_k^0\right)(E[J_{j,k}])^2(E[X_j])^2. \end{aligned} \tag{44}$$

We simplify this expression as follows. The first two terms comprise given constants. We add them together and call the sum $\sigma_{j,k,t}^2$. To simplify the third term, we recall that in the previous section we let $\Phi_{jk} = E[J_{j,k}]E[X_j]/E[X_k]$. Making these substitutions, we find that:

$$Var[A_{j,k,t}] = \Phi_{jk}^2\left(Var[P_{k,t-1}] + u_k\right) + \sigma_{j,k,t}^2, \tag{45}$$

$$\text{where } u_k = (E[X_k]^2)u_k^0.$$

5.3.2.2.2 The Variance of a Sum of Workflows

Now that we have $Var[A_{jkt}]$, the next step is to compute $Var\left[\sum_{k\in K} A_{j,k,t}\right]$, which is the variance of the internal arrivals to station j at time period $t-1$.

5.3.2.2.2.1 Initial Derivation

We have:

$$Var\left[\sum_{k\in K} A_{j,k,t}\right] = \sum_{k\in K} Var[A_{j,k,t}] + \sum_{k\in K}\sum_{\substack{l\in K \\ l\neq k}} Cov[A_{j,k,t}, A_{j,l,t}] \tag{46}$$

We will use the following lemma to compute this sum.

Lemma 5. *Let N_k be a set K of random variables, each with a finite expectation and a finite variance. Let Y_{ik} be a set of independent, identically distributed random variables, independent from every N_k, that have a common mean $E[Y_k]$, and a common variance, $Var[Y_k]$.*

Define $T = \sum_{k\in K}\sum_{i=1}^{N_k} Y_{ik}$. *Then:*

$$Var[T] = \sum_{k\in K}\left(E[N_k]Var[Y_k] + (E[Y_k])^2 Var[N_k]\right) + \sum_{k\in K}\sum_{\substack{l\in K\\ l\neq k}} E[Y_k]E[Y_l]Cov[N_k, N_l]$$

$$= \sum_{k\in K}\left(Var\left[\sum_{i=1}^{N_k} Y_{ik}\right]\right) + \sum_{k\in K}\sum_{\substack{l\in K\\ l\neq k}} E[Y_k]E[Y_l]Cov[N_k, N_l].$$

We can apply Lemma 5 directly to $Var\left[\sum_{k\in K} A_{j,k,t}\right]$ by letting the N_k's in this derivation equal the N_k's in the lemma, and by letting the lemma's $Y_{ik} = \sum_{l=1}^{J_{jk}^i} X_j^{il}$. Doing so yields:

$$(47)\quad Var\left[\sum_{k\in K} A_{jkt}\right] = \sum_{k\in K}\left(Var[A_{jkt}]\right) + \sum_{k\in K}\sum_{\substack{l\in K\\ l\neq k}}\left(E[J_{jk}]E[X_j]\right)\left(E[J_{jl}]E[X_j]\right)Cov[N_{k,t-1}, N_{l,t-1}].$$

As before, we need to find $Cov[N_{k,t-1}, N_{l,t-1}]$ in terms of $Cov[P_{k,t-1}, P_{l,t-1}]$. We begin by calculating $Cov[q_{k,t-1}, q_{l,t-1}]$. Recall that $Q_{k,t-1} = \sum_{l=1}^{q_{k,t-1}} X_{k,t-1}^l$. Applying Lemma 5, we find that:

$$(48)\quad Var\left[\sum_{k\in K} Q_{k,t-1}\right] = \sum_{k\in K}\left(Var[Q_{k,t-1}]\right) + \sum_{k\in K}\sum_{\substack{l\in K\\ l\neq k}}(E[X_k])(E[X_l])Cov[q_{k,t-1}, q_{l,t-1}].$$

But, by the bilinearity property of covariance, we also have that:

$$(49)\quad Var\left[\sum_{k\in K} Q_{k,t-1}\right] = \sum_{k\in K} Var[Q_{k,t-1}] + \sum_{k\in K}\sum_{\substack{l\in K\\ l\neq k}} Cov[Q_{k,t-1}, Q_{l,t-1}].$$

If we equate the above two expressions for $Var\left[\sum_{k\in K} Q_{k,t-1}\right]$, we find:

$$(50)\quad \sum_{k\in K}\sum_{\substack{l\in K\\ l\neq k}} Cov[Q_{k,t-1}, Q_{l,t-1}] = \sum_{k\in K}\sum_{\substack{l\in K\\ l\neq k}} E[X_k]E[X_l]Cov[q_{k,t-1}, q_{l,t-1}].$$

Since this equation must hold for all possible sets K, we must have that:

(51) $$Cov[Q_{k,t-1}, Q_{l,t-1}] = E[X_k]E[X_l]Cov[q_{k,t-1}, q_{l,t-1}], \forall k, \forall l.$$

Now, recall that $Q_{k,t-1} = L_k P_{k,t-1}$. Using the bilinearity property of covariance, this implies that $Cov[Q_{k,t-1}, Q_{l,t-1}] = L_k L_l Cov[P_{k,t-1}, P_{l,t-1}]$. Substituting, and solving for Cov[$q_{k,t-1}$, $q_{l,t-1}$], we find:

(52) $$Cov[q_{k,t-1}, q_{l,t-1}] = \frac{L_k L_l Cov[P_{k,t-1}, P_{l,t-1}]}{E[X_k]E[X_l]}.$$

The next step is to find Cov[$N_{k,t-1}$, $N_{l,t-1}$] in terms of Cov[$q_{k,t-1}$, $q_{l,t-1}$]. We use the same two cases as before, when we found Var[$N_{k,t-1}$].

5.3.2.2.2.2 Case 1 -- Lower Bound (Double Control)

In the lower-bound case, $N_{k,t-1}$ approximately equals $q_{k,t-1} / L_k$. Then, the linearity property of covariance requires that $Cov[q_{k,t-1}, q_{l,t-1}] = L_k L_l Cov[N_{k,t-1}, N_{l,t-1}]$. Substituting this expression into the equation for Cov[$q_{k,t-1}$, $q_{l,t-1}$], we find:

(53) $$Cov[N_{k,t-1}, N_{l,t-1}] = \frac{Cov[P_{k,t-1}, P_{l,t-1}]}{E[X_k]E[X_l]}.$$

5.3.2.2.2.3 Case 2 – Upper Bound (Simple Control)

In this case, we process jobs and instructions on a first-come, first served basis, and assume that all of the X_k's have an exponential distribution. Recall this implies that $N_{k,t-1} = \sum_{i=1}^{q_{k,t-1}} p_i$, where p_i is a Bernoulli random variable with parameter $1 / L_k$. But then we can apply Lemma 5 directly to $Var\left(\sum_{k \in K} N_{k,t-1}\right)$, which yields:

(54) $$Var\left[\sum_{k \in K} N_{k,t-1}\right] = \sum_{k \in K}\left(Var[N_{k,t-1}]\right) + \sum_{k \in K}\sum_{\substack{l \in K \\ l \neq k}} \frac{Cov[q_{k,t-1}, q_{l,t-1}]}{L_k L_l}.$$

(Here, we have used the fact that $E[p_k] = 1 / L_k$.) Of course, by the bilinearity property of covariance, we also have:

(55) $$Var\left[\sum_{k \in K} N_{k,t-1}\right] = \sum_{k \in K} Var[N_{k,t-1}] + \sum_{k \in K}\sum_{\substack{l \in K \\ l \neq k}} Cov[N_{k,t-1}, N_{l,t-1}].$$

Equating these two expressions for $Var\left(\sum_{k \in K} N_{k,t-1}\right)$, we find:

$$\sum_{k\in K}\sum_{\substack{l\in K\\ l\neq k}} Cov[N_{k,t-1}, N_{l,t-1}] = \sum_{k\in K}\sum_{\substack{l\in K\\ l\neq k}} \frac{Cov[q_{k,t-1}, q_{l,t-1}]}{L_k L_l}. \tag{56}$$

Since the above expression must hold for all possible sets K, we see that:

$$Cov[N_{k,t-1}, N_{l,t-1}] = \frac{Cov[q_{k,t-1}, q_{l,t-1}]}{L_k L_l}, \forall k, \forall l. \tag{57}$$

Substituting this expression into the formula for Cov[$q_{k,t-1}$, $q_{l,t-1}$], we find:

$$Cov[N_{k,t-1}, N_{l,t-1}] = \frac{Cov[P_{k,t-1}, P_{l,t-1}]}{E[X_k]E[X_l]}. \tag{58}$$

Note that this is the same value for Cov[$N_{k,t-1}$, $N_{l,t-1}$] we found in Case 1.

5.3.2.2.2.4 Completing the Derivation of the Variance of Internal Arrivals

Recall that an expression for $Var\left[\sum_{k\in K} A_{j,k,t}\right]$ is:

$$Var\left[\sum_{k\in K} A_{jkt}\right] = \sum_{k\in K}\left(Var[A_{jkt}]\right) + \sum_{k\in K}\sum_{\substack{l\in K\\ l\neq k}}\left(E[J_{jk}]E[X_j]\right)\left(E[J_{jl}]E[X_j]\right)Cov[N_{k,t-1}, N_{l,t-1}]. \tag{59}$$

If we substitute in the expression for Cov[$N_{k,t-1}$, $N_{l,t-1}$] we found for both the upper and lower bound cases, and recall that $\Phi_{jk} = E[J_{jk}]E[X_j]/E[X_k]$, we find that:

$$Var\left[\sum_{k\in K} A_{jkt}\right] = \sum_{k\in K}\left(Var[A_{jkt}]\right) + \sum_{k\in K}\sum_{\substack{l\in K\\ l\neq k}}\Phi_{jk}\Phi_{jl}Cov[P_{k,t-1}, P_{l,t-1}]. \tag{60}$$

This gives us a formula for the internal arrivals to station j at time period $t-1$, as desired.

5.3.2.3 A Formula for the Variance of Arrivals

Now, we return to our original formula for Var[$A_{j,t}$], which was:

$$\begin{aligned} Var[A_{j,t}] &= \underbrace{Var\left[\sum_{k\in K}\sum_{l=1}^{N_{k,t-1}}\sum_{m=1}^{J^l_{j,k,t}} X^{klm}_{jt}\right]}_{\substack{\text{Arrivals from other stations}\\ \text{(Internal Arrivals)}}} + \underbrace{Var\left[\sum_{l=1}^{R^e_{j,t}}\sum_{m=1}^{J^l_{j,R,t}} X^{Rlm}_{jt}\right]}_{\text{External arrivals}} \\ &= Var\left[\sum_{k\in K} A_{j,k,t}\right] + \sigma^2_{ej}. \end{aligned} \tag{61}$$

Since we derived formulas for the variances of internal arrivals and external arrivals above, we now know $Var[A_{j,t}]$.

5.3.3 The Covariance of Workload and Arrivals

To calculate $Var[P_{j,t}]$, we need to calculate one final term: $Cov[P_{j,t-1}, A_{j,t}]$. To do this, consider $Cov[A_{j,j,t}, A_{j,t}]$. This is the covariance of the arrivals station j sends to itself with the arrivals from all of the other stations. By looking at the covariance terms in $Var\left[\sum_{k\in K} A_{j,k,t}\right]$, we see that

$$Cov[A_{j,j,t}, A_{j,t}] = \sum_{l\in K} \Phi_{jj}\Phi_{jl} Cov[P_{j,t-1}, P_{l,t-1}]. \tag{62}$$

But then, by the Bilinearity Property of Covariance,

$$Cov[P_{j,t-1}, A_{j,t}] = \sum_{l\in K} \Phi_{jl} Cov[P_{j,t-1}, P_{l,t-1}]. \tag{63}$$

5.3.4 Completing the Recursion Equation for Workload Variances

Now, we return to our original formula for $Var[P_{j,t}]$, which was:

$$Var[P_{j,t}] = \left(1-\frac{1}{L_j}\right)^2 Var[P_{j,t-1}] + \left(\frac{1}{L_j}\right)^2 Var[A_{j,t}] + 2\left(1-\frac{1}{L_j}\right)\left(\frac{1}{L_j}\right) Cov[P_{j,t-1}, A_{j,t}]. \tag{64}$$

If we substitute in the expressions we found for $Var[A_{j,t}]$ and $Cov[P_{j,t-1}, A_{j,t}]$, we find that:

$$\begin{aligned} Var[P_{j,t}] = {} & \left(1-\frac{1}{L_j}\right)^2 Var[P_{j,t-1}] + \left(\frac{1}{L_j}\right)^2 \sum_{k\in K}\left[\Phi_{jk}^2\left(Var[P_{k,t-1}] + u_k\right) + \sigma_{j,k,t}^2\right] \\ & + \left(\frac{1}{L_j}\right)^2 \sigma_{ej}^2 + \left(\frac{1}{L_j}\right)^2 \sum_{k\in K}\sum_{\substack{l\in K \\ l\neq k}}\left[\Phi_{jk}\Phi_{jl} Cov[P_{k,t-1}, P_{l,t-1}]\right] \\ & + 2\left(1-\frac{1}{L_j}\right)\left(\frac{1}{L_j}\right)\sum_{k\in K}\left[\Phi_{jk} Cov[P_{j,t-1}, P_{k,t-1}]\right] \end{aligned} \tag{65}$$

5.3.5 Calculating the Recursion in Matrix Form

As with the expectation calculations, we write all of the $Var[P_{j,t}]$ equations simultaneously in matrix form. Let:

- $\mathbf{S}_t$ be a square matrix with the $Var[P_{j,t}]$'s on the diagonal, and $Cov[P_{j,t}, P_{k,t}]$'s on the off-diagonal elements.
- $\mathbf{U}$ be a diagonal matrix with the u_k's on the diagonal.
- Σ be a diagonal matrix with diagonal elements $\Sigma_{jj} = \sigma_{ej}^2 + \sum_{k \in K} \sigma_{j,k,t}^2$.
- Φ be a square matrix whose *(j,k)* element is Φ_{jk}.
- $\mathbf{D}$ be a diagonal matrix with the $1 / L_k$'s on the diagonal.
- $\mathbf{B}$ be a square matrix that equals $\mathbf{(I - D - D\Phi)}$, where $\mathbf{I}$ is the identity matrix.

Then, a matrix equation for $\mathbf{S}_t$ in terms of $\mathbf{S}_{t-1}$ is:

$$\mathbf{S}_t = \mathbf{BS}_{t-1}\mathbf{B}'+(\mathbf{D\Phi})\mathbf{U}(\mathbf{D\Phi})'+\mathbf{D\Sigma D}. \tag{66}$$

This equation may be checked by term-by-term multiplication. If we recurse this equation, we find that:

$$\begin{aligned} \mathbf{S}_t &= \mathbf{BS}_{t-1}\mathbf{B}'+(\mathbf{D\Phi})\mathbf{U}(\mathbf{D\Phi})'+\mathbf{D\Sigma D} \\ &= \mathbf{B}(\mathbf{BS}_{t-2}\mathbf{B}'+(\mathbf{D\Phi})\mathbf{U}(\mathbf{D\Phi})'+\mathbf{D\Sigma D})\mathbf{B}'+(\mathbf{D\Phi})\mathbf{U}(\mathbf{D\Phi})'+\mathbf{D\Sigma D} \\ &= \mathbf{B}(\mathbf{B}(\mathbf{B}\ldots(\mathbf{BS}_{t-n}\mathbf{B}'+(\mathbf{D\Phi})\mathbf{U}(\mathbf{D\Phi})'+\mathbf{D\Sigma D})\ldots \\ &\quad \mathbf{B}'+(\mathbf{D\Phi})\mathbf{U}(\mathbf{D\Phi})'+\mathbf{D\Sigma D})\mathbf{B}'+(\mathbf{D\Phi})\mathbf{U}(\mathbf{D\Phi})'+\mathbf{D\Sigma D})\mathbf{B}'+(\mathbf{D\Phi})\mathbf{U}(\mathbf{D\Phi})'\mathbf{D\Sigma D}, \end{aligned} \tag{67}$$

which, assuming an infinite history of the system, becomes:

$$\mathbf{S} = \sum_{s=0}^{\infty} \mathbf{B}^s((\mathbf{D\Phi})\mathbf{U}(\mathbf{D\Phi})'+\mathbf{D\Sigma D})\mathbf{B}'^s. \tag{68}$$

Equation 6 -- Variances and Covariances of All Station Workloads

We have found a formula for the steady-state variances and covariances of all of the station workloads, as desired. □

(*Note*: The derivation of (68) assumes that the above matrix equation for $\mathbf{S}_t$ calculates the covariance between pairs of stations correctly. Otherwise, we would not be able to recurse on the equation for $\mathbf{S}_t$ to find the steady-state variances and covariances. Section 5.10 proves that the equation for $\mathbf{S}_t$ calculates the covariance terms correctly.)

5.4 Job-Rule Mechanics

The following three sections derive the LRS-MR model equations for networks in which all stations use the jobs shop rule. In this section, we mathematically describe the behavior of the workstations under the job-control rule, as declared in Section 4.2.

Measurement of Work. Under the job-shop rule, the model equations track the work at all of the stations in terms of jobs, not instructions. Consequently, the equations under the job-shop rule make no reference to instructions. In calculating the results of the model, we will first compute the expectation and variances of the workloads in jobs per period. Then, we will apply our formulas for the expectation and variance of a random sum of random variables to calculate the expectation and variance of the workloads in instructions.

Work Arrivals. Let station k be upstream of station j. Station k sends $N_{k,t-1}$ requests to station j at the start of period t, which is then converted into a random number of jobs. Then the work, in jobs, that station k sends to station j at the start of period t is:

$$a_{j,k,t} = \sum_{l=1}^{N_{k,t-1}} J_{jkt}^{l} \tag{69}$$

Now, let station j receive input from a subset of stations, K. (Note that K can include j, since a station can send requests to itself.) The total arrival (in jobs) to k at the start of period t, as a function of all the requests received from the stations in K plus the external requests, is:

$$a_{j,t} = \sum_{k \in K} \sum_{l=1}^{N_{k,t-1}} J_{j,k,t}^{l} + \sum_{l=1}^{R_{j,t}^{e}} J_{j,R,t}^{l} \tag{70}$$

Control rule. Each period, station j processes $1/L_j$ of the work (in jobs) in its queue. Mathematically, this is $N_{j,t} = q_{j,t} / L_j$. We relate the queue length at period t to the queue length at t-1. We have:

$$q_{j,t} = \underbrace{q_{j,t-1}}_{\substack{\text{Queue in jobs} \\ \text{last period}}} - \underbrace{N_{j,t-1}}_{\substack{\text{Jobs processed} \\ \text{last period}}} + \underbrace{a_{j,t.}}_{\substack{\text{Arrivals (in jobs)} \\ \text{this period}}} . \tag{71}$$

Now, substitute $q_{j,t} = L_j N_{j,t}$ into the preceding equation, and solve for $N_{j,t}$.

$$\Rightarrow L_j N_{j,t} = L_j N_{j,t-1} - N_{j,t-1} + a_{j,t}$$

$$\Rightarrow N_{j,t} = \left(1 - \frac{1}{L_j}\right) N_{j,t-1} + \frac{1}{L_j} a_{j,t}. \tag{72}$$

Finally, substitute in for $a_{j,t}$. This gives us a recursive formula for $N_{j,t}$:

$$N_{j,t} = \left(1 - \frac{1}{L_j}\right) N_{j,t-1} + \frac{1}{L_j}\left(\sum_{k \in K} \sum_{l=1}^{N_{k,t-1}} J_{j,k,t}^{l} + \sum_{l=1}^{R_{j,t}^{e}} J_{j,R,t}^{l}\right). \tag{73}$$

Equation 7-- Recursive Formula for the Jobs Processed at Station *j*

Objective. Again, we want to calculate the following statistics for all stations j in the network:

- $E[P_j]$, the expected work per period (in instructions) in steady state.
- $Var[P_j]$, the variance of the work per period in steady state.

To calculate these values, we will use (73) to find recursive relationships for $E[N_{j,t}]$ and $Var[N_{j,t}]$. Then, we will iterate the recursions to find the steady-state values of $E[N_j]$ and $Var[N_j]$. Next, we will use our formulas for the expectation and variances of random sums to calculate $E[P_j]$ and $Var[P_j]$.

5.5 Job-Rule Expectations

We know that $N_{j,t} = (1 - 1/L_j) N_{j,t-1} + (1/L_j) a_{j,t}$ Then, by the linearity of expectation, we must have:

$$E[N_{j,t}] = \left(1 - \frac{1}{L_j}\right) E[N_{j,t-1}] + \frac{1}{L_j} E[a_{j,t}]. \tag{74}$$

We first consider $E[a_{j,t}]$. Since

$$a_{j,t} = \sum_{k \in K} \sum_{l=1}^{N_{k,t-1}} J_{j,k,t}^{l} + \sum_{l=1}^{R_{j,t}^{e}} J_{j,R,t}^{l}, \tag{75}$$

the formula for the expectation of a random sum of random variables implies that:

$$E[a_{j,t}] = \sum_{k \in K} \underbrace{\left(E[J_{j,k}]\right)}_{\text{Let this be } \Phi_{jk}.} E[N_{k,t-1}] + \underbrace{E[R_j^e]E[J_{j,R}]}_{\text{Let this be } \mu_j.}$$

(76)

$$\Rightarrow E[a_{j,t}] = \sum_{k \in K} \Phi_{jk} E[N_{k,t-1}] + \mu_j.$$

We substitute this into the expression for $E[N_{j,t}]$, which yields:

(77)

$$\boxed{E[N_{j,t}] = \left(1 - \frac{1}{L_j}\right) E[N_{j,t-1}] + \frac{1}{L_j}\left(\sum_{k \in K} \Phi_{jk} E[N_{k,t-1}] + \mu_j\right).}$$

Equation 8 -- Recursive Formula for the Expected Jobs at Station *j*

This equation applies to all stations in the model. We write these equations for all stations simultaneously in matrix form.

- Let Φ be a square matrix whose (j,k) entry is Φ_{jk}.
- Let **D** be a diagonal matrix whose (j,j) entry is $1 / L_k$.
- Let $\mathbf{E[N_t]}$ be a vector whose jth entry is $E[N_{j,t}]$.
- Let μ be a vector whose jth entry is μ_j.

Then, we can write all of the $E[N_{j,t}]$ equations simultaneously as:

$$\begin{aligned} \mathbf{E[N_t]} &= \mathbf{(I - D)E[N_{t-1}] + D\Phi E[N_{t-1}] + D\mu}, \\ &= \underbrace{\mathbf{(I - D + D\Phi)}}_{\text{Call this B}} \mathbf{E[N_{t-1}] + D\mu}, \\ &= \mathbf{BE[N_{t-1}] + D\mu}. \end{aligned}$$

(78)

The proof of this is term-by-term multiplication. Infinitely iterating this recursion yields:

$$\mathbf{E[N_t]} = \sum_{s=0}^{\infty} \mathbf{B^s D}\mu.$$

(79)

Using linear algebra theory, we can evaluate this summation in closed form:

$$\boxed{\mathbf{E[N_t]} = \mathbf{(I - \Phi)^{-1}}\mu.}$$

(80)

Equation 9 -- Formula for the Expected Jobs at All Stations

Now, to calculate $E[P_j]$, the expected demand in instructions at station j, we recall that $P_{jt} = \sum_{l=1}^{J_{j,t}} X_{jt}^{l}$. But then, the formula for the expectation of a random sum implies that:

$$E[P_j] = \mathbf{E}[\mathbf{N}_t]_j E[X_j], \tag{81}$$

where $\mathbf{E}[\mathbf{N}_t]_j$ is the jth entry of $\mathbf{E}[\mathbf{N}_t]$, for all stations j. This gives us a formula for the steady-state expected workload at all the stations, as desired.□

5.6 Job-Rule Variances

5.6.1 Basic Formula

If we take the variance of $N_{j,t} = (1-1/L_j)N_{j,t-1} + (1/L_j)a_{j,t}$, we get:

$$Var[N_{j,t}] = \left(1-\frac{1}{L_j}\right)^2 Var[N_{j,t-1}] + \left(\frac{1}{L_j}\right)^2 Var[a_{j,t}] + 2\left(1-\frac{1}{L_j}\right)\left(\frac{1}{L_j}\right) Cov[N_{j,t-1}, a_{j,t}] \tag{82}$$

5.6.2 Variance of Arrivals

As with expectations, we consider Var[$a_{j,t}$] first. We have:

$$a_{j,t} = \sum_{k\in K}\sum_{l=1}^{N_{k,t-1}} J_{j,k,t}^{l} + \sum_{l=1}^{R_{j,t}^{e}} J_{j,R,t}^{l}. \tag{83}$$

Since $a_{j,t}$ comprises random sums of random variables, we can apply the formula for the variance of a sum to it.

First, we apply this formula directly to the second term of the expression for $a_{j,t}$. Here, the second term represents $a^e_{j,t}$, the arrivals from outside the network. Applying the formula, we find:

$$Var[a_{j,t}^{e}] = E[R_j^e]Var[J_{j,R}] + (E[J_{j,R}])^2 Var[R_j^e]. \tag{84}$$

The first term represents $a^K_{j,t}$, the arrivals from other stations. Finding its variance is a bit more complicated. We see that $a^K_{j,t}$ is a double sum of random variables, and that the variables of the outermost sum (the N_k's) are not independent. However, recall that Lemma 5 (in the appendix) gives a formula for the variance of this type of double summation. Applying that lemma yields:

$$Var[a_{j,t}^{K}] = \sum_{k\in K}\left(E[N_k]Var[J_{j,k}] + (E[J_{j,k}])^2 Var[N_k]\right) + \sum_{k\in K}\sum_{\substack{l\in K \\ l\neq k}} E[J_{j,k}]E[J_{j,l}]Cov[N_k, N_l] \tag{85}$$

5.6.3 Covariance of Arrivals and Production

The external arrivals are independent from the arrivals from other stations, so to find Var[$a_{j,t}$] we simply add together Var[$a^e_{j,t}$] and Var[$a^K_{j,t}$]. We can rewrite this sum as follows:

$$Var[a_{j,t}] = \sum_{k\in K}\Phi^2_{j,k}Var[N_k] + \sum_{k\in K}\sum_{\substack{l\in K\\ l\neq k}}\Phi_{jk}\Phi_{jl}Cov[N_k, N_l] + \Sigma_{jj}, \text{ where}$$

(86)
$$\Phi_{jk} = E[J_{j,k}], \text{ and}$$
$$\Sigma_{jj} = \sum_{k\in K}\Big(E[N_k]Var[J_{j,k}]\Big) + E[R^e_j]Var[J_{j,R}] + (E[J_{j,R}])^2 Var[R^e_j].$$

To calculate Var[$N_{j,t}$], we need to find one additional term: Cov[$N_{j,t-1}$, $a_{j,t}$]. This is the covariance of the job arrivals station j sends to itself with the arrivals from all of the other stations. To calculate this term, consider $Cov[a_{j,j,t}, a_{j,t}]$. Here, $a_{j,j,t}$ is the work in jobs station j sends to itself at the start of period t, and so is a function of $N_{j,t-1}$. By looking at the covariance terms in Var[$a^K_{j,t}$], we see that:

(87)
$$Cov[a_{j,j,t-1}, a_{j,t}] = \sum_{l\in K}\Phi_{jj}\Phi_{jl}Cov[N_{j,t-1}, N_{l,t-1}].$$

But then, by the bilinearity property of covariance, we must have:

(88)
$$Cov[N_{j,t-1}, a_{j,t}] = \sum_{l\in K}\Phi_{jl}Cov[N_{j,t-1}, N_{l,t-1}].$$

5.6.4 The Recursive Equation

We now return to our original formula for Var[$N_{j,t-1}$], which was:

(89)
$$Var[N_{j,t}] = \left(1-\frac{1}{L_j}\right)^2 Var[N_{j,t-1}] + \left(\frac{1}{L_j}\right)^2 Var[a_{j,t}] + 2\left(1-\frac{1}{L_j}\right)\left(\frac{1}{L_j}\right)Cov[N_{j,t-1}, a_{j,t}]$$

Substituting in the expressions for Var[$a_{j,t}$] and Cov[$N_{j,t-1}$, $a_{j,t}$] yields the recursive equation for the variances under the job-control rule:

(90)
$$\begin{aligned} Var[N_{j,t}] = {} & \left(1-\frac{1}{L_j}\right)^2 Var[N_{j,t-1}] + \left(\frac{1}{L_j}\right)^2\left(\sum_{k\in K}\left(\Phi^2_{jk}Var[N_{k,t-1}]\right) + \Sigma_{jj}\right) \\ & + \left(\frac{1}{L_j}\right)^2 \sum_{k\in K}\sum_{\substack{l\in K\\ l\neq k}}\left[\Phi_{jk}\Phi_{jl}Cov[N_{k,t-1}, N_{l,t-1}]\right] \\ & + 2\left(1-\frac{1}{L_j}\right)\left(\frac{1}{L_j}\right)\sum_{k\in K}\Phi_{jk}Cov[N_{j,t-1}, N_{k,t-1}]. \end{aligned}$$

5.6.5 Matrix Equations

As with the expectation calculations, we write all of the $Var[N_{j,t}]$ equations simultaneously in matrix form. Let:

- $\mathbf{S_t}$ be a square matrix with the $Var[N_{j,t}]$'s on the diagonal, and $Cov[N_{j,t}, N_{k,t}]$'s on the off-diagonal elements.
- Σ be a diagonal matrix with diagonal elements Σ_{jj}.
- Φ be a square matrix whose *(j,k)* element is Φ_{jk}.
- **D** be a diagonal matrix with the $1 / L_k$'s on the diagonal.
- **B** be a square matrix that equals **(I - D - DΦ),** where **I** is the identity matrix.

Then, a matrix equation for $\mathbf{S_t}$ in terms of $\mathbf{S_{t-1}}$ is:

$$\mathbf{S_t = BS_{t-1}B' + D\Sigma D}. \tag{91}$$

This equation may be checked by term-by-term multiplication. Infinitely iterating this equation yields:

$$\boxed{\mathbf{S} = \sum_{s=0}^{\infty} \mathbf{B^s (D\Sigma D) B'^s}.} \tag{92}$$

Equation 10 -- Variances and Covariances of Jobs at All Stations

(*Note*: The derivation of (92) assumes that the above matrix equation for $\mathbf{S_t}$ calculates the covariance between pairs of stations correctly. Otherwise, we would not be able to recurse on the equation for $\mathbf{S_t}$ to find the steady-state variances and covariances. Section 5.10 proves that the equation for $\mathbf{S_t}$ calculates the covariance terms correctly.)

But now, recall that we want to find the variances and covariances in terms of instructions at each station. We note that $P_{jt} = \sum_{l=1}^{N_{j,t}} X_{jt}^{l}$. Then we apply the formula for the variance of a random sum to find:

$$Var[P_j] = E[N_j]Var[X_j] + (E[X_j])^2 \mathbf{S_{jj}}. \tag{93}$$

Further, we found in Section 5.3 that $Cov[N_{k,t-1}, N_{l,t-1}] = Cov[P_{k,t-1}, P_{l,t-1}] / E[X_k]E[X_l]$. Then we can write:

$$Cov[P_{k,t-1}, P_{l,t-1}] = E[X_k]E[X_l]\mathbf{S_{kl}}. \tag{94}$$

These are formulas for the steady-state variances and covariances of all of the station workloads, as desired.□

5.7 Mixed Network Mechanics

In a *mixed network*, we have some stations operate according to an instruction-control rule, and other stations operate according to the job-shop control rule.

In general, we can apply the recursion equations we found in the previous sections. We use the instruction-control recursion equation at stations that operate under the instruction-control rule, and the job-control recursion equation at stations that operate under the job-control rule. Then, we write all of the recursion equations simultaneously in matrix form (whether they are instruction-rule or job-rule equations), and find recursive relationship that tells us what the expectation and variance of demands are for all stations.

However, to do this we must make some adjustments to the recursion equations. In a mixed model, we can have job-rule stations feed into instruction-rule stations, and vice versa. But, the LRS-MR model equations for job-rule stations maintain all statistics in terms of jobs, while the equations for instruction-rule stations maintain all statistics in terms of instructions. We will need to convert statistics in terms of jobs ($E[N_{k,t-1}]$, $Var[N_{k,t-1}]$) to statistics in terms of instructions ($E[P_{k,t-1}]$, $Var[P_{k,t-1}]$), and vice versa. Fortunately, this is simple to do. We make the following substitutions:

- *When a model equation calls for $E[N_{k,t-1}]$:* We use $E[N_{k,t-1}]$ directly if station k uses the job-control rule, since the model equations track this value under the job-control rule. If station j uses the instruction-control rule, the model equations track $E[P_{k,t-1}]$. In the latter case, we use the formula for $E[N_{k,t-1}]$ in terms of $E[P_{k,t-1}]$, which is:

$$E[N_{k,t-1}] = E[P_{k,t-1}]/E[X_{k,t-1}].$$

(This equation was derived in Section 5.2.)

- *When a model equation calls for $Var[N_{k,t-1}]$:* We use $Var[N_{k,t-1}]$ directly if station k uses the job-control rule. Otherwise, we use either the lower-bound or the upper-bound formula for $Var[N_{k,t-1}]$ in terms of $Var[P_{k,t-1}]$. The formulas are:

Lower bound: $$Var[N_{k,t-1}] \geq \frac{Var[P_{k,t-1}]}{(E[X_k])^2} - \frac{E[P_{k,t-1}]Var[X_k]}{L_k(E[X_k])^3}.$$

Upper bound: $Var[N_{k,t-1}] < \left(1-\frac{1}{L_k}\right)\frac{E[P_{k,t-1}]}{E[X_k]} + \frac{Var[P_{k,t-1}]}{E[X_k]^2} - \frac{E[P_{k,t-1}]Var[X_k]}{L_k(E[X_k])^3}$.

(These equations were derived in Section 5.3.)

- *When a model equation calls for* $\mathrm{Cov}[N_{k,t-1}, N_{l,t-1}]$, we make the following substitutions:

– If stations k and l both use job control, we use $\mathrm{Cov}[N_{k,t-1}, N_{l,t-1}]$ directly.

– If station k uses job control, and station l uses instruction control, we make the substitution $Cov[N_{k,t-1}, N_{l,t-1}] = Cov[N_k, P_l]/E[X_l]$.

– If stations k and l both use instruction control, we make the substitution $Cov[N_{k,t-1}, N_{l,t-1}] = Cov[P_k, P_l]/E[X_k]E[X_l]$.

(These equations were derived in Section 5.3.)

We then recalculate the recursion equations for each station, making these substitutions. Section 5.8 gives the results of these calculations for the expectation equations, and Section 5.9 gives the results of these calculations for the variance equations.

5.8 Mixed Network Expectations

5.8.1 Instruction-Rule Stations

Making the substitutions given in the previous section, we can derive the following recursion equation for instruction rule stations:

(95)

$$E[P_{j,t}] = \left(1-\frac{1}{L_j}\right)E[P_{j,t-1}] + \frac{1}{L_j}\left(\sum_{k\in K}\Phi_{jk}\rho_{k,t-1} + \mu_j\right),$$

where

$\rho_{k,t-1} = E[N_{k,t-1}]$, if k uses job-control;

$\rho_{k,t-1} = E[P_{k,t-1}]$, if k uses instruction-control;

$\Phi_{jk} = E[J_j]E[X_j]$, if k uses job-control;

$\Phi_{jk} = E[J_j]E[X_j]/E[X_k]$, if k uses instruction control; and

$\mu_j = E[R_j^e]E[J_{j,R}]E[X_j]$.

Equation 11 -- Recursion Equation for the Expectation of an Instruction-Rule Station

5.8.2 Job-Rule Stations

Making the substitutions given in the previous section, we eventually derive the following recursion equation for job-rule stations:

(96)

$$E[N_{j,t}] = \left(1 - \frac{1}{L_j}\right) E[N_{j,t-1}] + \frac{1}{L_j}\left(\sum_{k \in K} \Phi_{jk} \rho_{k,t-1} + \mu_j\right),$$

where

$\rho_{k,t-1} = E[N_{k,t-1}]$, if k uses job-control;

$\rho_{k,t-1} = E[P_{k,t-1}]$, if k uses instruction-control;

$\Phi_{jk} = E[J_j]$, if k uses job-control;

$\Phi_{jk} = E[J_j] / E[X_k]$, if k uses instruction control; and

$\mu_j = E[R_j^e] E[J_{j,R}]$.

Equation 12 -- Recursion Equation for the Expectation of a Job-Rule Station

5.8.3 The Matrix Equation for Expectations

Define the following vectors and matrices:

- **I** is the identity matrix.
- **D** is a diagonal matrix with the lead times on the diagonal.
- Φ is a matrix whose (j,k) entry is given by the formulas in the recursion equations above.
- ρ is a vector whose jth entry is given by the formulas in the recursion equations above.
- μ is a vector whose jth entry is given by the formulas in the recursion equations above.

Then we can rewrite all of the recursion equations in matrix form simultaneously in the following form:

(97)

$$\begin{aligned} \rho_t &= (\mathbf{I} - \mathbf{D})\rho_{t-1} + \mathbf{D}\Phi\rho_{t-1} + \mathbf{D}\mu, \\ &= \underbrace{(\mathbf{I} - \mathbf{D} + \mathbf{D}\Phi)}_{\text{Call this } \mathbf{B}}\rho_{t-1} + \mathbf{D}\mu, \\ &= \mathbf{B}\rho_{t-1} + \mathbf{D}\mu. \end{aligned}$$

The proof of this matrix equation is term-by-term multiplication. Infinitely iterating the above equation, and applying linear algebra theory, we find:

(98)

$$\boxed{\rho = (\mathbf{I} - \Phi)^{-1}\mu}$$

Equation 13 -- Results Vector Used to Calculate Expected Demands

But then, we have a results vector ρ whose jth entry is E[N_j] if station j uses the job-control rule, and E[P_j] if station j uses the instruction-control rule. In the former case, we can calculate E[P_j] by using the relationship E[P_j] = E[N_j]E[X_j]. This gives us a method to calculate the expectation of demand at all stations in the mixed-rule model, as desired.

5.9 Mixed Network Variances

5.9.1 Instruction-Rule Stations

Making the substitutions discussed in Section 5.7, we can derive the following recursion equation for an estimate of the variance of an instruction-rule station:

$$
\begin{aligned}
Var[P_{j,t}] = {} & \left(1-\frac{1}{L_j}\right)^2 Var[P_{j,t-1}] + \left(\frac{1}{L_j}\right)^2 \sum_{k\in K}\left[\Phi_{jk}^2\left(\mathbf{S}_{\mathbf{kk,t-1}} + u_k\right) + \sigma_{j,k,t}^2\right] \\
& + \left(\frac{1}{L_j}\right)^2 \sigma_{ej}^2 + \left(\frac{1}{L_j}\right)^2 \sum_{k\in K}\sum_{\substack{l\in K \\ l\neq k}}\left[\Phi_{jk}\Phi_{jl}\mathbf{S}_{\mathbf{kl,t-1}}\right] \\
& + 2\left(1-\frac{1}{L_j}\right)\left(\frac{1}{L_j}\right)\sum_{\substack{k\in K \\ k\neq j}}\left[\Phi_{jk}\mathbf{S}_{\mathbf{jl,t-1}}\right].
\end{aligned}
\tag{99}
$$

Equation 14 -- Recursion Equation for the Variance of an Instruction-Rule Station

In this equation:

- $\Phi_{jk} = E[J_{j,k}]E[X_j]/E[X_k]$ if station k uses instruction control.
- $\Phi_{jk} = E[J_{j,k}]E[X_j]$ if station k uses job control.
- $\sigma_{jkt}^2 = \frac{E[P_k]}{E[X_k]}\left(E[J_{jk}]Var[X_j] + Var[J_{j,k}](E[X_j])^2\right)$.
- $\sigma_{ej}^2 = E[R_j^e]\left(E[J_{jR}]Var[X_j] + Var[J_{jR}](E[X_j])^2\right) + Var[R_j^e]\left(E[J_{jR}]E[X_j]\right)^2$.
- $\mathbf{S}_{\mathbf{kk,t-1}} = Var[P_{k,t-1}]$ if station k uses instruction control.
- $\mathbf{S}_{\mathbf{kk,t-1}} = Var[N_{k,t-1}]$ if station k uses job control.
- $\mathbf{S}_{\mathbf{kl,t-1}} = Cov(N_{k,t-1}, N_{l,t-1})$ if stations k and l both use job control.
- $\mathbf{S}_{\mathbf{kl,t-1}} = Cov(P_{k,t-1}, N_{l,t-1})$ if station k uses instruction control and station j uses job control.
- $\mathbf{S}_{\mathbf{kl,t-1}} = Cov(P_{k,t-1}, P_{l,t-1})$ if stations k and l both use instruction control.

- $u_k = -\dfrac{E[P_k]Var[X_k]}{L_k E[X_k]}$ if station k uses instruction control with the lower-bound approximation.
- $u_k = \left(\dfrac{1}{L_k}\right)\left(1-\dfrac{1}{L_k}\right)\left(L_k E[P_k]\right) - \dfrac{E[P_k]Var[X_k]}{L_k E[X_k]}$ if station k uses instruction control with the upper-bound approximation.
- $u_k = 0$ if station k uses job control.

5.9.2 Job-Rule Stations

Making the substitutions discussed in Section 5.7, we can derive the following recursion equation for an estimate of the variance of the number of jobs produced at a job-rule station:

$$\begin{aligned} Var[N_{j,t}] = &\left(1-\frac{1}{L_j}\right)^2 Var[N_{j,t-1}] + \left(\frac{1}{L_j}\right)^2 \sum_{k\in K}\left[\Phi_{jk}^2\left(\mathbf{S}_{\mathbf{kk,t-1}} + u_k\right) + \sigma_{j,k,t}^2\right] \\ &+ \left(\frac{1}{L_j}\right)^2 \sigma_{ej}^2 + \left(\frac{1}{L_j}\right)^2 \sum_{k\in K}\sum_{\substack{l\in K \\ l\neq k}}\left[\Phi_{jk}\Phi_{jl}\mathbf{S}_{\mathbf{kl,t-1}}\right] \\ &+ 2\left(1-\frac{1}{L_j}\right)\left(\frac{1}{L_j}\right)\sum_{\substack{k\in K \\ k\neq j}}\left[\Phi_{jk}\mathbf{S}_{\mathbf{jl,t-1}}\right]. \end{aligned} \tag{100}$$

Equation 15 -- Recursion Equation for the Variance of a Job-Rule Station

In this equation:

- $\Phi_{jk} = E[J_{j,k}]/E[X_k]$ if station k uses instruction control.
- $\Phi_{jk} = E[J_{j,k}]$ if station k uses job control.
- $\sigma_{jkt}^2 = E[P_k]Var[J_{jk}]/E[X_k]$.
- $\sigma_{ej}^2 = E[R_j^e]\left(Var[J_{jR}]\right) + Var[R_j^e]\left(E[J_{jR}]\right)^2$.
- $\mathbf{S}_{\mathbf{kk,t-1}} = Var[P_{k,t-1}]$ if station k uses instruction control.
- $\mathbf{S}_{\mathbf{kk,t-1}} = Var[N_{k,t-1}]$ if station k uses job control.
- $\mathbf{S}_{\mathbf{kl,t-1}} = Cov(N_{k,t-1}, N_{l,t-1})$ if stations k and l both use job control.
- $\mathbf{S}_{\mathbf{kl,t-1}} = Cov(P_{k,t-1}, N_{l,t-1})$ if station k uses instruction control and station j uses job control.
- $\mathbf{S}_{\mathbf{kl,t-1}} = Cov(P_{k,t-1}, P_{l,t-1})$ if stations k and l both use instruction control.

- $u_k = -\dfrac{E[P_k]Var[X_k]}{L_k E[X_k]}$ if station k uses instruction control with the lower-bound approximation.
- $u_k = \left(\dfrac{1}{L_k}\right)\left(1-\dfrac{1}{L_k}\right)\left(L_k E[P_k]\right) - \dfrac{E[P_k]Var[X_k]}{L_k E[X_k]}$ if station k uses instruction control with the upper-bound approximation.
- $u_k = 0$ if station k uses job control.

5.9.3 The Matrix Equation for Variances

Define the following vectors and matrices:

- $\mathbf{S_t}$ be a square matrix whose $\mathbf{S_{j,j,t}}$ and $\mathbf{S_{j,k,t-1}}$ entries are given by the recursion equations above.
- **I** is the identity matrix.
- **D** is a diagonal matrix with the lead times on the diagonal.
- Φ is a matrix whose (j,k) entries are given by the formulas in the recursion equations above.
- **U** is a diagonal matrix with u_k's defined above on the diagonal.
- Σ is a diagonal matrix with diagonal elements $\Sigma_{jj} = \sigma_{ej}^2 + \sum_{k \in K} \sigma_{jkt}^2$.
- **B** be a square matrix that equals **(I - D - DΦ).**

Then we can rewrite all of the recursion equations in matrix form simultaneously in the following form:

(101) $$\mathbf{S_t = BS_{t-1}B' + (D\Phi)U(D\Phi)' + D\Sigma D.}$$

This equation may be checked by term-by-term multiplication. Infinitely iterating this recursion, we find:

(102) $$\boxed{\mathbf{S} = \sum_{s=0}^{\infty} \mathbf{B^s((D\Phi)U(D\Phi)' + DSD)B'^s}.}$$

Equation 16 -- Results Matrix Used to Estimate Network Covariances

But then, **S** is a results matrix whose entries are the following estimates:

- $\mathbf{S_{jj}}$ = Var[P_j] if station j uses instruction control.
- $\mathbf{S_{jj}}$ = Var[N_j] if station j uses job control. To find Var[P_j] in this case, we use the following relation: Var[P_j] = E[N_j]Var[X_j] + Var[N_j](E[X_j])^2.
- $\mathbf{S_{jk}}$ = Cov(P_j, P_k) if stations j and k both use instruction control.
- $\mathbf{S_{jk}}$ = Cov(N_j, P_k) if station j uses job control and station k uses instruction control. To find Cov(P_j, P_k) in this case, we use the following relation: Cov(P_j, P_k) = E[X_j] Cov(N_j, P_k).

- $\mathbf{S_{jk}} = \text{Cov}(N_j, N_k)$ if stations j and k both use instruction control. To find $\text{Cov}(P_j, P_k)$ in this case, we use the following relation: $\text{Cov}(P_j, P_k) = \text{E}[X_j]\text{E}[X_k]\ \text{Cov}(N_j, N_k)$.

These are estimates of the steady-state variances and covariances of demand for all of the stations, as desired. These estimates do assume that the above matrix equation for $\mathbf{S_t}$ calculates the covariance between pairs of stations correctly. Otherwise, we would not be able to recurse on the equation for $\mathbf{S_t}$ to find the steady-state variances and covariances. The following section proves that the equation for $\mathbf{S_t}$ calculates the covariance terms correctly.

5.10 Mixed Network Covariances

This section derives the covariances between pairs of stations, and shows that the formula for the station variances and covariances derived in the previous section correctly calculates the covariances.

5.10.1 Initial Development

Define the following variable: W_{it} is a measure of the work produced by station i at time t, and is:

- $W_{it} = P_{it}$ if station i uses the instruction-control rule, and;
- $W_{it} = N_{it}$ if station i uses the job-control rule.

We will derive a formula for $\text{Cov}[W_{i,t}, W_{j,t}]$ in terms of the workstation expectations, variances, and covariances at time *t-1*.

From Sections 5.1 and 5.4, we know that:

$$W_{j,t} = \left(1 - \frac{1}{L_j}\right) W_{j,t-1} + \frac{1}{L_j} A_{j,t}, \tag{103}$$

whether W_{jt} is measured in jobs or instructions. Here, A_{jt} measures the arrivals to station j at the start of period t, and is measured in instructions if station j uses the instruction-control rule, and measured in jobs if station j uses the job-control rule.

Then, we have that:

$$W_{i,t} + W_{j,t} = \left(1 - \frac{1}{L_i}\right) W_{i,t-1} + \frac{1}{L_i} A_{i,t} + \left(1 - \frac{1}{L_j}\right) W_{j,t-1} + \frac{1}{L_j} A_{j,t}. \tag{104}$$

Taking the variance of $(W_{i,t} + W_{j,t})$, we find:

$$
\begin{aligned}
Var[W_{i,t} + W_{j,t}] &= Var[W_{it}] + Var[W_{jt}] + 2Cov[W_{it}, W_{jt}] \\
&= \underbrace{\left(1-\frac{1}{L_i}\right)^2 Var[W_{i,t-1}] + \left(\frac{1}{L_i}\right)^2 Var[A_{i,t}] + 2\left(1-\frac{1}{L_i}\right)\left(\frac{1}{L_i}\right) Cov[W_{i,t-1}, A_{i,t}]}_{\text{We recognize this as Var}[W_{it}]} \\
&\quad + \underbrace{\left(1-\frac{1}{L_j}\right)^2 Var[W_{j,t-1}] + \left(\frac{1}{L_j}\right)^2 Var[A_{j,t}] + 2\left(1-\frac{1}{L_j}\right)\left(\frac{1}{L_j}\right) Cov[W_{j,t-1}, A_{j,t}]}_{\text{We recognize this as Var}[W_{jt}]} \\
&\quad + 2\left(1-\frac{1}{L_i}\right)\left(1-\frac{1}{L_j}\right) Cov[W_{i,t-1}, W_{j,t-1}] + 2\left(1-\frac{1}{L_i}\right)\left(\frac{1}{L_j}\right) Cov[W_{i,t-1}, A_{j,t}] \\
&\quad + 2\left(\frac{1}{L_i}\right)\left(1-\frac{1}{L_j}\right) Cov[A_{i,t}, W_{j,t-1}] + 2\left(\frac{1}{L_i}\right)\left(\frac{1}{L_j}\right) Cov[A_{i,t}, A_{j,t}].
\end{aligned}
\tag{105}
$$

Equating terms, and solving for Cov[W_{it}, W_{jt}], we find:

$$
\begin{aligned}
Cov[W_{it}, W_{jt}] &= \left(1-\frac{1}{L_i}\right)\left(1-\frac{1}{L_j}\right) Cov[W_{i,t-1}, W_{j,t-1}] + \left(1-\frac{1}{L_i}\right)\left(\frac{1}{L_j}\right) Cov[W_{i,t-1}, A_{j,t}] \\
&\quad + \left(\frac{1}{L_i}\right)\left(1-\frac{1}{L_j}\right) Cov[A_{i,t}, W_{j,t-1}] + \left(\frac{1}{L_i}\right)\left(\frac{1}{L_j}\right) Cov[A_{i,t}, A_{j,t}].
\end{aligned}
\tag{106}
$$

To calculate the terms in this expression, we will need to calculate Cov[$A_{i,t}$, $A_{j,t}$] and Cov[W_{it}, A_{jt}].

5.10.2 The Covariance of Arrivals

$A_{j,t}$ has the following form:

$$
A_{j,t} = \underbrace{\sum_{k \in K} \sum_{l=1}^{N_{k,t-1}} Y_{jkt}^{l}}_{\substack{\text{Arrivals from} \\ \text{other stations}}} + \underbrace{\sum_{l=1}^{R_{jt}^{e}} Y_{j,R,t}^{l}}_{\substack{\text{Arrivals from} \\ \text{outside the system}}}
\tag{107}
$$

Here, the N's are the incoming requests from the other stations, $R_{j,t}$ is incoming requests from outside the network, and the Y's are independent, identically distributed random variables representing the work per request from each source. (The Y's are the number of jobs per request if the station uses the job-control rule. They are summations of instructions if the station uses the instruction-control rule.)

By assumption, there is no dependence between any arrivals from outside the network and any other arrivals. Then, to find $Cov[A_{it}, A_{jt}]$, it is sufficient to find:

$$Cov[A_{i,t}, A_{j,t}] = Cov\left[\sum_{k\in K}\sum_{l=1}^{N_{k,t-1}} Y_{ikt}^{l}, \sum_{k\in K}\sum_{l=1}^{N_{k,t-1}} Y_{jkt}^{l}\right]. \tag{108}$$

To do so, we will extend Lemma 5 to calculate $Var(A_{it} + A_{jt})$, and use the resulting formula for $Var(A_{it} + A_{jt})$ to calculate $Cov[A_{it}, A_{jt}]$.

First, define T to be:

$$T = \sum_{k\in K}\sum_{l=1}^{N_{k,t-1}} Y_{ikt}^{l} + \sum_{k\in K}\sum_{l=1}^{N_{k,t-1}} Y_{jkt}^{l}. \tag{109}$$

The "Law of Total Variance" (see Lemma 2), tells us that

$$Var[T] = Var[E(T|N)] + E[Var(T|N)], \tag{110}$$

where N is some event. Here, let N be the event $\{N_{k,t-1} = n_k, \forall k\}$.

Now, using the linearity of expectation,

$$E(T|N) = \sum_{k\in K} N_k E[Y_{ik}] + \sum_{k\in K} N_k E[Y_{jk}]. \tag{111}$$

Then, noting that $E[Y_{ik}]$ and $E[Y_{jk}]$ are constants, and applying the bilinearity principle of covariance, we find that:

$$\begin{aligned} Var[E(T|N)] = &\sum_{k\in K}\sum_{l\in K} E[Y_{ik}]E[Y_{il}]Cov[N_k, N_l] + \sum_{k\in K}\sum_{l\in K} E[Y_{jk}]E[Y_{jl}]Cov[N_k, N_l] \\ &+ 2\sum_{k\in K}\sum_{l\in K} E[Y_{ik}]E[Y_{jl}]Cov[N_k, N_l]. \end{aligned} \tag{112}$$

Next, recalling that the variance of a fixed sum of independent random variables is the sum of their variances, we find:

$$\begin{aligned} Var(T|N) &= \sum_{k\in K}\sum_{l=1}^{N_k} Var[Y_{ik}] + \sum_{k\in K}\sum_{l=1}^{N_k} Var[Y_{jk}] \\ &= \sum_{k\in K} N_k Var[Y_{ik}] + \sum_{k\in K} N_k Var[Y_{jk}], \end{aligned} \tag{113}$$

and, taking the expectation over N, we find:

$$E[Var(T \mid N)] = \sum_{k\in K} E[N_k]Var[Y_{ik}] + \sum_{k\in K} E[N_k]Var[Y_{jk}], \tag{114}$$

Adding these terms together, we find:

$$
\begin{aligned}
Var[T] &= Var[E(T|N)] + E[Var(T|N)] \\
&= \sum_{k\in K}\sum_{l\in K} E[Y_{ik}]E[Y_{il}]Cov[N_k, N_l] + \sum_{k\in K}\sum_{l\in K} E[Y_{jk}]E[Y_{jl}]Cov[N_k, N_l] \\
&\quad + 2\sum_{k\in K}\sum_{l\in K} E[Y_{ik}]E[Y_{jl}]Cov[N_k, N_l] + \sum_{k\in K} E[N_k]Var[Y_{ik}] + \sum_{k\in K} E[N_k]Var[Y_{jk}].
\end{aligned} \tag{115}
$$

Now, let us group together the terms of $Var[T]$ as follows:

$$
\begin{aligned}
Var[T] &= \left(\sum_{k\in K}\sum_{l\in K} E[Y_{ik}]E[Y_{il}]Cov[N_k, N_l] + \sum_{k\in K} E[N_k]Var[Y_{ik}]\right) \\
&\quad + \left(\sum_{k\in K}\sum_{l\in K} E[Y_{jk}]E[Y_{jl}]Cov[N_k, N_l] + \sum_{k\in K} E[N_k]Var[Y_{jk}]\right) \\
&\quad + 2\sum_{k\in K}\sum_{l\in K} E[Y_{ik}]E[Y_{jl}]Cov[N_k, N_l],
\end{aligned} \tag{116}
$$

which we recognize to be:

$$
Var[T] = Var[\sum_{k\in K}\sum_{l=1}^{N_{k,t-1}} Y_{ikt}^l] + Var[\sum_{k\in K}\sum_{l=1}^{N_{k,t-1}} Y_{jkt}^l] + 2\sum_{k\in K}\sum_{l\in K} E[Y_{ik}]E[Y_{jl}]Cov[N_k, N_l]. \tag{117}
$$

Now, by the definition of covariance, we know:

$$
Var[T] = Var[\sum_{k\in K}\sum_{l=1}^{N_{k,t-1}} Y_{ikt}^l] + Var[\sum_{k\in K}\sum_{l=1}^{N_{k,t-1}} Y_{jkt}^l] + 2Cov[\sum_{k\in K}\sum_{l=1}^{N_{k,t-1}} Y_{ikt}^l, \sum_{k\in K}\sum_{l=1}^{N_{k,t-1}} Y_{jkt}^l]. \tag{118}
$$

So, equating terms, we solve for $Cov[A_{it}, A_{jt}]$:

$$
Cov[A_{it}, A_{jt}] = Cov\left[\sum_{k\in K}\sum_{l=1}^{N_{k,t-1}} Y_{ikt}^l, \sum_{k\in K}\sum_{l=1}^{N_{k,t-1}} Y_{jkt}^l\right] = \sum_{k\in K}\sum_{l\in K} E[Y_{ik}]E[Y_{jl}]Cov[N_{k,t-1}, N_{l,t-1}]. \tag{119}
$$

Now, this expression for $Cov[A_{it}, A_{jt}]$ is in terms of $E[Y_{ik}]$ and $Cov[N_{k,t-1}, N_{l,t-1}]$. In section 5.8, we found that we replace the $E[Y_{ik}]$ terms with:

- $E[J_{ik}]E[X_i]$ if station i uses the instruction-control rule.
- $E[J_{ik}]$ if station i uses the job-control rule.

Next, in section 5.9, we found that, for $k \neq l$, we can replace the $Cov[N_{k,t-1}, N_{l,t-1}]$ term with:

- $Cov[N_{k,t-1}, N_{l,t-1}]$, if stations k and l both use the job-control rule;
- $Cov[P_{k,t-1}, N_{l,t-1}]$ / $E[P_k]$ if station k uses the instruction-control rule and station l uses the job-control rule; and

- $\text{Cov}[P_{k,t-1},P_{l,t-1}] / \text{E}[P_k]\text{E}[P_l]$ if stations k and l both use the instruction-control rule.

Finally, for $k = l$, the $\text{Cov}[N_{k,t-1},N_{l,t-1}]$ becomes $\text{Var}[N_{k,t-1}]$, and we replace this term with:

- $\text{Var}[N_{k,t-1}]$ if station k uses the job-control rule; and
- $(\text{Var}[P_{k,t-1}] + u_k) / \text{E}[X_k]^2$ if station k uses one of the instruction-control rules. (Recall that u_k is one of the two variance correction terms discussed in section 5.3 of the LRS-MR model paper.)

But then, we can rewrite the expression for $\text{Cov}[A_{it},A_{jt}]$ as follows:

$$Cov[A_{it},A_{jt}] = \sum_{k\in K}\sum_{l\in K}\Phi_{ik}\Phi_{jl}\mathbf{S}_{\mathbf{k,l,t-1}} + \sum_{k\in K}\Phi_{ik}\Phi_{jk}\mathbf{U}_{\mathbf{kk}}, \tag{120}$$

where:

- $\Phi_{\mathbf{ik}} = \text{E}[J_{ik}]$ if stations i and k both use the job-control rule.
- $\Phi_{\mathbf{ik}} = \text{E}[J_{ik}]\text{E}[X_i]$ if station i uses the instruction-control rule and station k uses the job-control rule.
- $\Phi_{\mathbf{ik}} = \text{E}[J_{ik}] / \text{E}[X_k]$ if station i uses the job-control rule and station k uses the instruction-control rule.
- $\Phi_{\mathbf{ik}} = \text{E}[J_{jk}]\text{E}[X_i] / \text{E}[X_k]$ if stations i and k both use the job-control rule.
- $\mathbf{S}_{\mathbf{k,l,t-1}} = \text{Cov}[N_{k,t-1}, N_{l,t-1}]$, if stations k and l both use the job-control rule (recall that $\text{Cov}[N_{k,t-1},N_{l,t-1}]$ terms become $\text{Var}[N_{k,t-1}]$ terms when $k = l$);
- $\mathbf{S}_{\mathbf{k,l,t-1}} = \text{Cov}[P_{k,t-1}, N_{l,t-1}] / \text{E}[P_k]$ if station k uses the instruction-control rule and station l uses the job-control rule;
- $\mathbf{S}_{\mathbf{k,l,t-1}} = \text{Cov}[P_{k,t-1},P_{l,t-1}] / \text{E}[P_k]\text{E}[P_l]$ if stations k and l both use the instruction-control rule;
- $\mathbf{U}_{\mathbf{kk}} = 0$ if station k uses the job-control rule; and
- $\mathbf{U}_{\mathbf{kk}} = u_k$, if station k uses one of the instruction-control rules, and where u_k is the appropriate variance correction term, as discussed in section 5.3.

5.10.3 The Covariance of Arrivals and Workload

Now, we need to calculate the covariance of work arrivals with workload in the previous period, or $Cov[A_{i,t-1},W_{j,t-1}]$. To do so, let us consider $Cov[A_{i,t-1},A_{j,j,t-1}]$, which is the covariance of the arrivals at station i with the arrivals station j sends to itself. We can write this as:

$$Cov[A_{i,t},A_{j,j,t}] = Cov\left[\sum_{k\in K}\sum_{l=1}^{N_{k,t-1}}Y_{ikt}^{l}, \sum_{l=1}^{N_{j,t-1}}Y_{jjt}^{l}\right]. \tag{121}$$

In the previous subsection, we found that the covariance of all the arrivals at station i with all of the arrivals to station j is:

$$(122) \qquad Cov[A_{it}, A_{jt}] = Cov\left[\sum_{k\in K}\sum_{l=1}^{N_{k,t-1}} Y_{ikt}^{l}, \sum_{k\in K}\sum_{l=1}^{N_{k,t-1}} Y_{jkt}^{l}\right] = \sum_{k\in K}\sum_{l\in K}\Phi_{ik}\Phi_{jl}\mathbf{S}_{\mathbf{k,l,t-1}} + \sum_{k\in K}\Phi_{ik}\Phi_{jk}\mathbf{U}_{\mathbf{kk}}.$$

But then, by the bilinearity principle of covariance, the covariance of all the arrivals at station i with the single arrival of the work station j sends to itself is:

$$(123) \qquad Cov[A_{it}, A_{j,j,t}] = \sum_{k\in K}\Phi_{ik}\Phi_{jj}\mathbf{S}_{\mathbf{k,j,t-1}} + \Phi_{ij}\Phi_{jj}\mathbf{U}_{\mathbf{jj}}.$$

Now, $\mathbf{S}_{\mathbf{k,j,t-1}}$ is a linear function of $Cov[W_{k,t-l}, W_{j,t-l}]$. Then, the bilinearity principle implies that to change $Cov[A_{i,t}, A_{j,j,t}]$ to $Cov[A_{i,t}, W_{j,t-1}]$, we simply drop the Φ_{jj} terms from the equation for $Cov[A_{i,t}, A_{j,j,t}]$. This gives us:

$$(124) \qquad Cov[A_{it}, W_{j,t-1}] = \sum_{l\in K}\Phi_{ik}\mathbf{S}_{\mathbf{k,j,t-1}} + \Phi_{ij}\mathbf{U}_{\mathbf{jj}}.$$

5.10.4 A Formula for the Covariance Terms

Recall that a formula for Cov$[W_{it}, W_{jt}]$ is:

$$(125) \qquad \begin{aligned} Cov[W_{it}, W_{jt}] = {} & \left(1-\frac{1}{L_i}\right)\left(1-\frac{1}{L_j}\right)Cov[W_{i,t-1}, W_{j,t-1}] + \left(1-\frac{1}{L_i}\right)\left(\frac{1}{L_j}\right)Cov[W_{i,t-1}, A_{j,t}] \\ & + \left(\frac{1}{L_i}\right)\left(1-\frac{1}{L_j}\right)Cov[A_{i,t}, W_{j,t-1}] + \left(\frac{1}{L_i}\right)\left(\frac{1}{L_j}\right)Cov[A_{i,t}, A_{j,t}]. \end{aligned}$$

Using the formulas of the previous two subsections for Cov$[W_{i,t-l}, A_{jt}]$ and Cov$[A_{it}, A_{jt}]$, we find:

$$(126) \qquad \begin{aligned} Cov[W_{it}, W_{jt}] = {} & \left(1-\frac{1}{L_i}\right)\left(1-\frac{1}{L_j}\right)Cov[W_{i,t-1}, W_{j,t-1}] \\ & + \left(1-\frac{1}{L_i}\right)\left(\frac{1}{L_j}\right)\left(\sum_{k\in K}\left(\Phi_{jk}\mathbf{S}_{\mathbf{k,i,t-1}}\right) + \Phi_{ji}\mathbf{U}_{\mathbf{ii}}\right) \\ & + \left(\frac{1}{L_i}\right)\left(1-\frac{1}{L_j}\right)\left(\sum_{k\in K}\left(\Phi_{ik}\mathbf{S}_{\mathbf{k,j,t-1}}\right) + \Phi_{ij}\mathbf{U}_{\mathbf{jj}}\right) \\ & + \left(\frac{1}{L_i}\right)\left(\frac{1}{L_j}\right)\left(\sum_{k\in K}\sum_{l\in K}\Phi_{ik}\Phi_{jl}\mathbf{S}_{\mathbf{k,l,t-1}} + \sum_{k\in K}\Phi_{ik}\Phi_{jk}\mathbf{U}_{\mathbf{kk}}\right). \end{aligned}$$

Finally, we can write all of the non-variance Cov$[W_{it}, W_{jt}]$ equations in matrix form as follows:

(127) $$\mathbf{S}_t = \mathbf{B}\mathbf{S}_{t-1}\mathbf{B}' + (\mathbf{D}\Phi)\mathbf{U}(\mathbf{D}\Phi)' + \mathbf{D}\Sigma\mathbf{D},$$

where all the matrix variables are the same as they were for Equation 19. By term-by-term multiplication, we can check that the *(i,j)* entry of $\mathbf{S}_t$ is the recursion equation above. ■

This completes the derivation of the LRS-MR model. □

6. Appendix: Lemmas Used in Model Derivations

6.1 Expectation of a Random Sum of Random Variables

Lemma 1. *Let N be a random variable with finite expectation, and X_i be a set of independent, identically distributed random variables, independent of N, that have a common mean E[X]. Define* $Q = \sum_{i=1}^{N} X_i$. *Then* $E[Q] = E[N]E[X]$.

Proof. (c.f. Rice, 1995, pp. 137-138.) We first prove the following result: $E[Y] = E[E(Y|X)]$. (This result is sometimes called "the law of total expectation.") This law states that. To prove this result, we will show that:

(128) $$E(Y) = \sum_x E(Y|X = x)p_x(x),$$

$$\text{where } E(Y|X = x) = \sum_y y p_{y|x}(y|x).$$

The proposed formula for E(Y) is a double summation, and we can interchange the order of this summation. Doing so yields:

(129) $$\sum_x E(Y|X = x)p_X(x) = \sum_y y \sum_x p_{Y|X}(y|x)p_X(x).$$

Now, by the definition of conditional probability, we have:

(130) $$p_Y(y) = \sum_x p_{Y|X}(y|x)p_X(x).$$

Substituting, we find that:

$$\sum_y y \sum_x p_{Y|X}(y|x) p_X(x) = \sum_y y p_Y(y) = E(Y), \tag{131}$$

which is the desired result.

We now consider $E[Q]$. Using the result, $E[Q] = E[E(T|N)]$. Using the linearity of expectation, $E(Q|N=n) = nE[X]$, and $E(Q|N) = NE[X]$. Then we have:

$$E[Q] = E[E(Q|N)] = E[NE(X)] = E[N]E[X], \tag{132}$$

which is the desired result. □

6.2 Variance of a Random Sum of Random Variables

Lemma 2. *Let N be a random variable with finite expectation and a finite variance. Let X_i be a set of independent, identically distributed random variables, independent of N, that have a common mean E[X], and a common variance, Var[X].*

Define $Q = \sum_{i=1}^{N} X_i$. *Then* $Var[Q] = E[N]Var[X] + (E[X])^2 Var[N]$.

Proof. (c.f. Rice, 1995, pp. 138-139.) We first prove the following result:
$Var[Y] = Var[E(Y|X)] + E[Var(Y|X)]$. (This result can be thought of as the "law of total variance.")
By definition of variance, we have:

$$Var(Y|X) = E(Y^2|X=x) - [E(Y)|X=x]^2. \tag{133}$$

Then the expectation of Var(Y|X) is:

$$E[Var(Y|X) = E[E(Y^2|X)] - E\{[E(Y|X)]^2\}. \tag{134}$$

Similarly, the variance of a conditional expectation is:

$$Var[E(Y|X)] = E\{[E(Y|X)]^2\} - \{E[E(Y|X)]\}^2. \tag{135}$$

Next, we can use the law of total expectation to rewrite Var(Y) as:

$$Var(Y) = E(Y^2) - [E(Y)]^2 = E[E(Y^2|X)] - \{E[E(Y|X)]\}^2. \tag{136}$$

Substituting, we find that:

$$\begin{aligned} Var(Y) &= E[E(Y^2|X)] - \{E[E(Y|X)]\}^2 \\ &= E[E(Y^2|X)] - E\{[E(Y|X)]^2\} + E\{[E(Y|X)]^2\} - \{E[E(Y|X)]\}^2 \\ &= E[Var(Y|X)] + Var[E(Y|X)], \end{aligned} \tag{137}$$

which is the desired result.

Now consider $Var[Q]$. Using the result, we have that

(138) $$Var[Q] = Var[E(T|N)] + E[Var(T|N)].$$

Because $E(Q|N) = NE(X)$, we have that

(139) $$Var[E(Q|N)] = [E(X)]^2 Var(N).$$

Further, the fact that the X_i's are independent allows us to write:

(140) $$Var(Q|N) = Var\sum_{i=1}^{N} X_i = N(Var[X]),$$

and, taking expectations, we find that:

(141) $$E[Var(Q|N)] = E(N)Var(X).$$

Substituting into the expression for $Var[T]$, we find:

(142) $$Var[Q] = E[N]Var[X] + (E[X])^2 Var[N],$$

which is the desired result. □

6.3 Uniform Distribution of Arrivals in a Poisson Process

Lemma 3. *Let X_i be a set of independent, identically distributed random variables, that come from an exponential distribution with parameter λ. Consider a queue containing N jobs, where N is some positive integer, and where each job has length X_i. Then the breakpoints of the first $N-1$ jobs in the queue will be uniformly distributed.*

Proof. (c.f. Gallager, 1995, p. 45.) First, note that the queue has a total length of $Q = \sum_{i=1}^{N} X_i$. Next, define S_i to be the location in the queue of the breakpoint between the ith and $i+1$th job. We know $S_N = Q$, since the end of the last job marks the end of the queue.

We will calculate the joint distribution of $S_1, S_2, \ldots S_{N-1}$, which *is* $f(S \mid N-1) = f(S_1=s_1, \ldots, S_{N-1}=s_{n-1} \mid N-1$ breakpoints in Q*)*. Now, for a small δ, $f(S \mid N-1)\delta$ approximately equals the probability of no breakpoints in the intervals $(0, s_1]$, $(s_1+\delta, s_2]$, ... ,$(s_{N-1}+\delta, Q]$, and precisely one breakpoint in each of the intervals $(s_i, s_i+\delta]$, $i = 1$ to $N-1$, conditional on the event that exactly $N-1$ breakpoints occurred.

We first consider the unconditional probability of $f(s_1, \ldots s_{N-1})\delta$. Since the X_i's are exponential with parameter λ, the probability of no arrivals in one of the $(s_i+\delta, s_{i+1}]$ intervals equals $\exp[-\lambda(s_{i+1} - s_i - \delta)]$. Similarly, the probability of one of the arrivals falling in one of the $(s_i, s_i+d]$ intervals is $\lambda\delta\exp[-\lambda\delta]$. Then, the unconditional probability is simply the product of all the $\exp[-\lambda(s_{i+1} - s_i - \delta)]$ and $\lambda\delta\exp[-\lambda\delta]$ terms. Now, there is one exponential term for each subinterval of $(0, Q]$, so multiplying them together yields $\exp[-\lambda Q]$. Further, there are $N - 1$ $(\lambda\delta)$ terms, so we have:

$$f(s_1,\ldots,s_{N-1})\delta = (\lambda\delta)^{N-1}\exp[-\lambda Q]. \tag{143}$$

Now, using conditional probability, we know that:

$$f(s_1,\ldots,s_{N-1} \mid N-1 \text{ breakpoints in Q})\delta = \frac{f(s_1,\ldots,s_{N-1})\delta}{P(N-1 \text{ breakpoints in Q})}. \tag{144}$$

Since the X_i's are exponentially distributed, $P(N - 1$ breakpoints) is given by a Poisson distribution. (Effectively, $P(N - 1$ breakpoints) is the probability of $N - 1$ arrivals of a Poisson process with parameter λ in an interval of length Q). Then we have:

$$\begin{aligned} f(s_1,\ldots,s_{N-1} \mid N-1 \text{ breakpoints in Q})\delta &= \frac{f(s_1,\ldots,s_{N-1})\delta}{P(N-1 \text{ breakpoints in Q})}, \\ &= \frac{(\lambda\delta)^{N-1}\exp[-\lambda Q]}{\frac{(\lambda Q)^{N-1}\exp[-\lambda Q]}{(N-1)!}}, \\ &= \frac{(\lambda\delta)^{N-1}(N-1)!}{(\lambda Q)^{N-1}}. \end{aligned} \tag{145}$$

Dividing by δ and taking the limit as $\delta \to 0$, we find:

$$f(s_1,\ldots,s_{N-1} \mid N-1 \text{ breakpoints in Q}) = \frac{(N-1)!}{Q^{N-1}}, 0 < s_1 < \ldots < s_{N-1} < Q. \tag{146}$$

Then $f(S \mid N - 1)$ has a uniform distribution, as desired. □

6.4 An Upper Bound on the Variance of the Number of Jobs Processed Per Period, for "Nice Distributions" of the Number of Instructions Per Job

Lemma 4. *Suppose that X_k, a nonnegative random variable for the distribution for the number of instructions per job at station k, satisfies the following condition: For all values of a constant $t_0 \geq 0$,*

$$E[X_k - t_0 | X > t_0] \leq E[X_k].$$

Then the following result holds:

$$Var[N_k] \leq E[q_k]\left(\frac{1}{L_k} - \frac{1}{L_k^2}\right) + \frac{1}{L_k^2} Var[q_k],$$

where N_k is the number of jobs contained in the first $1 / L_k$ fraction of instructions in the queue at station k, and q_k is the total number of jobs in the queue.

Discussion. In words, the condition $E[X_k - t_0 | X > t_0] \leq E[X_k]$ means the following: Suppose we arrive at station k at a random time, and wait for the current job in the queue to finish. Further, suppose we know that station k has been working on the current job for at least t_0 time units. Then the condition says that the expectation of the time remaining until the job is finished is less than the unconditional expected time to process a job at station k.

Distributions which satisfy this property can be thought of as "nice" distributions. Most common distributions satisfy this property, including deterministic, uniform, triangular, normal, and beta distributions. The exponential distribution satisfies this property with equality: by definition, the fact that we have been waiting t_0 for the completion of a job tells us nothing about when the job will be done.

An example distribution in which the property is not satisfied is the following: suppose that the time to complete a job is either two seconds or five hours. If we wait for more than two seconds, we expect to wait a long time before the current job is completed.

Proof. We consider two different nonnegative distributions: X_k, which satisfies the condition of the lemma, and X_k', which is an exponential distribution. Both distributions have the same mean, $E[X_k]$. The variance of X_k is not known; the variance of X_k' is $(E[X_k])^2$. However, an established result from queuing theory is that:

$$\frac{Var[X_k]}{E[X_k]} < \frac{Var[X_k']}{E[Y_k']}, \tag{147}$$

since X_k satisfies the property that $E[X_k - t_0 | X > t_0] \leq E[X_k]$. Since X_k and X_k' have the same mean, we have that $Var[X_k] < Var[X_k']$.

Now, define N_k to be the number of jobs whose endpoints are in the first Q_k / L_k instructions of the work queue, given that the distribution of the number of instructions per job is X_k. Similarly, define N_k'

to be the number of jobs whose endpoints are in the first Q_k / L_k instructions of the work queue, given that the distribution of the number of instructions per job is X_k'. Mathematically, we define N_k to be:

$$N_k = \left\{ N_k : \sum_{i=1}^{N_k} X_{i,k} \le \frac{Q_k}{L_k} \le \sum_{i=1}^{N_k+1} X_{i,k} \right\}, \tag{148}$$

and the definition of N'_k is similar. We want to show that $Var[N_k] \le Var[N_k']$. We will use the Law of Total Variance to prove this result.

Recall this law states that $Var[Y] = Var[E(Y|X)] + E[Var(Y|X)]$. Here, let $Y = N_k$ or N'_k as appropriate, and let X be the event that Q_k and q_k equal certain fixed values.

Var[E(Y|X)]: Given q_k, $E[N_k] = E[N_k'] = q_k / L_k$. Then,

$$Var[E(N_k|X)] = Var[E(N'_k|X)] = Var(q_k) / L_k^2. \tag{149}$$

E[Var(Y|X)]: Using the definition of variance for discrete distributions, we have that:

$$Var(N_k|X) = \sum_{N_k=1}^{q_k} \left(N_k - \frac{q_k}{L_k} \right)^2 p(N_k|q_k, L_k). \tag{150}$$

The expression for $\mathrm{Var}(N'_k \mid X)$ is similar.

Recall the probability that N_k equals a particular value, n_k, is the probability that the sum of the instructions of the first n_k jobs in the queue is less than or equal to Q_k / L_k, and that the sum of the instructions of the first $n_k + 1$ jobs is greater than Q_k / L_k. Then we can rewrite the above equation as:

$$Var(N_k|X) = \sum_{N_k=1}^{q_k} \left(N_k - \frac{q_k}{L_k} \right)^2 p\left(\sum_{i=1}^{N_k} X_{i,k} \le \frac{Q_k}{L_k} \le \sum_{i=1}^{N_k+1} X_{i,k} \right). \tag{151}$$

If we take the expectation of this expression, we find that:

$$E[Var(N_k|X)] = \sum_{N_k=0}^{q_k} \left(N_k - \frac{E[q_k]}{L_k} \right)^2 p\left(\sum_{j=1}^{N_k} X_{j,k} \le \frac{E[Q_k]}{L_k} \le \sum_{j=1}^{N_k+1} X_{j,k} \right). \tag{152}$$

The expression for $\mathrm{Var}(N'_k \mid X)$ is similar. Then, the fact that $Var[X_j] \le Var[X'_j]$ implies that:

$$
\begin{aligned}
E[Var(N_k|X)] &= \sum_{N_k=0}^{q_k}\left(N_k - \frac{E[q_k]}{L_k}\right)^2 p\left(\sum_{j=1}^{N_k} X_{j,k} \le \frac{E[Q_k]}{L_k} \le \sum_{j=1}^{N_k+1} X_{j,k}\right) \\
&\le \sum_{N'_k=0}^{q_k}\left(N'_k - \frac{E[q_k]}{L_k}\right)^2 p\left(\sum_{j=1}^{N'_k} X'_{j,k} \le \frac{E[Q_k]}{L_k} \le \sum_{j=1}^{N'_k+1} X'_{j,k}\right) \\
&\le E[Var(N'_k|X)].
\end{aligned} \tag{153}
$$

The inequality follows from the following argument: since $Var[X_j] \le Var[X'_j]$, the overall probability that $\sum_{j=1}^{N_k} X_{j,k}$ takes on values comparatively far from $E[Q_k]/L_k$ is less than the probability that $\sum_{j=1}^{N_k} X'_{j,k}$ takes on values comparatively far from $E[Q_k]/L_k$. The inequality follows.

Var(Y) = Var[E(Y|X)] + E(Var(Y|X): We have shown that $Var[E(N_k|X)] = Var[E(N'_k|X)]$, and that $E[Var(N_k|X)] \le E[Var(N'_k|X)]$. Adding these two terms together, we find that:

$$
\begin{aligned}
Var[N_k] &= Var[E(N_k|X)] + E(Var[N_k|X) \\
&\le Var[E(N'_k|X)] + E(Var[N'_k|X) = Var[N'_k],
\end{aligned} \tag{154}
$$

Now, we found in section 5.3 that

$$
Var[N'_k] \le E[q_k]\left(\frac{1}{L_k} - \frac{1}{L_k^2}\right) + \frac{1}{L_k^2} Var[q_k]. \tag{155}
$$

(Recall that the result follows from the fact that X'_k is an exponential distribution.) The result of the lemma immediately follows. □

6.5 Covariance of a Sum of Random Sums of Random Variables

Lemma 5. *Let N_k be a set K of random variables, each with a finite expectation and a finite variance. Let X_{ik} be a set of independent, identically distributed random variables, independent from every N_k, that have a common mean $E[X_k]$, and a common variance, $Var[X_k]$.*

Define $T = \sum_{k\in K}\sum_{i=1}^{N_k} X_{ik}$. *Then:*

$$
Var[T] = \sum_{k\in K}\left(E[N_k]Var[X_k] + (E[X_k])^2 Var[N_k]\right) + \sum_{k\in K}\sum_{\substack{l\in K \\ l\ne k}} E[X_k]E[X_l]Cov[N_k, N_l].
$$

Proof. We assume the following result: $Var[Y] = Var[E(Y|X)] + E[Var(Y|X)]$. (This result was proved in the development of Lemma 2.) Let N be the event $\{N_k = n_k, \forall k\}$. Using this result, we have that:

$$Var[T] = Var[E(T|N)] + E[Var(T|N)]. \tag{156}$$

Now, using the linearity of expectation,

$$E(T|N) = \sum_{k \in K} N_k E[X_k]. \tag{157}$$

But then, noting that $E[X_k]$ is a constant, and applying the bilinearity principle of covariance, we find that:

$$\begin{aligned} Var[E(T|N)] &= Var\left[\sum_{k \in K} N_k E[X_k]\right], \\ &= \sum_{k \in K} (E[X_k])^2 Var[N_k] + \sum_{k \in K} \sum_{\substack{l \in K \\ l \neq k}} E[X_k] E[X_l] Cov[N_k, N_l]. \end{aligned} \tag{158}$$

Next, recalling that the variance of a fixed sum of independent random variables is the sum of their variances, we have:

$$Var(T|N) = Var\left(\sum_{k \in K} \sum_{i=1}^{N_k} X_{ik}\right) = \sum_{k \in K} \sum_{i=1}^{N_k} Var[X_k] = \sum_{k \in K} N_k Var[X_k], \tag{159}$$

and, taking the expectation over N, we find:

$$E[Var(T|N)] = E\left(\sum_{k \in K} N_k Var[X_k]\right) = \sum_{k \in K} E[N_k] Var[X_k]. \tag{160}$$

We therefore have:

$$\begin{aligned} Var[T] &= Var[E(T|N)] + E[Var(T|N)] \\ &= \sum_{k \in K} \left(E[N_k] Var[X_k] + (E[X_k])^2 Var[N_k]\right) + \sum_{k \in K} \sum_{\substack{l \in K \\ l \neq k}} E[X_k] E[X_l] Cov[N_k, N_l], \end{aligned} \tag{161}$$

which is the desired result. □

Chapter 6: Steady-State Optimization

1. Introduction

Previous chapters have focused on the steady-state performance evaluation of MR models. We assumed that all the model parameters were specified, and computed the resulting moments of production and queue lengths. In this chapter, we focus on the steady-state optimization of MR models. Here, we use nonlinear programs to find the optimal model parameters. We develop nonlinear programs for a variety of objectives, including:

- Maximize performance (usually, minimize a weighted sum expected waiting times or queue lengths; alternately, maximize throughput) given a network budget.
- Minimize network cost, given performance requirements.
- Minimize production or queue length variances, given performance and / or inventory requirements.

These nonlinear programs use the moment equations developed in previous chapters to define objectives and / or constraints. We develop the nonlinear programs for two families of MR models: those with linear control rules (classes Lxx-MR), and those with general control rules, especially concave rules such as Karmarkar's clearing functions (classes Gxx-MR).

Section 2 discusses a key issue involved in formulating optimization problems, the formulation of station capacities. This issue is particularly relevant to models with linear control rules, since these models do not have capacities *per se*. Instead, stations process a fixed fraction of the work-in-queue, regardless of queue size. Thus, production capacities and capacity constraints must be implied from the station's production moments, which may require simple simulations (an example simulation is presented). With general-control rule models, formulating capacity constraints becomes a matter of fitting nonlinear production functions to station capacities.

Section 3 formulates a variety of optimization problems for models with linear and general control rules. Within linear control rule models, we formulate problems for LLS-MR and LRS-MR models. Within general control rule models, we restrict our attention to GLS-MR models. In general, the optimization problems balance network cost against network performance.

Section 4 discusses solution techniques for the optimization problems. Most of the discussion focuses on the commonly used Method of Moments, which minimizes an augmented Lagrangian function to solve the optimization problems.

Section 5 considers the optimization of LRS-MR models explicitly. We find that LRS-MR optimization problems are more complex than LLS-MR models, since the user may choose one of three control rule types for each station. These choices make the resulting optimization problems combinatorial

as well as nonlinear. To simplify the problems, we present a simple dynamic programming heuristic to determine control rule types.

As noted, most of the nonlinear programs involve a network performance measure, often a weighted sum of expected waiting times. Section 6 shows how to set the weights so that the resulting sum approximately corresponds to a very useful measure: the expected end-to-end completion time for all work flows in a network.

Finally, Section 7 presents two optimization examples. We first consider the problem of minimizing waiting times in an eighteen-station LRS-MR network subject to a budget constraint, and show how the solution changes as the budget constraint increases. We then consider the problem of minimizing cost in a thirteen-station GLS-MR network subject to constraints on the expected waiting times.

2. Formulation of Station Capacities

The nonlinear programs we will consider include terms related to station production capacities, such as capacity constraints. In a maximum performance problem, for example, we install capacity at each station to maximize performance, while keeping the price of the capacity within a budget constraint. Similarly, for a minimum cost problem we economize on capacity purchases while ensuring that the capacity installed meets performance requirements. In this section, we consider techniques to model capacity in MR models. We first consider models with linear control rules, and then consider models with general control rules.

2.1 Capacities for Models with Linear Control Rules

As noted, models with linear control rules do not define station capacities. It is assumed that stations process the required fractions of their work-in-queues, regardless of the size of the queues. Instead, we present an indirect method to model station capacities.

2.1.1 Capacities and Bounded Control Rules

In Chapter 3, we discussed the idea of a bounded control rule, which is:

$$P_i = \min(z_i, \alpha_i Q_i), \tag{1}$$

where z_i is the capacity (or production bound) of the station, and $\alpha_i Q_i$ is the amount that would be produced if the station's capacity was not bounded. This control rule clearly introduces station capacities. Unfortunately, the bounded control rule cannot be analyzed directly.

Instead, we consider an alternate approach: set the capacity level z_i high enough that the station will not have to produce the maximum capacity most of the time. Such a z_i allows us to ignore station capacities when determining production and queue length moments, so that the MR-model moments may be used directly. A common way to do this is to set the capacity equal to the expected production plus some multiple times the standard deviation of production. Thus, capacity becomes:

$$z_i = E(P_i) + k_i\sqrt{\text{var}(P_i)}\,, \tag{2}$$

where k_i is a constant, called the *safety factor*. Once k_i is determined, this formula maps the moments of production into a station capacity, suitable for use in the optimization problems. For example, suppose that the cost of the capacity at all stations must be less than some budget constraint, C. In terms of capacity, this constraint is $\sum_i c_i z_i \le C$, where c_i is the unit price of capacity at station i. Applying equation (2), this constraint becomes:

$$\sum_i c_i \cdot \left(E(P_i) + k_i\sqrt{\text{var}(P_i)}\right) \le C\,, \tag{3}$$

where $\text{E}(P_i)$ and $\text{var}(P_i)$ are the production moments from the MR-model's moment equations.

2.1.2 Interpolating the Safety Factor

The drawback with equation (2) is that we have not specified k_i. If the production quantities were normally distributed, setting k_i to be 2.0 would dictate that the station would produce less than its capacity approximately 97% of the time. However, there is no guarantee that a station's production will be normally distributed. Further, knowing that the station will produce less than its capacity 97% of the time does not imply that the resulting system will behave as if there were no capacity constraint. There may be significant errors between the actual and modeled production and queue length moments, even with $k_i = 2.0$. Alternately, $k_i = 2.0$ may be too conservative; the actual system might be similar to the model if $k_i = 1.5$. Thus, setting $k_i = 2.0$ might require too much capacity.

Setting k_i to be an arbitrary value, such as the common $k_i = 2.0$, is the best we can do without any knowledge of the production distributions. If we do know something about the distributions, however, we can use simulations to interpolate a function for k_i as a function of the MR model results and parameters (namely, $\text{E}(\mathbf{P})$, $\text{var}(\mathbf{P})$, and the lead times). Unlike the $k_i = 2.0$ assignment, the function $k_i(\cdot)$ specifically sets k_i to a value needed to make a station with control rule (1), expected production $\text{E}(P_i)$, production variance $\text{var}(P_i)$, and lead time L_i behave as if it used an uncapacitated linear control rule.

To create the function, we define a measure of similarity between stations using the bounded and linear control rules. Define H_i to be $\text{E}(Q_i)$ given that the station uses a bounded control rule, divided by

$E(Q_i)$ given that the station uses a linear control rule. By definition, $H_i \geq 1$, since a station always produces as least as much with the linear control rule as with the bounded control rule. Thus, an H_i of 1.01 implies that the two expected queue lengths will be within 1% of each other. In our models, H_i is a user-set parameter, and we will make k_i great enough to meet the desired value of H_i. Generally, H_i should be kept fairly close to 1.

Alternately, H_i is the expected waiting time, $E(W_i)$, with a bounded control rule, divided by $E(W_i)$ with a linear control rule. This result follows from Little's Law, which here states that $E(P_i) \cdot E(W_i) = E(Q_i)$, and that $E(P_i)$ is the same whether the bounded or linear control rule is used.

In addition to H_i, we expect that k_i depends on the variance of production at the station, and the lead time at the station. Rather than consider the variance of production directly, we use the coefficient of variation, $s_i = \sqrt{\text{var}(P_i)} / E(P_i)$. This formulation makes k_i invariant to the type of unit measuring production. The lead time, L_i, is the expected waiting time at the station if it used a linear control rule, and is $L_i = 1/\alpha_i$.

Therefore, k_i is specified by a function $k_i(H_i, s_i, L_i)$. The form of the function, however, cannot be known analytically, since we cannot analyze the bounded control rule analytically. We do know that the form will likely be a function of the production distribution. We propose the following simulation technique to interpolate $k_i(H_i, s_i, L_i)$:

1. Create a simulation of a single station using the bounded control rule, such that the work arrivals to the station create the known production distribution.
2. Simulate the station with a variety of k_i's, s_i's, and L_i's. Record the observed H_i for each simulation. (Note that $E(Q_i)$ given a linear control rule is found analytically.)
3. Through regression analysis, find a function of k_i, s_i, and L_i that explains H_i well. Solve this function for k_i; the result will be the interpolated $k_i(H_i, s_i, L_i)$.

Obviously, the results of this process will be inexact. The resulting function comes from a regression of a set of simulations. Further, the technique ignores work arrivals from upstream stations, which may have a different distribution (due to averaging) than arrivals from outside the network. Nonetheless, if applicable, this approach should generate safety factors that are more appropriate than arbitrary assignments.

2.1.3 An Example Interpolation of a Safety-Factor Function

Here, we develop a single-station simulation, design and perform a simulation sequence, and perform regression analysis to interpolate a function $k_i(H_i, s_i, L_i)$.

We first identify a distribution for the work arrivals. Here, we assume that the work arrivals come from a Gamma distribution. In general, if the arrival distribution is not known exactly, but is believed to be "well-behaved", the Gamma distribution is a good choice, for the following reasons.

- It produces only non-negative values, important since we assume that arrivals are non-negative.
- A Gamma distribution is completely defined by its mean and variance, the two statistics estimated for MR-models. Indeed, a Gamma distribution exists for any pair of mean and variance values greater than zero, making it possible to simulate a wide range of mean and variance combinations.
- A sum of gamma distributions is also a Gamma distribution (or, in the case of linear control rules, a multi-period average of Gamma distributions is also a Gamma distribution).
- Finally, Gamma distributions are widely implemented in mathematical and simulation packages.

We assume that the expected arrivals, $E(A_i)$, equals 1000 units for all periods. (As discussed above, the units used do not impact the results.) Thus, $E(P_i)$ also equals 1000 units. The variance of the arrivals varies with the simulation parameters, in accordance with the following formula:

$$\text{var}(A_i) = (2L_i - 1) \cdot (E(P_i) \cdot s_i)^2 . \tag{4}$$

This formula follows from $s_i = \sqrt{\text{var}(P_i)} / E(P_i)$ and $\text{var}(A_i) = (2L - 1) \cdot \text{var}(P_i)$. The latter formula comes from taking the steady-state variance of the production recursion equation for Tactical Planning Models,

$$P_{it} = \left(1 - \frac{1}{L_i}\right) P_{i,t-1} + \frac{1}{L_i} A_{it} , \tag{5}$$

and solving the resulting equation for $\text{var}(A_i)$.

To specify the bounded control rule, we set the capacity of the station, z_i, to be:

$$z_i = E(P_i) + k_i \cdot \sqrt{\text{var}(P_i)} , \tag{6}$$

or, using the definition of s_i,

$$z_i = (1 + k_i s_i) E(P_i) . \tag{7}$$

We next design the simulation runs. The simulations follow a combinatorial design, so that one simulation is performed for each combination of the following values of L_i, s_i, and k_i:

Parameter	**Values simulated**
L_i (Lead time)	{1, 2, 3, 5, 10, 20, 40, 80, 160}
s_i (Standard deviation / expectation)	{0.05, 0.1, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2}
k_i (Safety factor)	{0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0, 2.2, 2.4, 2.6}

The design yields 792 separate simulations. We performed this set of runs in MATLAB. Each simulation ran for 8,000 periods.

We next perform regression analyses on the simulation data. Our goal is to find a function $k_i(H_i, s_i, L_i)$, which generates the desired safety factor k_i as a function of H_i, s_i, and L_i. We do so in several steps. First, we find a function that explains H_i as a function of k_i, s_i, and L_i. This function, $H_i(k_i, s_i, L_i)$ is solved for k_i, yielding the form of the function $k_i(H_i, s_i, L_i)$. We then re-estimate the parameters of $k_i(H_i, s_i, L_i)$ through a separate regression analysis. (Simply solving $H_i(k_i, s_i, L_i)$ for k_i, as is, can result in inaccurate estimates for k_i.)

We note several desirable properties for $k_i(H_i, s_i, L_i)$.

- The function arises from a regression equation that fits the data well (high adjusted R^2 statistic).
- The function produces "reasonable" k-values for most values of s. Ideally, k-values estimated by the function should increase with s. It also means that k-values estimated by the function should not get overly large, head towards infinity, or become negative as s varies. Instead, k-values estimated by the function approach zero as s approaches zero, and the rate of increase in k-values should become small as s gets large.
- The function can be written in closed form, allowing it to be substituted into nonlinear programs.

To get insight into the relationships between H_i, k_i, s_i, and L_i, we present several scatter diagrams of the simulation results. Figure 20 relates H_i to the safety factor, k_i. The diagram implies that H_i is inversely proportional to k_i.

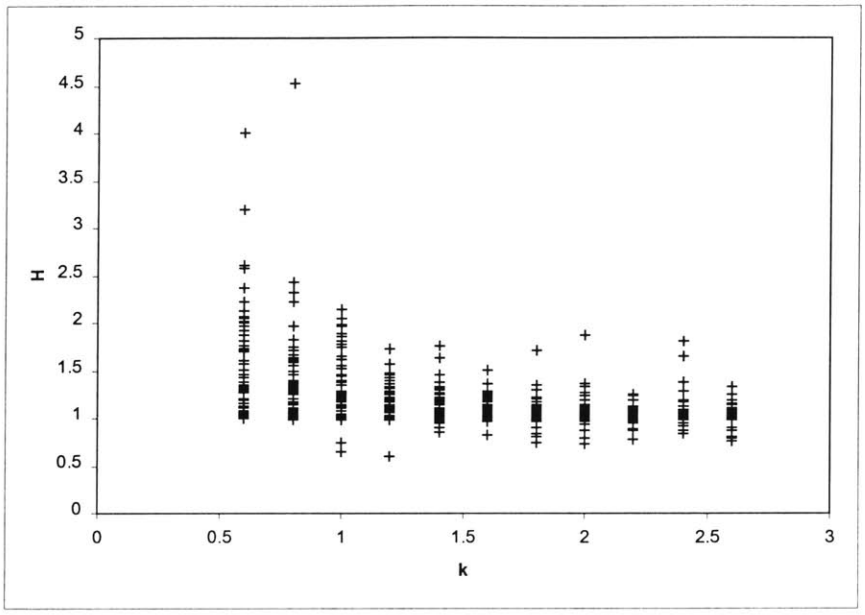


Figure 20 -- H_i v. Safety Factor

Figure 21 relates H_i to s_i, the "normalized" standard deviation. The diagram implies that H_i is directly related to s_i in some way. It is difficult to tell the exact nature of the relationship by looking at the graph, other than that it appears to be nonlinear. (Interestingly, one of the best fits is a negative linear relationship between the inverse of H_i and s_i.)

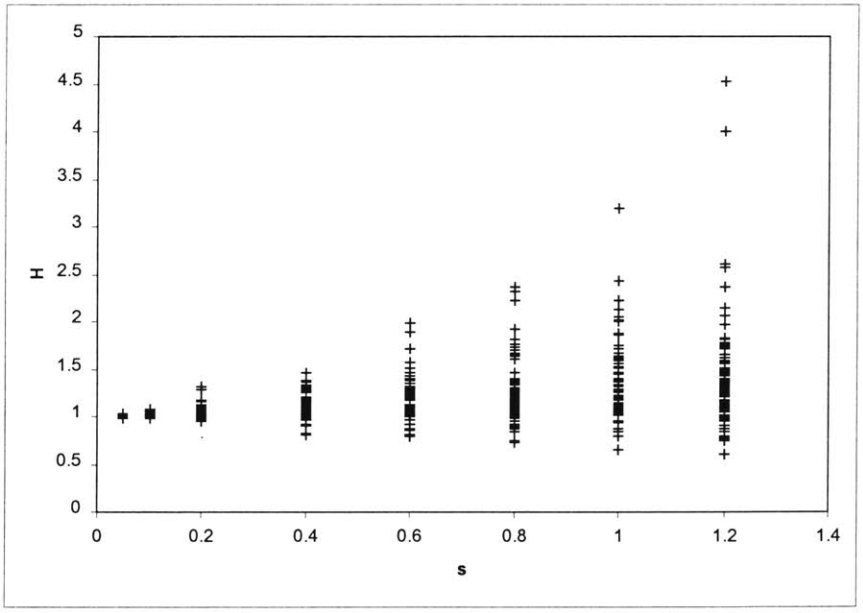


Figure 21 -- H_i v. Production Standard Deviation

Finally, Figure 22 relates H_i to L_i, the lead time at the station. At first glance, there appears to be a direct relationship between H_i and L_i. However, the relationship is an illusion caused by a few outliers

on the right side of the graph. The relationship between the mean of H_i and L_i actually is not statistically significant. (In some simulation runs used for debugging, outliers appeared on the left side of the graph.)

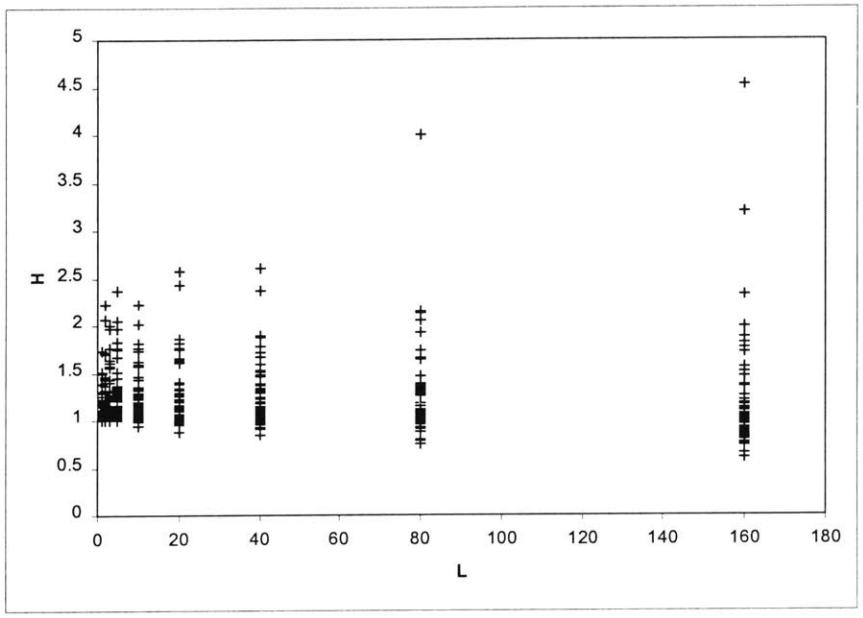


Figure 22 -- H_i v. Lead Time

Using these plots as a guide, we performed a large number of regression analyses on the simulation data. These regressions tested relationships between H_i and k_i, L_i, and s_i. As it turned out, the class of functions that best explained H_i all had the following form: all related $1/H_i$ against s_i and a function of s_i / k_i. In general, L_i was not a significant factor in explaining $1/H_i$.

The table below summarizes the regression results for this class of functions. The table also shows the regression result for the one function found in which L_i was significant. (To simplify the notation, the i-subscripts are not shown).

Regression Equation	**Adjusted R^2 Statistic**
$1/H \sim s + \sqrt{sk}$	0.5380
$1/H \sim s + sk$	0.5541
$1/H \sim s + s^2 k$	0.4672
$1/H \sim s + s/\sqrt{k}$	0.6041
$1/H \sim s + s/k$	0.6053
$1/H \sim s + s/k^2$	0.5852
$1/H \sim s + \sqrt{s}/k$	0.5890
$1/H \sim s + s^2/k$	0.5261
$1/H \sim s + \sqrt{s/k}$	0.5654
$1/H \sim L + s + s/k$	0.6151 (L-term significant; less than .0004)

All of these functions explain $1/H_i$ fairly well, given the noise of the data. All other functions considered had far lower R^2 statistics ($R^2 < 0.4$).

We consider the three functions with $R^2 > 0.6$, all of which might lead to suitable functions for k_i. Of these three functions, the function $1/H_i \sim s_i + s_i/k_i$ has some usability advantages over the other two. It is a function of only two variables, not three, which simplifies its use in comparison to $1/H_i \sim L_i + s_i + s_i/k_i$, especially given that the effect of L_i in the latter function is extremely weak. Further, solving it for k_i gives rise to a comparatively simple function for k_i, as opposed to $1/H \sim s + s/\sqrt{k}$.

The regression plot of the fitted function $1/H_i \sim s_i + s_i/k_i$ is:

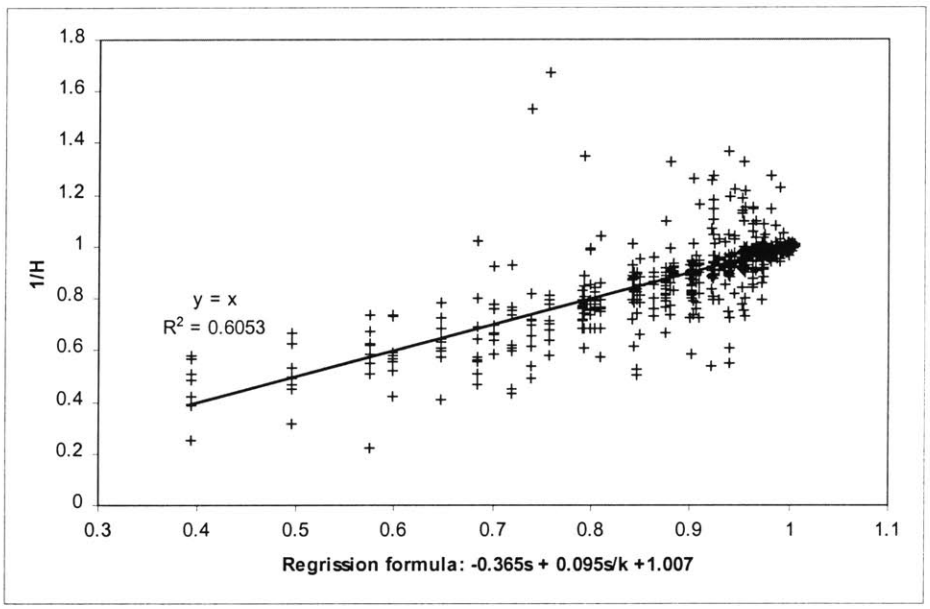


Figure 23 -- Regression Plot of the Interpolated $1/H_i$ Function

With the exception of the three outliers at the top of the graph, this plot shows a clearly linear relationship, meaning that the fitted function has a form close to the true relationship between H_i and the other variables. The regression plots for the other two functions with $R^2 > 0.6$ look virtually identical to the above graph. Consequently, we may choose from any of the three functions, and we select the function $1/H_i \sim s_i + s_i/k_i$ for the reasons mentioned above.

Solving $1/H_i \sim s_i + s_i/k_i$ for k_i yields a nonlinear function for k_i that is difficult to regress; inverting the resulting function yields:

$$\frac{1}{k_i} \sim \frac{1}{H_i s_i} + \frac{1}{s_i}, \tag{8}$$

which is a linear regression in terms of $1/k_i$ and the other variables. Fitting this function yields an R^2 of 0.5487, a reasonably good fit. As with our exploration of the class of functions for $1/H_i$, however, we consider a number of functions for $1/k_i$, similar to (8), to determine if we can find a better fitting function. As it turns out, fitting the function:

$$\frac{1}{k_i} \sim \frac{1}{s_i\sqrt{H_i}} + \frac{1}{s_i}, \tag{9}$$

yields a higher R^2 coefficient of 0.5625. The regression graph of (9) is:

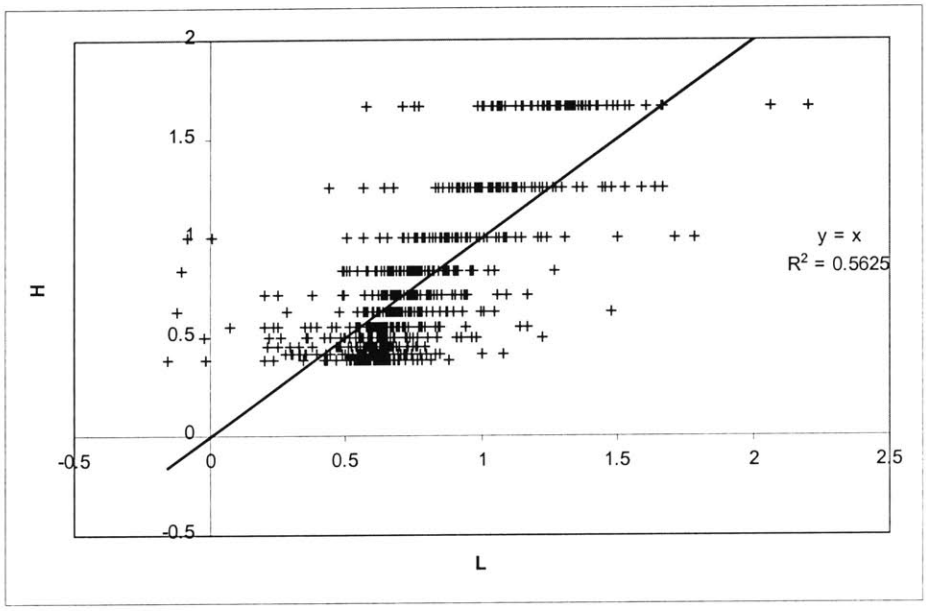


Figure 24 -- Regression Plot of the Interpolated $1 / k_i$ Function

which clearly shows a linear relationship. The regression graph of (8) showed a more pronounced curvature, implying that (9) does a better job capturing the true relationship between $1/k_i$ and the other variables. Thus, we use (9) to create our function for the safety factor k_i. The fitted function is:

$$k_i(H_i, s_i, L_i) = \left(-\frac{2.5881}{s_i\sqrt{H_i}} + \frac{2.5935}{s_i} + 0.5233 \right)^{-1} \tag{10}$$

Figure 25 uses (10) to plot the safety factor k_i against different values of s_i, parameterized by H_i.

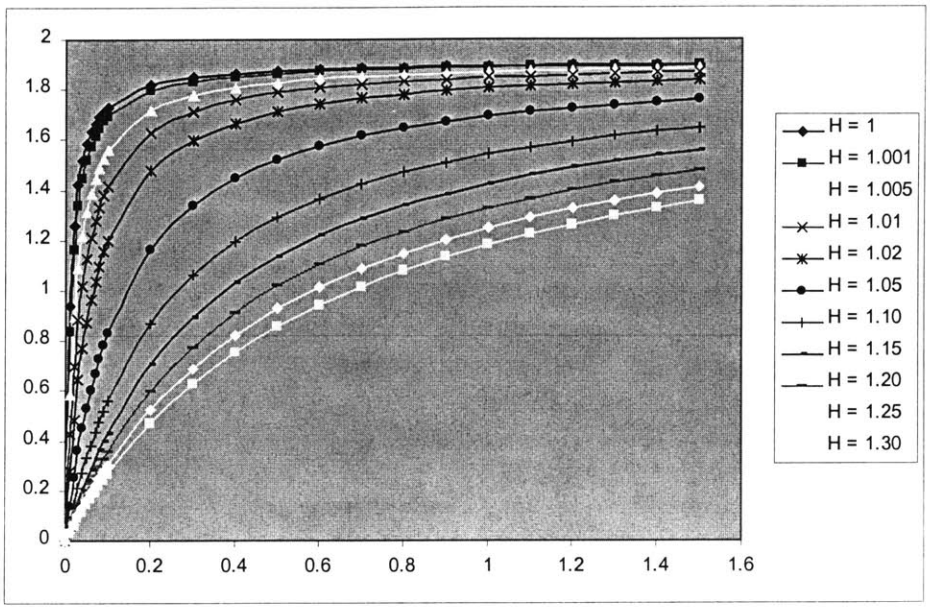


Figure 25 -- Safety Factors as a Function of s_i and H_i

The graph implies several facts about our interpolated $k_i(H_i, s_i, L_i)$. First, the function satisfies all of the desired properties for a safety-factor function. It has been written in closed form. It generates a reasonable value of k_i for all positive values of s_i. It approaches 0 as s_i approaches 0, and its rate of increase slows as s_i increases. In this case, k_i approaches the following limit as s_i increases:

$$\lim_{s\to\infty} k_i(H_i, s_i, L_i) = \lim_{s\to\infty}\left(-\frac{2.5881}{s_i\sqrt{H_i}} + \frac{2.5935}{s_i} + 0.5233\right)^{-1} = \frac{1}{0.5233} = 1.911. \qquad (11)$$

Second, the graph demonstrates the tradeoffs between H_i and k_i. We see that by allowing fairly small tolerances in H_i (say, allowing the simulated waiting time to be 10% greater than the lead time) we substantially reduce the safety factor and the resulting capacity installed at a given station. Further, the gains in k_i decrease as H_i increased; for example, the gains resulting from increasing H_i from 1.10 to 1.20 are not as significant as increasing H_i from 1.05 to 1.10. These tradeoffs suggest the following:

- The H_i parameters may be decision variables in the optimization problems. In particular, for optimization problems where "performance" is measured by expected waiting times, we can vary H_i to find a minimum k_i that yields a desired expected waiting time.
- However, in practice we should limit how much we increase any particular H_i. The simulations calculated expected waiting times at a single station, not for an entire network. We expect that having high H_i parameters – which cause stations to hit their capacity bounds frequently – may cause significant deviations in the downstream work flows. Consequently, having H_i values significantly

greater than one may cause network-wide effects that will result in the inaccuracy of the analytic results.

- In addition, we may wish to place lower bounds on k_i, regardless of the values implied by $k_i(H_i, s_i, L_i)$. The lowest safety factor tested in the simulation was 0.6; in other simulations, lower values of k_i produced very long waiting times.

Finally, the graph shows that the initial approximation of simply setting $k_i = 2.0$ had significant value, at least in the case where work arrivals come from a Gamma distribution. If we set the H_i values close to one (i.e., we want the capacitated network to behave similarly to the uncapacitated network), we note that the k_i values rapidly approach the upper bound of 1.911 as s_i increases. Indeed, the graph implies that if we know that s_i is likely to be at least 0.2, there is little need to substitute $k_i(H_i, s_i, L_i)$ into the nonlinear programs. We can simply set $k_i = 1.9$ for all stations in the network, significantly simplifying the resulting optimization problems.

2.2 Capacities for Models with General Control Rules

Capacity modeling for stations with general control rules is dependent on the form of the control rule. Often, we use bounded concave control rules. At first glance, the bound suggests a natural capacity. For example, clearing functions (discussed at length in Chapter 3) have the form:

$$P_i = \frac{M_i Q_i}{Q_i + \beta_i}, \tag{12}$$

where M_i and β_i are parameters, with M_i being an asymptotic bound on production. Obviously, M_i might be used as the "capacity" of the station.

However, it is not the case that a station's capacity should automatically become the asymptotic bound. Instead, the user should take some care to match the rated capacity of the station, which generates costs in the optimization problem (for example, the rated capacity of a machine), with the actual production behavior of the station, represented by the production function.

For example, suppose we model the behavior of a machine with a clearing function, and suppose the machine has some rated capacity, y_i. It may be the case that y_i is a true production bound, in which case $M_i = y_i$. However, the machine's rated capacity might exceed its true capacity, in which case the true production bound, M_i, will be lower than y_i. In an entirely different scenario, y_i may be a rated capacity, but the users find that the capacity may be exceeded in overtime scenarios. In this case, $M_i > y_i$. In practice the analyst should calculate M_i by analyzing the observed behavior of the station, and mapping the rated capacity to the actual production bound.

Note that most production functions have parameters besides the asymptotic bound; these parameters usually determine how quickly production rises to meet the asymptotic bound. In the case of clearing functions, for instance, this parameter is β_i. In Chapter 3, we usually assumed that $\beta_i = M_i$, so that the rate of production equaled one as the queue length approached zero. However, there is no reason why β_i must be one; it may be any positive value. In practice, β_i should be set by analyzing the observed behavior of the station.

Thus, in general, our goal is to find functions that map the rated capacity of a station (which entails costs) to the parameters of the production function. With a clearing function, for instance, we find *translating functions* f_M and f_β such that we can write the production function as a function of the rated capacity, y_i:

$$P_i = \frac{f_M(y_i) \cdot Q_i}{Q_i + f_\beta(y_i)}. \tag{13}$$

This form of the production function may then be used in optimization problems that involve capacity cost and network performance tradeoffs.

Translating functions are dependent on the production function and the structure of the rated capacities. In general, the functions will be interpolated from statistical data.

3. General Forms of Optimization Problems

In this section, we discuss general forms of optimization problems of MR-models. For the sake of simplicity, we assume that the control rules at all stations are single-queue control rules. Derivations of nonlinear programs for multi-queue control rules are similar, with the addition of notation to deal with multi-queue rules. (For example, with single-queue linear control rules, **D** is a diagonal matrix of smoothing parameters; with multi-queue linear control rules, **D** is a nondiagonal matrix.)

3.1 Definitions

Terminology. The following are terms and symbols that will be used in the mathematical programs.

- $\mathbf{F(x, y)}$ is a vector (or matrix) valued function with vector (or matrix) valued arguments **x** and **y**. $F_j(\mathbf{x},\mathbf{y})$ is the jth element of $\mathbf{F(x, y)}$ if the function is vector-valued. Similarly, $F_{ij}(\mathbf{x},\mathbf{y})$ is the (i,j) element of $\mathbf{F(x, y)}$ if the function is matrix-valued.
- The expression $\mathbf{A} \bullet \mathbf{B}$ represents the array product of matrices (or vectors) **A** and **B**. $\mathbf{A} \bullet \mathbf{B}$ produces a matrix whose (i,j) entry is $A_{ij}B_{ij}$.
- The expression $\mathbf{A} \cdot \mathbf{B}$ or $\mathbf{AB}$ represents ordinary matrix or vector multiplication.

- **e** is a vector of ones.
- diag(**A**) is a function that produces a vector whose entries are the diagonal entries of **A**.

Decision variables. The following variables will be decision variables in some of the nonlinear programs.

- **z** is a vector whose *j*th entry is the capacity installed at station *j*.
- μ is a vector whose *j*th entry represents the expected workload arriving at station *j* from outside the network.
- (Lxx-MR models only) **L** is a vector whose *j*th entry is the lead time at station *j*. (Recall that in Lxx-MR models, $L_j = 1/\alpha_j$, where $0 < \alpha_j \leq 1$ is a smoothing parameter.) Associated with the matrix **D**, a diagonal matrix with the smoothing parameters on the diagonal, so that $D_{jj} = \alpha_j = 1/L_j$.
- (LRS-MR models only) **r** is a vector whose *j*th entry represents the control rule used at station *j* (instruction-control, job-control, or double-control).
- (LRS-MR models only) $\mathbf{R_E}$ is a vector whose jth entry represents the expected number of job requests entering station *j* from outside the network. This variable is used in place of μ for LRS-MR models.

Fixed variables. The following variables are fixed inputs to the nonlinear programs.

- **w** is a vector whose *j*th entry is a weight that measures the significance of the expected waiting time or queue length at station *j*, or the significance of the expected input at a station (depending on the nonlinear program).
- (Lxx-MR models only) **H** is a vector whose *j*th entry is the similarity measure used to model bounded control rules, discussed in Section 2.1. Recall that H_j is the expected queue length of station *j*, using a bounded control rule, divided by the expected queue length of station *j*, using an unbounded linear control rule. (In Section 2.1, we discussed the possibility of using **H** as a decision variable; however, we will assume that **H** is fixed at a value close to a vector of ones for the duration of this paper.)

Implied variables. The MR models have a number of other inputs not listed above. However, these inputs define the model of the shop, and remain constant across all nonlinear programs using that model. Thus, we do not show these inputs in the nonlinear programs. Examples of these implied inputs include input covariance matrices and workflow matrices. For GLS-MR models, implied inputs also include the control rules, and the transition functions mapping capacity vector **z** to control rule parameters. For LRS-MR models, the implied inputs include the expectation and variance of the jobs per request at each

station, and the expectation and variance of the instructions per job at each station (as discussed in Chapter 5).

We may evaluate models in which the input covariance matrix is a function of the expected arrivals (i.e., we associate greater expected arrivals with a greater variance in those arrivals). In this case, the covariance matrices become functions of μ. Nonetheless, the input covariance matrices need not be specified in the nonlinear programs, since μ is already listed as a decision variable.

Functions. The following functions will be used in the mathematical programs.

- Function **c(z)** calculates the cost of building a network, given a vector of station capacities.

- Function $\mathbf{P_E(\mu)}$ produces a vector of expected workloads given a vector of expected arrivals. (For LRS-MR networks, this function becomes $\mathbf{P_E(R_E)}$.)

- Function $\mathbf{S_P()}$ produces a steady-state covariance matrix of the production quantities. Its argument list depends on the model class. For models with linear control rules, $\mathbf{S_P}$ is a function of the lead times and expected arrivals (assuming the input covariances depend on the expected arrivals); we have $\mathbf{S_P(L, \mu)}$. For LRS-MR models, $\mathbf{S_P}$ is a function of the expected requests and the control rule choices, as well; we have $\mathbf{S_P(L, R_E, r)}$. For GLS-MR models, $\mathbf{S_P}$ is a function of the expected arrivals and the station capacities; we have $\mathbf{S_P(z, \mu)}$.

- (GLS-MR models) Function $\mathbf{Q_E(\mu, z)}$ produces a vector of estimated expected queue lengths given a vector of expected arrivals and a vector of station capacities. This vector may come from either the MomentsP or MomentsQ algorithm (discussed in Chapter 3). Note that for linear control rule models, we simply write the expected queue lengths to be $\mathbf{D^{-1}P_E(\mu)}$.

- (GLS-MR models) Function $\mathbf{S_Q(z, \mu)}$ produces an estimate of the steady-state covariance matrix of the queue length given a vector of capacities and a vector of expected arrivals. The matrix may come from either the MomentsP or MomentsQ algorithm.

- (Lxx-MR models) Function $\mathbf{k(H, L, \mu)}$ computes a vector of safety factors, as discussed in Section 2.1. It is a function of the **H** vector, the lead times, and the production variances calculated by $\mathbf{S_P()}$ (which, in turn, may be functions of μ). As discussed in 2.1, we require the capacity at each station

to be the expected production plus k_j times the standard deviation of production at station j. Note that for LRS-MR models, $\mathbf{k}$ is also a function of the control rule choices, $\mathbf{r}$, as well.

3.2 Optimization Problem Formulations for Models with Linear Control Rules

We use the above functions and variables to define a series of general mathematical programs for models with linear control rules. For each program, we list the constraints along with a description of what the constraints mean. The following programs are for LLS-MR models; the same programs apply to LRS-MR models by changing μ to $\mathbf{R_E}$ and by changing the argument lists of the $\mathbf{S_P}()$ and $\mathbf{K}()$ functions to include $\mathbf{r}$, the vector of control rule choices.

3.2.1 Maximum Performance Given Throughput and Budget

Constraint	Description
$\min \mathbf{w}'\cdot(\mathbf{H} \bullet \mathbf{L})$	Minimize a weighted sum of the expected waiting times at all of the stations.
subject to	
$\mathbf{P_E}(\boldsymbol{\mu}) + \mathbf{k}(\mathbf{H},\mathbf{L},\boldsymbol{\mu}) \bullet \sqrt{\text{diag}(\mathbf{S_P}(\mathbf{L},\boldsymbol{\mu}))} \le \mathbf{z}$	Capacity required to meet the current performance level is less than the capacity currently installed, for all stations.
$\mathbf{c}(\mathbf{z}) \le C$	Total cost of building a network with the specified capacities is less than the total budget, C.
$\boldsymbol{\mu} \ge \boldsymbol{\mu}_{\min}$	Expected inputs to the network (throughput) are greater than a set of minimums.
$\mathbf{L} \ge \mathbf{e}$	The lead times are all greater than one.
r_j = {instruction, job, double}, all j *(LRS-MR models only)*	The control rule at each station is set to be one of the three possible control rules (instruction- control, job-control, or double-control).

This nonlinear program seeks to minimize waiting times. It is similar to a nonlinear program that seeks to minimize a weighted sum of expected queue lengths (i.e. "minimize expected inventory"). The only change that would need to be made is to the objective function. The new objective function is $\min \mathbf{w}'\cdot\left[H \bullet \left(D^{-1} \cdot P_E(\mu)\right)\right]$.

3.2.2 Maximum Throughput Given Performance Requirements and Budget

Constraint	Description
$\max \mathbf{w}'\cdot\boldsymbol{\mu}$	Maximize a weighted sum of the expected arrivals at

all of the stations.

subject to

$\mathbf{P_E(\mu) + k(H,L,\mu) \bullet \sqrt{diag(S_P(L,\mu))} \le z}$ — Capacity required to meet the current performance level is less than the capacity currently installed, for all stations.

$\mathbf{c(z)} \le C$ — Total cost of building a network with the specified capacities and control rules is less than the total budget, C.

$\mathbf{w' \cdot (H \bullet L)} \le T$ — A weighted sum of the expected waiting times is less than some constant, T.

$\mathbf{L \ge e}$ — The lead times are all greater than one.

$\mu_{j,\min} \le \mu_j \le \mu_{j,\max}, \forall j$ — Impose expected arrival bounds as needed. (For example, if new arrivals only appear at one station, μ_j for all the other stations equals 0.)

r_j = {instruction, job, double}, all j *(LRS-MR models only)* — The control rule at each station is set to be one of the three possible control rules (instruction- control, job-control, or double-control).

3.2.3 Minimum Budget Given Performance Requirements and Throughput

Constraint	Description
$\min \mathbf{c(z)}$	Minimize the total cost of the network.

Subject to

$\mathbf{P_E(\mu) + k(H,L,\mu) \bullet \sqrt{diag(S_P(L,\mu))} \le z}$	Capacity required to meet the current performance level is less than the capacity currently installed, for all stations.
$\mathbf{\mu \ge \mu_{min}}$	Expected inputs to the network (throughput) are greater than a set of minimums.
$\mathbf{w' \cdot (H \bullet L)} \le T$	A weighted sum of the expected waiting times is less than some constant, T.
$\mathbf{L \ge e}$	The lead times are all greater than one.
r_j = {instruction, job, double}, all j *(LRS-MR models only)*	The control rule at each station is set to be one of the three possible control rules (instruction- control, job-control, or double-control).

3.3 Optimization Problem Formulations for Models with General Control Rules

Here, we define a series of general mathematical programs for models with general control rules (class GLS-MR). For each program, we list the constraints along with a description of what the constraints mean.

3.3.1 Maximum Performance Given Throughput and Budget

Constraint	Description
$\min \sum_j \frac{w_j \cdot Q_{E,j}(\boldsymbol{\mu},\mathbf{z})}{P_{E,j}(\boldsymbol{\mu})}$	Minimize a weighted sum of the expected waiting times at all of the stations. (Here, we use the fact that the expected waiting time is $E(Q_j) / E(P_j)$.)
subject to	
$\mathbf{c}(\mathbf{z}) \le C$	Total cost of building a network with the specified capacities is less than the total budget, C.
$\boldsymbol{\mu} \ge \boldsymbol{\mu}_{\min}$	Expected inputs to the network (throughput) are greater than a set of minimums.

This program has a simpler form than the equivalent program for linear control rules, since the functions mapping $\mathbf{z}$ to control rule parameters are be part of function $\mathbf{Q_E}(\boldsymbol{\mu}, \mathbf{z})$.

The form of this program is similar to a nonlinear program to minimize a weighted sum of expected queue lengths (i.e. minimize weighted inventory). The only change is to the objective function, which becomes $\min \mathbf{w}' \cdot \mathbf{Q}_E(\boldsymbol{\mu},\mathbf{z})$.

3.3.2 Maximum Throughput Given Performance Requirements and Budget

Constraint	Description
$\max \mathbf{w}' \cdot \boldsymbol{\mu}$	Maximize a weighted sum of the expected arrivals at all of the stations.
subject to	
$\mathbf{c}(\mathbf{z}) \le C$	Total cost of building a network with the specified capacities and control rules is less than the total budget, C.
$\sum_j \frac{w_j \cdot Q_{E,j}(\boldsymbol{\mu},\mathbf{z})}{P_{E,j}(\boldsymbol{\mu})} \le T$	A weighted sum of the expected waiting times is less than some constant, T.
$\mu_{j,\min} \le \mu_j \le \mu_{j,\max}, \forall j$	Impose expected arrival bounds as needed. For example, if new arrivals only appear at one station, μ_j for all the other stations equals 0.)

3.3.3 Minimum Budget Given Performance Requirements and Throughput

Constraint	Description
$\min \mathbf{c(z)}$	Minimize the total cost of the network.
Subject to	
$\boldsymbol{\mu} \geq \boldsymbol{\mu}_{\min}$	Expected inputs to the network (throughput) are greater than a set of minimums.
$\sum_j \frac{w_j \cdot Q_{E,j}(\boldsymbol{\mu}, \mathbf{z})}{P_{E,j}(\boldsymbol{\mu})} \leq T$	A weighted sum of the expected waiting times is less than some constant, T.

4. Nonlinear Programming Techniques

The nonlinear programs discussed in Section 3 can all be expressed in the following general form:

$$\begin{aligned} &\text{minimize} \quad f(x) \\ &\text{subject to} \quad g(x) \leq 0, \ x \in X \end{aligned} \tag{14}$$

In the above program, x is a vector containing all the variables being solved for in the model, and $f(x)$ is the objective function (which depends on the type of program being solved). The term $g(x) \leq 0$ is the set of all the *computationally hard* inequality constraints. If we cannot optimize an objective function over a constraint easily, the constraint is "hard". The term $x \in X$ means that all the variables must be within a set of *computationally easy* constraints. Here, "easy" means it takes roughly the same amount of work to optimize the objective function with or without the constraints; for example, simple upper- and lower-bound constraints are considered to be computationally easy. In our nonlinear programs, the following constraints are computationally hard:

- The constraints requiring the capacity to be sufficient to meet performance requirements.
- The constraints requiring the capacity costs to be less than some amount (budget constraints).

The other constraints are upper- and lower-bound constraints, which are computationally easy. These constraints include upper and lower bounds on the vector of expected arrivals, and lower bounds on the lead time vectors (for linear control-rule models).

Generally, for the programs in Section 3, we can rewrite the computationally hard inequality constraints to be equality constraints. This is beneficial, since the nonlinear programming techniques we will discuss are simpler to use with equality-constrained problems. We usually use production functions

such that station performance (measured by expected queue length or waiting time) strictly improves as the amount of capacity increases, and cost functions such that costs strictly increase with the installed capacity. These two assumptions imply the following rules:

- If there is a performance constraint, the installed capacity should always equal the capacity required to meet a performance level. Otherwise, one could reduce cost by reducing capacity at some of the stations.
- If there is a budget constraint, the cost of the installed capacity should equal the constraint. Otherwise, if the capacity is less than the budget constraint, one could improve performance or throughput by increasing capacity.

These two rules imply that the computationally hard inequality constraints can be rewritten as equality constraints without affecting the optimality of the solution.

These arguments assume that each nonlinear program has at most one performance constraint and one budget constraint (like those in Section 3). The same arguments apply to programs with multiple performance and budget constraints as well, provided that each individual station is not affected by more than one performance constraint or budget constraint. Otherwise, if a station appears in multiple budget constraints say, it usually will satisfy one of the constraints with equality, with the others being not binding.

Consequently, we will rewrite the nonlinear programs to have the form:

$$\begin{aligned} &\text{minimize} \quad f(x) \\ &\text{subject to} \quad h(x) = 0,\ x \in X, \end{aligned} \tag{15}$$

where:

- $f(x)$ is the objective function, which is either maximize throughput, maximize performance, or minimize network cost;
- $h(x)$ is the set of performance-capacity and budget constraints, and
- X represents the set of the remaining constraints, which usually are bounds on the lead times and expected inputs to the network.

To solve these programs, we use a common technique, the *Method of Multipliers*, as follows:

1. Form the *augmented Lagrangian* function, $L_c(x,\lambda) = f(x) + \lambda' h(x) + \frac{c}{2}\|h(x)\|^2$. Here, λ is a vector of Lagrange multipliers. The $\|h(x)\|^2$ term calculates the sum of the squared penalty violations, and c is a positive penalty parameter that determines the "cost" of the sum of the squared penalty violations.
2. Optimize the following nonlinear program: $\min\{L_c(x,\lambda) | x \in X\}$.

3. Update the Lagrange multiplier to be $\lambda = \lambda + c \cdot h(x)$. Increase c by some amount (usually, c is multiplied by a factor of 5 to 10).
4. Repeat steps 1 to 3 until the x-vector converges to desired limits.

It can be shown that the Method of Moments converges to a local minimum. Further, the solution found is optimal if the constraints are convex, and if the objective function is convex (c.f. Bertsekas, 1995c).

In the example nonlinear programs in Section 6, we will use linear cost functions and linear or concave control rules. Concave control rules imply convex inverse production functions, meaning that the expressions for the expected queue lengths will also be convex functions. Thus, the solutions to the examples will be globally optimal.

To solve each of the augmented Lagrangian functions, we use the technique of Gradient Projection with Diagonal Scaling and the Armijo Rule. (All techniques c.f. Bertsekas, 1995c) The technique works as follows:

1. *Diagonally-scaled gradient calculation.* We begin with a vector of current variables x_k that satisfies $x \in X$, but need not satisfy $h(x) = 0$ exactly. We calculate the gradient of $L_c(x_k, \lambda)$ with respect to x_k and λ, yielding $\nabla L_c(x_k, \lambda)$. We also compute the principal second derivatives of $L_c(x_k, \lambda)$. (A principal second derivative has the form $\partial^2 f(x,y)/dx^2$.) We then calculate the descent direction $-H_k \nabla L_c(x_k, \lambda)$, where H_k is a diagonal matrix with entries $H_{k,ii} = (\partial^2 L_c(x_k, \lambda)/\partial x_i^2)^{-1}$. Sometimes, this calculated direction is not a descent direction; in that case, the feasible direction is just the negative gradient, $-\nabla L_c(x_k, \lambda)$.

2. *Projection with Armijo Rule.* The next vector of current variables, x_{k+1}, is $x_{k+1} = [x_k - s_k H_k \nabla L_c(x_k, \lambda)]^+$. Here, s_k is a positive scalar, and the operator $[\cdot]^+$ represents the projection of $\bar{x}_k = x_k - s_k H_k \nabla L_c(x_k, \lambda)$ onto the lower- and upper-bound constraints. The projection is simple: if a component of $\bar{x}_k$ exceeds one of the bounds, that component is set to be the violated bound. Other components of $\bar{x}_k$ are left as they are. We choose s_k so that $L_c(x_{k+1}, \lambda)$ will be sufficiently smaller than $L_c(x_k, \lambda)$ to guarantee that the sequence of x_k's will converge to a local minimum. The Armijo Rule provides this guarantee if the following condition holds:

$$L_c(x_k, \lambda) - L_c(x_k(B^m), \lambda) \geq \sigma \nabla_x L_c(x_k, \lambda)'(x_k - x_k(B^m)),$$

where $x_k(B^m) = x_k - B^m H_k \nabla L_c(x_k, \lambda)$, B is a scalar fixed between zero and one, m is a nonnegative integer, and σ is a small constant greater than zero (usually set to be less than 0.1). We set s_k to be the largest value of B^m such that the Armijo condition holds.

3. *Iteration.* We set $x_{k+1} = [x_k - s_k H_k \nabla L_c(x_k, \lambda)]^+$, and repeat step one. We repeat the iteration until the x_k vectors converge.

The biggest bottleneck in the nonlinear optimization is the gradient calculation. The augmented Lagrangian is a function of the "difficult" constraints in the network problems, notably the budget constraint and the performance-capacity constraints. The performance-capacity constraints are functions of $\mathbf{S_P()}$, the covariance matrix of production. Recall that the covariance matrix of production is calculated via a power series of matrices, for models with general or linear control rules. Consequently, it is difficult to find the partial derivatives of the augmented Lagrangian directly.

Thus, we may approximate the gradient and second partial derivatives using difference formulas. The first derivatives of the gradient are approximated by *the central difference formula*:

$$\frac{\partial f(x_i)}{\partial x_i} \approx \frac{1}{2h}\left(L_c(x + he_i) - L_c(x - he_i)\right). \tag{16}$$

In this expression, h is a small positive scalar, and e_i is a vector whose ith element is one, and whose other elements are zero. The error of this estimate is $O(h^2)$, which is quite good. "Practical experience" (Bertsekas, 1995) suggests that h be chosen to balance the error from the approximation against the computer's roundoff error. For a machine using 16-digit precision, an h of about 10^{-5} will be used.

The second derivatives of the augmented Lagrangian are calculated as follows:

$$\frac{\partial^2 f(x_i)}{(\partial x_i)^2} \approx \frac{1}{h^2}\left(L_c(x + he_i) + L_c(x - he_i) - 2L_c(x)\right). \tag{17}$$

This formula is less accurate than the formula of the first derivative, but practical experience suggests that the second derivatives do not need to be computed extremely accurately to get good convergence performance.

The two formulas require two evaluations of the augmented Lagrangian (which requires computing two covariance matrices) for each first and second partial derivative. This means every iteration of the nonlinear optimization algorithm requires computing $O(n)$ covariance matrices. This is a significant bottleneck, which may hinder our ability to optimize large networks.

Consequently, if one needs to solve a nonlinear program for a particular large model frequently, one may be better off trying to calculate the derivatives of the augmented Lagrangian exactly (using the program MAPLE, for instance). For smaller models that will not be solved often, however, calculating the derivatives often will not be worth the effort.

5. Optimization of LRS-MR Models

For the most part, optimization of LRS-MR models is similar to the optimization of LLS-MR models. The LRS-MR model equations require more fixed inputs (expectations and variances of jobs per request at each station, and expectations and variances of instructions per job at each station), but the moment calculations given the inputs are similar to those of LLS-MR models. Thus, the constraints and objective functions are similar for LRS-MR and LLS-MR models. Indeed, we optimize an LRS-MR model in Section 7.

However, LRS-MR models do have one significant complication: the choice of control rule at each station (instruction-, job- or double-control; see Chapter 5 for more details). The choice affects the production variance calculations, which in turn affect the capacity calculations needed in the nonlinear programs. (Recall that Lxx-MR models use var(**P**) to estimate the capacity needed to meet expected queue length requirements.) The choice of control rules makes LRS-MR optimization problems combinatorial as well as nonlinear. Solving a nonlinear combinatorial problem to optimality usually is an intractable problem.

Consequently, we use a structured approach to solve LRS-MR optimization problems approximately. First, we use a heuristic to assign control rules to the stations. Then, we solve the resulting nonlinear program given fixed control rule assignments.

A conventional heuristic approach would be to solve some sort of program which allowed "fractional" control rules, and then used a rounding scheme to convert the fractional results to integers representing actual rules. Unfortunately, this approach does not apply to the LRS-MR model; the variance functions depending on the control rules are not defined for "fractional" rules. Therefore, we design a heuristic using a different approach.

As discussed in Chapter 5, our preferred choice would be to use "double control rules" which process fixed fractions of jobs and instructions simultaneously. However, double-control rules usually do not exist in practice, leaving us with instruction-control and job-control rules. We recall the following properties of the control rules in creating a heuristic:

- In comparison with the job-control rule, the instruction-control rule decreases the production variance of a station, but increases the variances of downstream stations. The instruction-control rule smoothes the instructions processed by a station, but not the flow of jobs.
- In comparison with the instruction-control rule, the job-control rule increases the production variance of a station, but decreases the variances of the downstream stations. The job-control rule smoothes the flow of jobs out of the station, but not the instructions processed by the station.

These two properties suggest a dynamic programming heuristic for networks that are acyclic graphs. (An acyclic graph is a network with no feedback loops.) Let the estimated required capacity as a function of var(**P**) be the vector-valued function **Z**(var(**P**)). Consider the following heuristic:

Heuristic AssignRules

1. Sort the stations in the network in topological order, from 1 to n. (A topological ordering gives stations the property that if there is a work flow from station i to j, station j's number is higher than station i's number.)
2. Set all the control rules to be instruction control rules. Evaluate var(**P**) and **Z**(var(**P**)).
3. Starting with station n, consider all the stations in reverse topological order, as follows. Let the station considered be station i. Evaluate quantity $Z_{N,i}$, the sum of the Z_j(var(**P**))'s for stations i to n. Mathematically, this means calculating $Z_{Ni} = \sum_{j=i}^{n} Z_j(\text{var}(\mathbf{P}))$. Next, switch the control rule at station i to be the job-control rule. Recalculate the sum of the Z_j(var(**P**))'s for stations i to n, using the new control rule at station i, and call this quantity Z_{Ji}. If $Z_{Ji} < Z_{Ni}$, make the change to the job-control rule permanent; otherwise, switch station i's control rule back to instruction-control.
4. Let $i \leftarrow i-1$, and repeat step 3 until all stations have been considered.

This heuristic is not guaranteed to be optimal for two reasons. First, it assumes that switching the rule at station i does not change the relative values of the control rule choices for stations $i+1,\ldots,n$. Second, it ignores the effects of multiple stations sending work to a single downstream station (since the sending stations are only considered individually). Nonetheless, this heuristic does account for the dominant behavior of control rule assignments, and makes locally optimal decisions on whether the job-control or instruction-control rule is preferred. The resulting vector of control rule choices is likely to be near optimal. If a better solution is desired, one may explore various neighborhood search heuristics, using AssignRules to generate a starting solution.

AssignRules assumes that the modeled network is acyclic. One of the strengths of MR-models, however, is that they allow feedback loops. In such cases, we use an approximate version of AssignRules; we find a topological order of the network by assuming that the feedback loops do not exist, and run the rest of the heuristic unchanged. Obviously, ignoring feedback loops degrades the performance of the heuristic. However, if the flows going across the feedback loops are small (for example, a small percentage of products failing testing being sent upstream for rework), the impact of the control rule choices across the feedback loop will be small, as well. Then AssignRules should produce near-optimal rule choices in these cases, as well.

For example consider the following data-processing network, similar to computer networks used by the U.S. Department of Defense. (This network was studied in Section 3 of Chapter 5.)

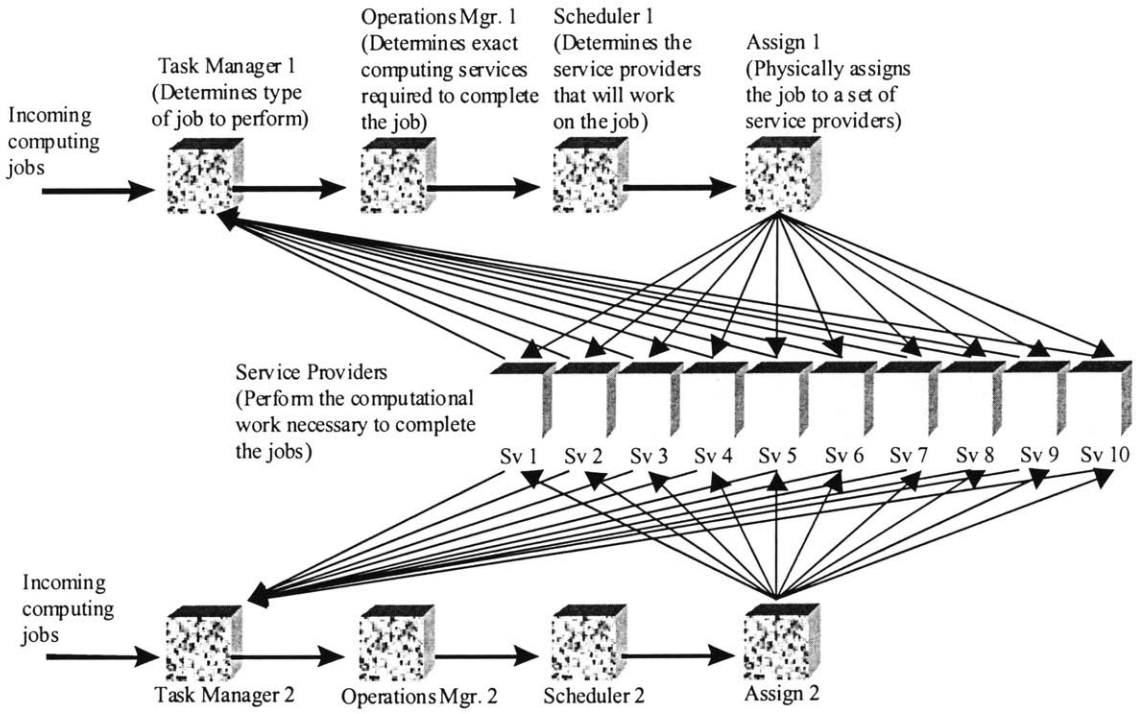


Figure 26 -- An Example LRS-MR Network

In this network, large numbers of requests are sent through control chains, which distribute them to a number of service providers. The service providers then process the jobs, sending a (usually) small fraction of the requests back to the control chains for reprocessing. The control chains see lots of jobs with small numbers of instructions per job; the service providers see fewer jobs, but with large numbers of instructions per job. This description implies that the control chains greatly influence the work of the service providers, but the service providers do not impact the control chains that heavily. Not

surprisingly, the heuristic usually (depending on input parameters) has the control stations use the job control rule, and the service provider stations use the instruction control rule.

6. A Performance Measure: Expected End-to-End Completion Times

The MR-model optimization problems usually involve a network performance measure, either as the objective function or as a constraint. Often, this measure is a weighted sum of expected waiting times or queue lengths. How to set these weights may not be obvious. If the problem is to minimize expected inventory costs, for instance, the vector of weights equals the vector of per-unit holding costs. However, suppose the problem is to maximize network performance as measured by "waiting times". Then it is not obvious how to set the weights.

This section discusses a way to set the weights that corresponds to an important performance measure in practice – the average end-to-end (ETE) completion time of all the job flows in the network. (We will explain why this method produces approximate results at the end of the section.)

We first explain what a job flow is. With the exception of the LRS-MR models, all the MR models we have studied have treated work as a continuous fluid rather than as distinct jobs. (Indeed, the LRS-MR model works by showing that the model moments can be treated as if they came from a continuous fluid.) Nonetheless, in practice, the shop being modeled processes distinct jobs. For example, the mainframe subcomponents shop, first considered in Chapter 3, routes circuit boards through a variety of processing steps. Consequently, a job flow tracks the number of jobs flowing through a station, not units of work.

We treat a job flow as if it were a fluid, like the work flows. However, unlike work flows, the number of jobs leaving a station exactly equals the number of jobs the station sends to downstream stations. A job leaving a station is assumed to appear, as is, at another station; no extra jobs are created or destroyed. We can make this assumption without a loss of generality, by assuming that components combined or split apart in the actual network maintain their job-flow identity. For example, assume that two parts are combined into a single circuit board at a station; the job flow leaving that station is still one (the circuit board now represents "two jobs".) Similarly, suppose a circuit board is split into two components that take different paths; each component represents "one half of a job."

Like work flows, we define a job-flow matrix $\mathbf{\Phi}^{\mathbf{F}}$. Similar to the workflow matrix $\mathbf{\Phi}$, we define Φ_{ij}^{F} to be the fraction of jobs leaving station j that arrive at station i. Mathematically, the constraint that no extra jobs are created or destroyed becomes $\sum_j \Phi_{ij}^{F} = 1$ for all stations j. In general, $\mathbf{\Phi} \neq \mathbf{\Phi}^{\mathbf{F}}$.

We also define a vector of expected job arrivals, $\boldsymbol{\mu}^{\mathbf{F}}$. We define μ_i^F to be the expected jobs that first enter the network through station i. For example, in the mainframe subcomponents model, an

average of 40 circuit boards enter processing per period at one station, and an average of 30 circuit boards enter processing per period at another station. The two μ_i^F's are 40 and 30, respectively.

Importantly, we calculate the steady-state expected job flow across each station, E(**F**), the same way we calculate E(**P**). We have:

$$E(\mathbf{F}) = (\mathbf{I} - \mathbf{\Phi}^{\mathbf{F}})\boldsymbol{\mu}^{\mathbf{F}}. \tag{18}$$

We now define our performance measure – the average end-to-end completion time of the job flows. In words, this is the average, across all stations, of the average waiting time at a station weighted by the number of jobs visiting that station. Note that if a job visits a station more than once we count it more than once. By definition, this measure is the following double-summation:

$$W_{ETE} = \sum_{\text{stations } i} \sum_{n=1}^{\infty} \left(W_i \cdot \frac{f_{i,n}}{F_k} \right). \tag{19}$$

Here, W_i is the expected waiting time at station i. F is the total flow of jobs across entire network, measured as the average number of jobs that arrive to the shop each time period. F is therefore the sum of the elements of $\boldsymbol{\mu}^{\mathbf{F}}$. For example, for the mainframe subcomponents network, $F = 70$. Finally, $f_{i,n}$ is the job flow across station i of work that has visited station i exactly n times. For example, suppose all jobs go through station i. However, station i has a feedback loop to itself, so that half of the work the leaves station i returns to station immediately. Then $f_{i,k,1} = F, f_{i,k,2} = F/2, f_{i,k,3} = F/4$, etc. We can simplify W_{ETE} quickly by recognizing that $\sum_{n-1}^{\infty} f_{i,n}$ is the steady-state expected job flow across station i, E(F_i).

Then, we rewrite W_{ETE} into a form representing W_{ETE} as the inner product of the expected waiting times and a vector of weights, as desired:

$$\begin{aligned} W_{ETE} &= \sum_i W_i \cdot w_i, \\ w_i &= \frac{1}{F} \cdot E(F_i), \\ E(\mathbf{F}) &= (\mathbf{I} - \mathbf{\Phi}^{\mathbf{F}})\boldsymbol{\mu}^{\mathbf{F}}. \end{aligned} \tag{20}$$

We note that this scheme for setting the weights produces the true W_{ETE} provided the control rules are linear; with general control rules, the resulting W_{ETE} may be approximate. The reason for this statement is as follows: the formula for W_{ETE} uses the expected waiting times, the W_i's. The difficulty is that the actual end-to-end completion times do not depend on the W_i's per se – they depend on the average waiting times the jobs actually see as they enter the stations. The two quantities generally are not the same; the exception is for queues obeying the ASTA property, or "arrivals see time averages." MR

models with linear control rules and independent work arrivals from outside the station obey this property, as shown in the following lemma.

Lemma. *Suppose we have a MR model with linear control rules and independent and stationary work arrivals. Then, all stations in the MR model obey the ASTA property.*

Proof. An alternate definition of ASTA is that the steady-state distribution of $\mathbf{Q}$ equals the steady-state distribution of $\mathbf{Q}_t$ just after an arrival has occurred. In Chapter 2, we found that the steady-state distribution of $\mathbf{Q}$ is:

$$\mathbf{Q} \equiv \sum_{s=0}^{\infty} (\mathbf{I} - \mathbf{D} + \mathbf{\Phi D})^s \boldsymbol{\varepsilon}_{t-s}, \tag{21}$$

where $\mathbf{D}$ is the matrix of smoothing factors, Φ is the workflow matrix, and ε_t is the vector of exogenous work arrivals.

The steady distribution of $\mathbf{Q}_t$ just after an arrival has occurred is given by the inventory balance equation, which is:

$$\mathbf{Q}_t \equiv \mathbf{Q}_{t-1} - \mathbf{P}_{t-1} + \mathbf{A}_t, \tag{22}$$

and, using formulas from Chapter 2, (22) can be rewritten as:

$$\mathbf{Q}_t \equiv (\mathbf{I} - \mathbf{D} + \mathbf{\Phi D})\mathbf{Q}_{t-1} + \boldsymbol{\varepsilon}_t. \tag{23}$$

Now, since the network is assumed to be in steady-state, $\mathbf{Q}_{t-1}$ has the same distribution has (21). Then (23) becomes:

$$\mathbf{Q}_t \equiv \sum_{s=1}^{\infty} (\mathbf{I} - \mathbf{D} + \mathbf{\Phi D})^s \boldsymbol{\varepsilon}_{t-s} + \boldsymbol{\varepsilon}_t, \tag{24}$$

which exactly equals (21), as desired. □

As noted, however, the above lemma does not apply to MR models with general control rules. In such cases, the computed W_{ETE} will be approximate, along with all other results pertaining to general control-rule models.

7. Examples of MR-Model Optimization Problems

In this section, we present two examples of MR-model optimization problems. First, we maximize the performance of an LRS-MR network given a budget constraint. In doing so, we use the interpolated functions for the safety factors (developed in Section 2.1) to create nonlinear programs. The model being optimized is the Department of Defense network discussed in Section 3 of Chapter 5, and the performance objective is to minimize a weighted sum of the expected waiting times.

Second, we minimize the cost of a GLS-MR model subject to constraints on the expected waiting times. Here, we optimize a model similar to the mainframe subcomponents model considered by Fine and Graves (1989).

7.1 Maximizing the Performance of an LRS-MR Network

7.1.1 Using a Safety Factor Function

Before we present the specific optimization problem, we show how to use the interpolated safety factor function developed in Section 2.1. Recall that through our statistical analyses we interpolated the following safety factor function:

$$k_i(H_i, s_i, L_i) = \left(-\frac{2.5881}{s_i\sqrt{H_i}} + \frac{2.5935}{s_i} + 0.5233 \right)^{-1} = \frac{s_i}{-2.5881/\sqrt{H_i} + 2.5935 + 0.5233 s_i}. \quad (25)$$

We replace $\mathbf{k(H, L, R_E)}$, with this function. Further, we also place lower bounds on the safety factors (requiring k to be greater than 0.4, for example). Thus, we replace the following constraint in general form,

$\mathbf{P_E(R_E) + k(H,L,R_E) \bullet \sqrt{diag(S_P(L,R_E))} \le z}$, Capacity required to meet the current performance level is less than the capacity currently installed, for all stations.

with the following sets of constraints:

$\mathbf{P_E(R_E) + k \bullet \sqrt{diag(S_P(r,L,R_E))} \le z}$, Same constraint as above

$k_j = \dfrac{\sqrt{S_{P,jj}(\mathbf{L},\mathbf{R_E})}/P_{E,j}(\mathbf{R_E})}{-2.5881/\sqrt{H_i} + 2.5935 + 0.5233\left(\sqrt{S_{P,jj}(\mathbf{L},\mathbf{R_E})}/P_{E,j}(\mathbf{R_E})\right)}, \forall j,$ Safety factors are at least that required by function $K_j(H_j, s_j, L_j)$

$\mathbf{k \ge K}$, Safety factors are at least some minimum value

We assume that the H_j's are parameters close to one, set by the user.

Clearly, we want to choose the smallest possible safety factors, since we want to minimize the amount of capacity that needs to be installed to meet a given performance level. Then we can substitute

the right sides of the inequalities for the safety factors directly into the performance-capacity constraints. Doing so, and simplifying the resulting expression, yields the following performance-capacity constraints:

$$P_{E,j}(\mathbf{R_E}) + \frac{S_{P,jj}(\mathbf{L},\mathbf{R_E})}{P_{E,j}(\mathbf{R_E})\left(-2.5881/\sqrt{H_j}+2.5935\right)+0.5233\sqrt{S_{P,jj}(\mathbf{L},\mathbf{R_E})}} \le z_j, \forall \text{ stations } j. \qquad (26)$$

7.1.2 Problem Description

We find the maximum performance of the example network given in Section 3 of Chapter 5, subject to a budget constraint. Figure 27 shows the network.

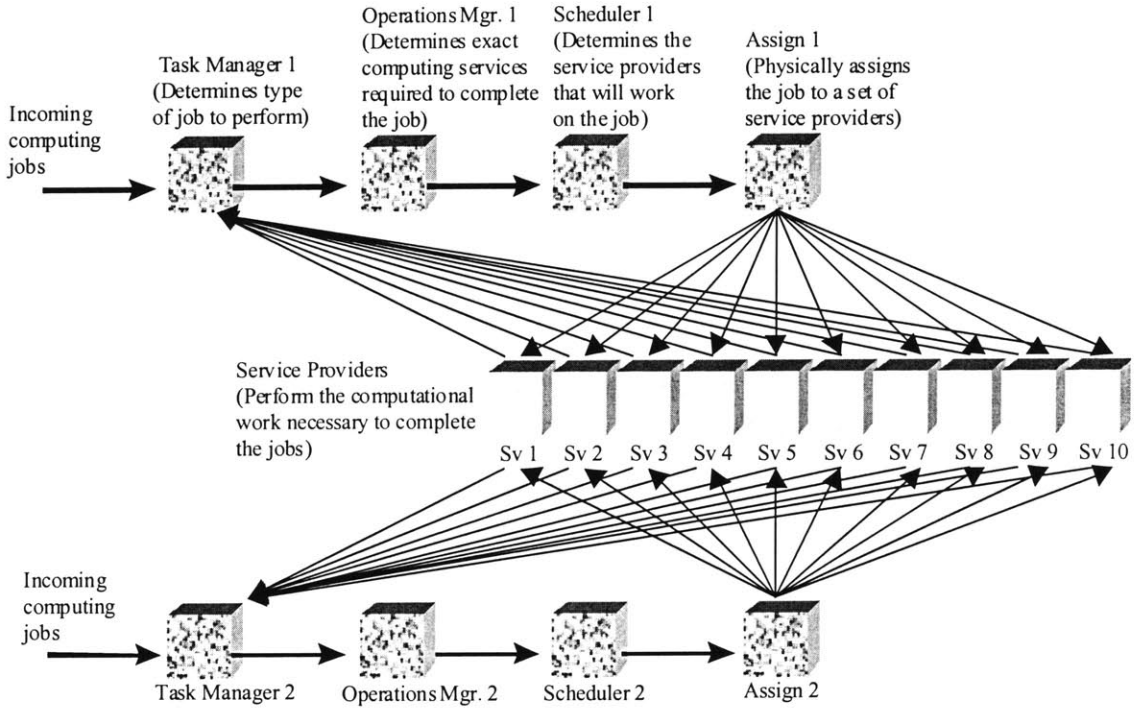


Figure 27 -- An Example LRS-MR Network

Here, we define maximum performance to be a weighted sum of the expected waiting times, which, for linear control rule networks, becomes a weighted sum of the station lead times. The input parameters for this network were presented in Section 3 of Chapter 5. To create a nonlinear program to maximize performance, we add the following data to the input parameters:

- The cost of adding a single unit of capacity anywhere in the network is $1.00.
- H_j, the parameter used in setting the safety factor, is fixed at 1.005 for all stations.
- The expectation and variance of all the inputs to the network are fixed to be the values given in Section 3 of Chapter 5.

- The weights of the lead times are chosen so that the weighted sum will equal W_{ETE}.

Consequently, the only variables are the lead times, since they are also the expected waiting times. Therefore, all the expected production values become constants with respect to the nonlinear program, so the production variances are solely functions of the lead times. To maximize performance, we make the lead times as small as possible while keeping the needed capacity within the network's budget constraint. We have the following nonlinear program:

$\min \sum_i w_i L_i$, such that:	Minimize a weighted sum of the lead times.
$E(P_i) + \frac{S_{P,ii}(L)}{0.0119 \cdot E(P_i) + 0.5233\sqrt{S_{P,ii}(L)}} \le z_i, \forall_i,$	Expected production plus a multiple of the standard deviation of production is less than the capacity installed at the station. (The constraints result from substituting $H_i = 1.005$ for all stations, and simplifying)
$\sum_i \$1.00 z_i \le C,$	Cost of the total capacity installed is less than a budget constraint.
$L_i \ge 1, \forall_i.$	All lead times are greater than one.

We simplify this program further. All of the station capacities (the z_i's) are used solely in a single budget constraint, implying the production-capacity equations will be satisfied with equality. Then we replace the capacities with the expressions for the capacity demands in the budget constraint, yielding:

$$\min \sum_i w_i L_i, \text{ such that:}$$

$$\sum_i 1.00 \left(E(P_i) + \frac{S_{P,ii}(L)}{0.0119 \cdot E(P_i) + 0.5233\sqrt{S_{P,ii}(L)}} \right) \le C,$$

$$L_i \ge 1, \forall_i.$$

This nonlinear program has only the lead times as variables, and has only a single nonlinear constraint. Consequently, it is fairly simple to optimize.

7.1.3 Optimization Results

The above nonlinear program was solved for a variety of network budgets, ranging from about 1.1 to 1.5 times the sum of the stations' expected production. In "monetary" terms, the budget therefore ranged from \$3,550,000 to \$5,000,000. The following graph shows the results.

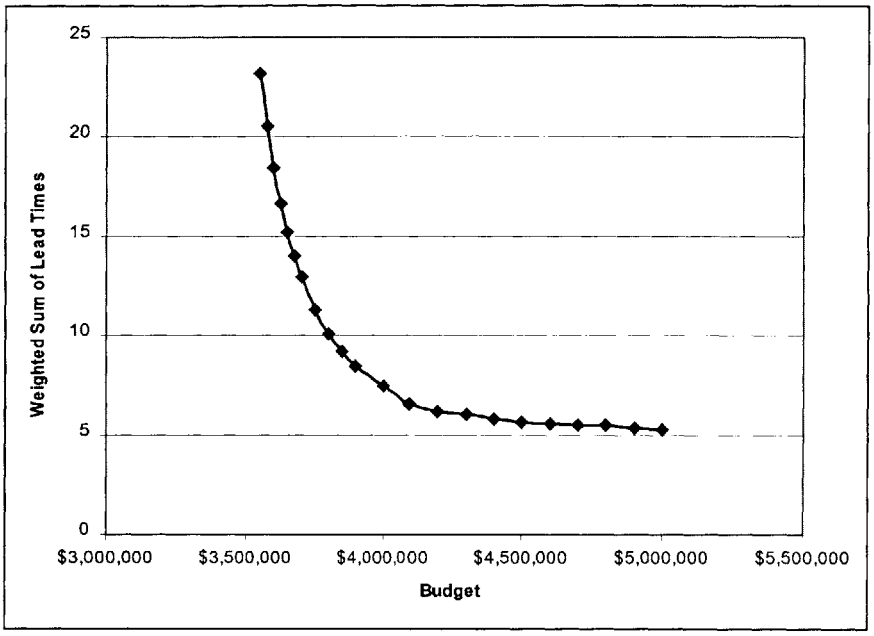


Figure 28 -- Optimal Performance Curve for the LRS-MR Model

We see that performance initially improves rapidly, but then levels off. An initial \$200 thousand increase in the budget, from \$3.55 million to \$3.75 million, halves the weighted sum of the lead times (from 23.2 to 11.3). However, an increase more than three times as large, from \$3.75 million to \$4.5 million, is needed to halve the weighted sum of the lead times again (from 11.3 to 5.6). Performance improvements will level off after this budget value. The minimum possible value of the weighted sum is 5, which occurs when all lead times are set to one.

7.2 Minimizing the Cost of a GLS-MR Network

In this section, we minimize the capacity cost of a network similar to the Mainframe Subcomponents job shop in Chapter 3, first considered by Fine and Graves (1989).

7.2.1 Problem Assumptions

Before considering the specifics of the problem, we derive the form of the minimum cost program. We assume that the cost of capacity is linear, so that the cost of one unit of capacity at station$i$ is given by c_i. We also assume that the vector of expected arrivals, μ, is fixed, implying that expected production is also fixed. (Recall that in chapter 3 we found that expected production was solely a function of μ).

We assume that all stations use clearing functions as production functions. Recall that a clearing function has the form:

$$p_i(Q_i) = \frac{M_i Q_i}{\beta_i + Q_i}, \tag{27}$$

where M_i is the asymptotic maximum production of the station, and β_i is a parameter determining the rate at which production approaches the maximum. For the sake of simplicity, we assume that the capacity at station i, z_i, equals M_i, and that $M_i = \beta_i$. With this choice of β_i, the first derivative of the clearing function approaches one as the work-in-queue approaches zero, implying that the station will (approximately) process small queue levels in a single period. Substituting, the production function used at each station is:

$$p_i(Q_i) = \frac{z_i Q_i}{z_i + Q_i}. \tag{28}$$

We assume that we use the *MomentsP* algorithm, developed in Chapter 3, to estimate the expected queue lengths. Using MomentsP, a second-order estimate of $E(Q_i)$ is:

$$E(Q_i) \approx \frac{z_i E(P_i)}{z_i - E(P_i)} + \frac{z_i^2}{(z_i - E(P_i))^3} \cdot \text{var}(P_i). \tag{29}$$

Note that $E(P_i)$ will be a constant, since $E(\mathbf{P})$ is a function of μ, and we assume μ is fixed. However, $\text{var}(P_i)$ is a function of $\mathbf{z}$. Using MomentsP, a first-order estimate for $\text{var}(\mathbf{P})$ as a function of $\mathbf{z}$ is:

$$\mathbf{S_P(z)} = \text{var}(\mathbf{P}) \approx \sum_{s=0}^{\infty} \mathbf{B}^s \mathbf{\Psi}^{-1} \Sigma \mathbf{\Psi}^{-1} \mathbf{B}'^s, \text{where } \mathbf{B} = \mathbf{I} - \mathbf{\Psi}^{-1} + \mathbf{\Psi}^{-1}\mathbf{\Phi}, \tag{30}$$

where Σ is the input covariance matrix of production, and Ψ is a diagonal matrix with elements:

$$\Psi_{ii} = \frac{z_i^2}{(z_i - E(P_i))^2}. \tag{31}$$

By definition, $\text{var}(P_i)$ is the (i,i) element of $\mathbf{S_P(z)}$.

We derived the general form of a minimum cost problem for GLS-MR networks in Section 3.3.3. If we substitute the linear cost function, the fixed μ, and the formulas for the expected queue lengths into this nonlinear program, we find:

Constraint	Description
$\min \mathbf{c}' \cdot \mathbf{z}$	Minimize the total cost of the network.
Subject to	
$\mathbf{z} \geq \boldsymbol{\mu}$	Capacity installed at a station must be greater than the expected production at that station.
$\sum_i \frac{w_i}{E(P_i)} \cdot \left(\frac{z_i E(P_i)}{z_i - E(P_i)} + \frac{z_i^2}{(z_i - E(P_i))^3} \cdot S_{P,ii}(\mathbf{z}) \right) \leq T$	A weighted sum of the expected waiting times is less than some constant, T.

7.2.2 Problem Description

We find the approximate minimum cost of a network similar to the mainframe subcomponents network considered in Section 4.3 of Chapter 3. (By definition, any "optimal" results using GLS-MR models will be approximate, since the queue length and production moment calculations are approximations.) Recall that the shop contains 13 stations; work enters through the NB Release and OTN Release stations, and exits through the Encapsulation station. The following figure diagrams the job shop. Note that the numbers on each arc (i,j) represent the expected amount of work that arrives at station j given one unit of work at station i.

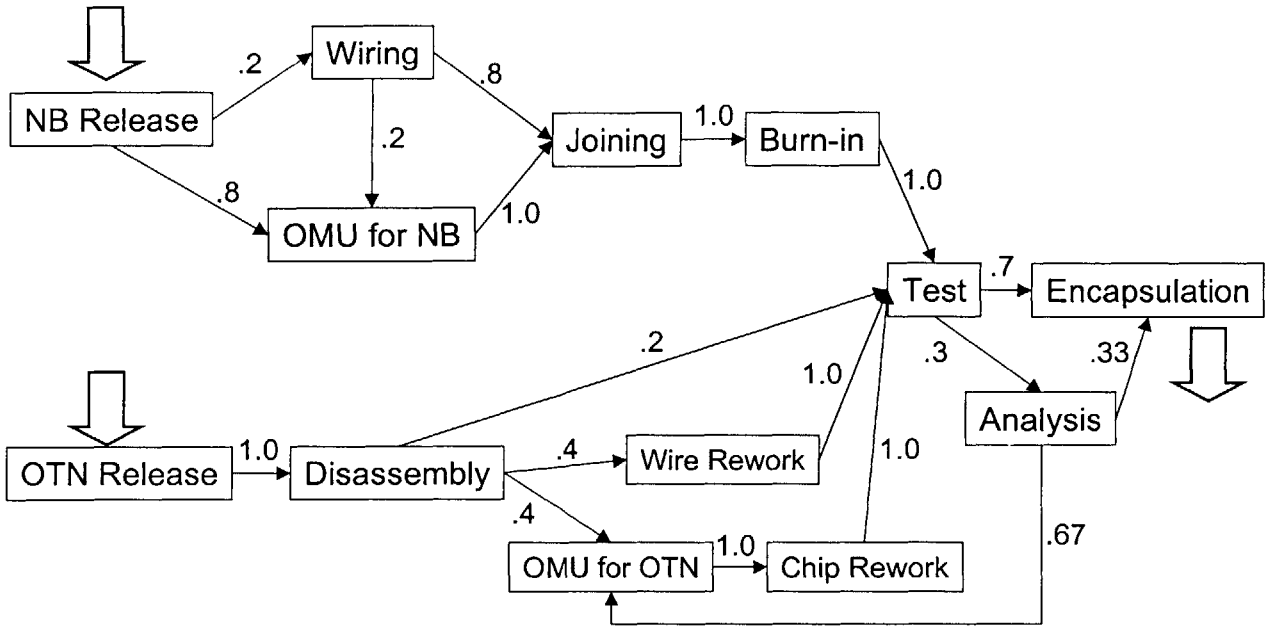


Figure 29 -- An Example GLS-MR Network

The following table shows the input data for the job shop. The "maximum capacities" are the capacities stated in Fine and Graves (1989). Our problem is to optimize these capacities to minimize network cost.

Station	*Average Arrivals per Period*	*Input Standard Deviation*	*Maximum Capacity* (z_i)	*Expected Production*
NB release	40	20	80	40
Wiring		6.3	16.3	12
OMU for NB		0	60	30.4
Joining		0	50	40
Burn-in		10	50	40
OTN release	30	12.5	60	30
Disassembly		12.5	45	30
OMU for OTN		0	60	29.6
Chip rework		12.5	45	29.6
Wire rework		12.5	18.8	12
Test		17.5	125	87.6
Analysis		0	50	26.3
Encapsulation		7.5	87.5	70

We assume that the cost of capacity is the same at all stations, and equals \$100 per unit. We set the w_i's on the average waiting times in the constraints so that the resulting sum will equal the estimated expected end-to-end completion time for all work flows (discussed in Section 6). Thus, we seek to minimize network cost, given a requirement on the average expected end-to-end completion time.

When substituting the weights back into the expected waiting time constraint, we find that $w_i / E(P_i) = 0.0144$ for all stations. This result is due to two reasons. First, the workflow matrix for this job shop is actually a fractional routing matrix, so that $\mathbf{\Phi} = \mathbf{\Phi}_F$. Second, the vector of expected arrivals actually equals the expected number of components entering the shop each period, so that $\boldsymbol{\mu} = \boldsymbol{\mu}^F$.

7.2.3 Optimization Results for a Single Instance

We first study a single instance of the optimization problem. The estimated ETE completion time of the network, using the above capacities and calculating the second-order estimated expected queue lengths, is 24.8421 periods. The capacity cost of the network is \$74,760. Our first question is to determine whether we can reduce the cost of the network while keeping the same estimated ETE completion time.

The answer is yes; for this instance, the cost of the optimal solution is \$67,619. The following graph compares the current capacity allocations and the optimal capacity allocations.

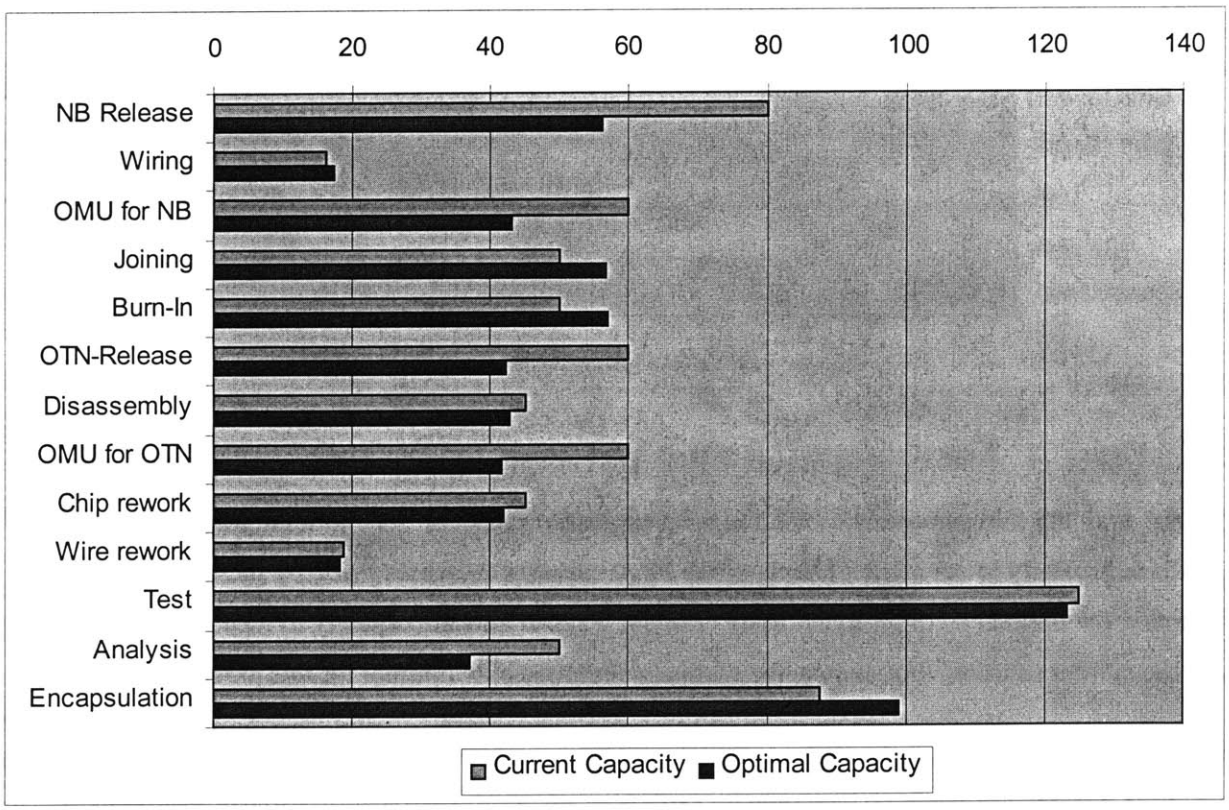


Figure 30 -- Current v. Optimal Capacities in the GLS-MR Network

The optimal solution economizes on capacity at a number of stations, especially NB release, OMU for NB, OTN Release, Analysis, and OMU for OTN. The optimal solution makes a few modest capacity

increases at the Wiring, Joining, Burn-In, and Encapsulation stations, but these are significantly less than the capacity decreases. The total reduction in cost from the current to optimal solution is about 9.5%

Figure 31 shows how the optimal solution keeps the performance constant while economizing on capacity.

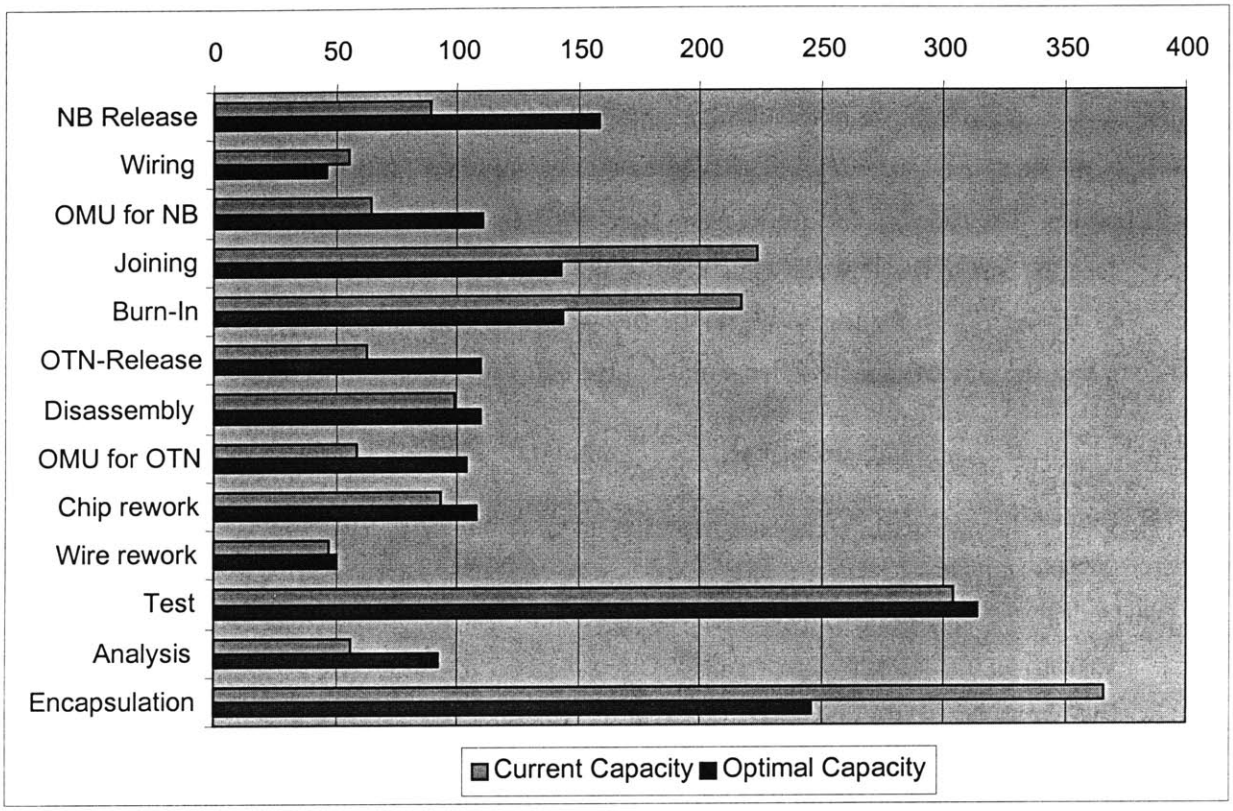


Figure 31 – Expected Queue Lengths in the Current and Optimal GLS-MR Networks

We see that the comparatively small capacity increases made by the optimal solution reduce queue lengths sufficiently to counteract effects of the large capacity decreases made by the optimal solution. In summary, the optimal solution found more efficient ways to allocate capacity to meet a performance goal than the current solution. (Of course, the optimal solution is only optimal for this particular instance.)

7.2.4 A Minimum-Cost Performance Curve

We solve the minimum-cost problem for a variety of required ETE completion times, ranging from 9 periods to 50 periods. (The asymptotic minimum ETE completion time for this problem is 6.82 periods, which assumes that each station processes its entire queue in one period.) The following graph shows the results.

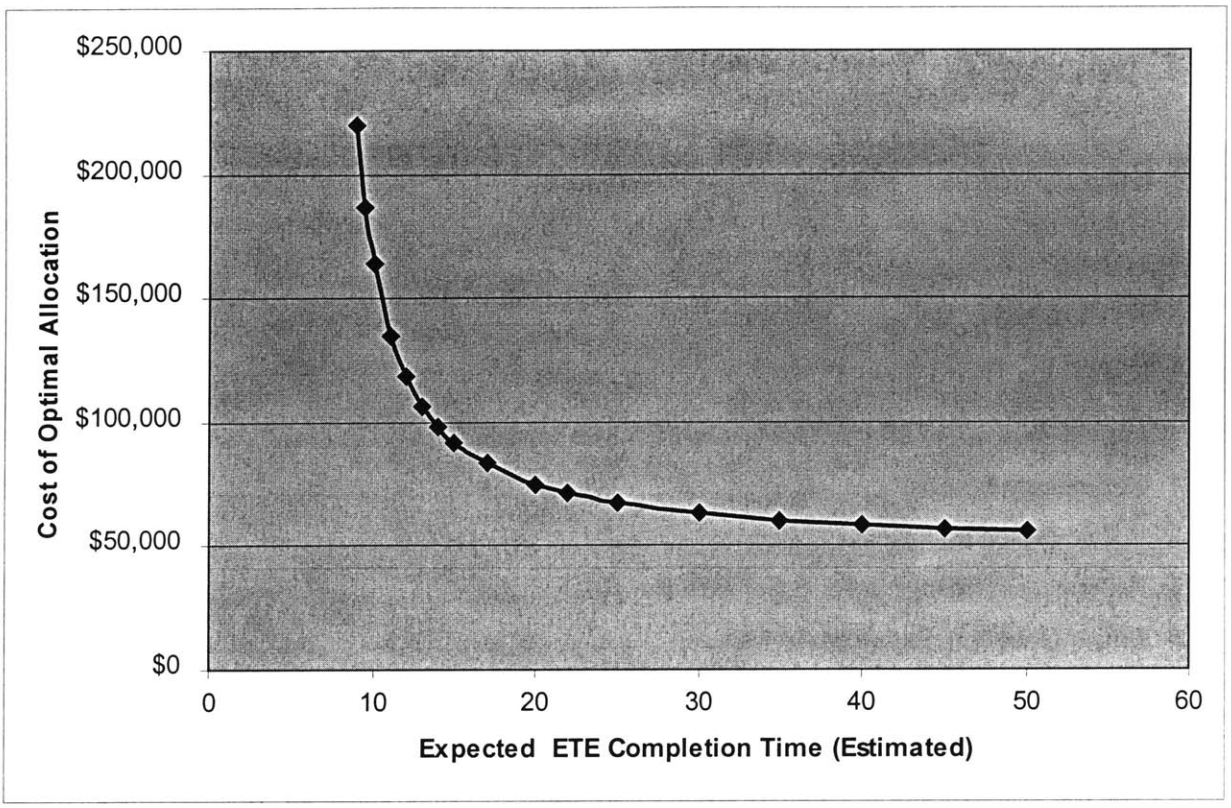


Figure 32 -- Optimal Cost Curve for the GLS-MR Network

We see that, initially, one can greatly economize by accepting slightly worse network performance, but these savings level out quickly. Initially, one can more than halve the capacity cost by going from an ETE completion time of 9 periods to 13 periods. However, it is not possible to halve the cost again, even by almost quadrupling the ETE completion time to 50 periods.

In summary, we have shown how to derive and solve optimization problems involving MR models, including maximum performance problems and minimum cost problems. We have considered two examples of optimization problems. In both examples, we found that one could initially make substantial gains in the optimal solution's objective value through small relaxations in the constraints, but that successive constraint relaxations resulted in smaller improvements to the optimal objective value. These results imply that in designing MR-model networks, we will often find a "knee" in the tradeoff curve between performance and cost. Solutions at the knee will provide good performance at a reasonable cost, thus providing excellent indicators of how to design the network.

Chapter 7: Transient Analysis and Optimization of Models with Linear Control Rules

1. Introduction

The previous chapters of this thesis have focused heavily on steady-state analysis and optimization. In this chapter, we turn to the transient analysis and optimization of MR models. Here, we will use the moment recursion equations directly to track the production and queue length moments over time, as opposed to using the moment recursion equations to calculate steady-state results. We will also study the dynamic optimization of MR models under special conditions.

The use of MR models for transient analysis can be a powerful tool. One can predict how a network will behave from a given starting point, given a set of predictions about changes to the inputs and other model parameters over time. MR networks usually only provide production and queue length moments (unless all arrivals are normally distributed), but these are likely to be the statistics the analyst most cares about.

Transient analysis allows the analyst to predict the near-term behavior of a network. It also allows the analyst to answer a variety of what-if questions (such as what would happen if certain types of arrival or processing disturbances occurred), without recourse to simulation studies. It is also not a capability generally allowed in conventional queuing networks. While a great deal of research on the transient behavior of queues has been done, most of this research has focused on individual queues, and the results have often been ungainly. For example, Abate and Whitt (1988) showed that the transient solution to an M/M/1 queue is given by a complicated second-order Bessel function. Cohen (1969) showed that, for G/G/1 systems beginning at rest, the transient terms of the expected queue lengths decay in an exponential manner. Venkatakrishnan, Barnett, and Odoni (1993) developed a numerical method for tracking the behavior of a two-queue M/M/1 queue with a varying arrival rate, and used it to estimate the daily average delays at Logan Airport. More recently, Asmussen (1995) and Karandkar and Kulkarni (1995) combined deterministic fluid flow models with an assumption that actual queue levels followed a Reflected Brownian Motion process, but this result only applies to individual stations.

A notable exception to only considering the transient behavior of a single queue is the work of Keilson and Servi (1994). They calculated the transient joint distribution of station populations in networks of $M(t)/G/\infty$ queues with respect to changes in job-arrival rates. In particular, they found that the transient joint distribution consists of product-form equations similar to the stationary $M/G/\infty$ case, with the terms corresponding to job arrival rates being function of time as opposed to constants. However, their analysis is not a complete transient analysis, since they require the initial state of the network to be a valid $M/G/\infty$ probability distribution as opposed to a deterministic assignment of jobs at stations.

The major drawback of transient analysis is that the moment results are exact only for models using linear control rules. As discussed in Chapter 3, the moment recursion equations for models with general control rules are approximations rather than exact formulas – in particular, the expectation equations create second-order estimates, while the variance equations create first-order estimates that feed into the expectation equations. Thus, moment estimates generated for models with general control rules may degrade quickly, as the recursion equations successively generate approximations from approximations. We expect that the approximate estimates may be useful for predicting network behavior over a few periods, though. Nonetheless, all the models discussed in this chapter use linear control rules and have stationary arrivals, as in Chapter 2 (MR model class LLS-MR).

Section 2 presents mathematical equations for the transient behavior resulting from several common changes. We consider the transient behavior resulting from a one-period impulse and a permanent increase in expected arrivals. We also consider the transient behavior resulting from a linear increase in work arrivals, and the behavior resulting from an oscillation between a low expected arrival rate and a high expected arrival rate. This section largely reviews the work of Parrish (1987), who studied the effects of these transients on the original Tactical Planning Models (recall that these are LLS-MR models in which a station's production is a fixed fraction of its work-in-queue).

Section 3 discusses the direct use of the moment recursion equations to track the behavior of a LLS-MR shop over time. In addition to discussing the use of the recursion equations to track queue levels and production quantities on a period-by-period basis, we show how to calculate the moments of the aggregate network production over a number of periods. As an example, we consider a ten-station shop that produces precision grinders, first considered by Graves (1986). We track the moments of production at the final station on a period-by-period basis, and compute the moments for 100 periods of aggregate production for all stations. We calculate these results for two different scenarios: a scenario in which expected production orders to the first station increase linearly, and a scenario in which the expected orders in any particular period come from a uniform distribution. (In both scenarios, the standard deviation of the orders is directly proportional to the expected orders.)

Section 4 moves from transient performance analysis to transient optimization for a special class of problems involving LLS-MR models. Suppose that our problem is to minimize a quadratic function of the queue lengths and / or production quantities over a finite set of periods. (A function involving production variances would be an example.) Then, it can be shown that the optimal policy for the production in any given period is given by a linear control rule, and that these policies may be found through a deterministic dynamic programming formulation. The section develops the dynamic programming procedure, and applies it to a 4-station example job shop.

2. Transient Analysis of Simple Changes to LLS-MR Models

In this section, we describe the mathematical behavior of LLS-MR networks resulting from simple changes to new work arrivals. We describe the mathematical behavior resulting from the following four changes:

- A large, single-period change in the expected arrivals, with the distribution of work arrivals returning to the steady-state distribution immediately afterwards.
- A permanent change in the expected arrivals.
- Linear growth in expected arrivals over successive periods.
- A regular oscillation in which the expected arrivals vary between a low level and a high level.

All of these changes affect only the expected work arrivals; the work arrival variances are assumed to be constant. Thus, the studied changes affect the transient behavior of expected production and queue lengths solely.

Our work draws heavily on the work of Parrish (1987), who considered the effects of these transients on Tactical Planning Models. (Our derivations are somewhat different that Parrish's, however.) Recall that the Tactical Planning Model assumes that the work produced at a station is a fixed fraction of the work-in-queue at that station, or:

$$P_{it} = \alpha_i Q_{it}, \tag{1}$$

where α_i is a smoothing factor between 0 and 1.

For the most part, Parrish's analysis applies directly to the more general LLS-MR models, which have production functions of the following form:

$$P_{it} = \sum_j \alpha_{ij} Q_{jt} + \beta_i + \gamma_{it}, \tag{2}$$

where the α_{ij}'s are general smoothing factors, β_i is a constant production quantity and γ_{it} is a random variable with zero mean and finite variance. In matrix vector form, (2) becomes:

$$\mathbf{P}_t = \mathbf{D}\mathbf{Q}_t + \boldsymbol{\beta} + \boldsymbol{\gamma}_t, \tag{3}$$

where $\mathbf{D}$ is the matrix of the α_{ij}'s, β is the vector of constant production quantities, and γ_t is the vector of production fluctuations.

Parrish's analysis applies for several reasons. First, we only consider results affecting expected production and queue length quantities, so we do not consider the terms representing production fluctuations. Further, in Chapter 2 we found that the β_i terms cancel out of the recursion equation for

production, so we will only incorporate β_i when we calculate the transient expected queue lengths from the transient expected production quantities. Finally, Parrish's analysis does not require **D** to be diagonal, provided some additional conditions are met (see the next section).

We note that knowing the transient behavior of expected production immediately yields the transient behavior of the expected queue lengths. By solving (3) for $\mathbf{Q}_t$, and taking the expectations of both sides, we find that:

$$\mathbf{E}(\mathbf{Q}_t) = \mathbf{D}^{-1}\left(\mathbf{E}(\mathbf{P}_t) - \boldsymbol{\beta}\right) \tag{4}$$

Thus, in the following sections, we focus exclusively on formulas for the transient expectations of the production quantities.

2.1 Preliminary Results

We first state several results of use in calculating the transient behavior of LLS-MR networks. These results are stated without proof; details are given in Parrish (1987).

Consider the matrix $\mathbf{B} = (\mathbf{I} - \mathbf{D} + \mathbf{D}\Phi)$, where **I** is the identity matrix, Φ is the workflow matrix of an LLS-MR model, and **D** is the matrix of smoothing factors that determines the amount of work produced at each station. Recall that for Tactical Planning Models, **D** is a diagonal matrix with diagonal entries between 0 and 1; for general LLS-MR models, **D** can contain negative and off-diagonal entries. **B** is of great significance in analyzing LLS-MR models, as the recursion equation relating production quantities in one period to production quantities in the previous period is:

$$\mathbf{P}_t = \mathbf{B}\mathbf{P}_{t-1} + \mathbf{D}\boldsymbol{\varepsilon}_t + \boldsymbol{\gamma}_t, \tag{5}$$

where ε_t is vector of arrivals, and γ_t is the vector of production fluctuations. The recursion equation for $E(\mathbf{P}_t)$ is:

$$\mathbf{P}_t = \mathbf{B} \cdot E\left(\mathbf{P}_{t-1}\right) + \mathbf{D}\boldsymbol{\mu}, \tag{6}$$

where μ is $E(\varepsilon_t)$. No γ terms appear, since $E(\gamma_t) = 0$, by assumption.

Assume that **B** has a spectral radius less than one. Then we have the following results concerning power series of **B**:

$$\sum_{s=0}^{\infty} \mathbf{B}^s = \left(\mathbf{I} - \mathbf{B}\right)^{-1} \tag{7}$$

$$\sum_{s=0}^{t-1} \mathbf{B}^s = \left(\mathbf{I} - \mathbf{B}\right)^{-1}\left(\mathbf{I} - \mathbf{B}^t\right) \tag{8}$$

$$\mathbf{B}^t(\mathbf{I}-\mathbf{B})^{-1} = (\mathbf{I}-\mathbf{B})^{-1}\mathbf{B}^t \tag{9}$$

$$\sum_{s=0}^{t-1} s\mathbf{B}^s = (\mathbf{I}-\mathbf{B})^{-1}(\mathbf{I}-\mathbf{B}^t)(\mathbf{I}-\mathbf{B})^{-1} - (\mathbf{I}-\mathbf{B})^{-1}\left[\mathbf{I}+(t-1)\mathbf{B}^t\right] \tag{10}$$

All four results are given in Parrish (1987).

In addition to these algebraic results, we present two matrix-theory results. Without loss of generality, suppose that **B** is nonnegative (recall that in Chapter 2, we showed how to represent negative smoothing factors using the Pull Model formulation of Leong (1989)). Then a theory of nonnegative matrices states $(\mathbf{I}-\mathbf{B})^{-1} \geq 0$, (Lancaster and Tismenetsky, 1985, p. 531) and a result from the Frobenius theory of reducible nonnegative matrices states that **B** has a real, positive eigenvalue equal to its spectral radius (Gantmacher, 1959, p. 66).

2.2 A One-Period Impulse

We first present the behavior resulting from a one period impulse. Assume the network is in steady-state, so that the expected production in period zero is simply E(**P**). In period zero, suppose that the vector of expected work arrivals is $\delta + \mu$ rather than μ. From (6), we find that:

$$E(\mathbf{P}_1) = \mathbf{B}\cdot E(\mathbf{P}) + \mathbf{D}\cdot(\boldsymbol{\mu}+\boldsymbol{\delta}). \tag{11}$$

By the definition of steady-state expected production, $E(\mathbf{P}) = \mathbf{B}\cdot E(\mathbf{P}) + \mathbf{D}\boldsymbol{\mu}$. Then we can rewrite (11) as:

$$E(\mathbf{P}_1) = E(\mathbf{P}) + \mathbf{D}\boldsymbol{\delta}. \tag{12}$$

If we repeatedly iterate (11), remembering that $E(\mathbf{P}) = \mathbf{B}\cdot E(\mathbf{P}) + \mathbf{D}\boldsymbol{\mu}$, we find that:

$$E(\mathbf{P}_t) = E(\mathbf{P}) + \mathbf{B}^{t-1}\mathbf{D}\boldsymbol{\delta}. \tag{13}$$

Thus, we see that the transient expected production always equals E(**P**) plus a term representing the remaining impact of the impulse. Since the spectral radius of **B** is less than one, this term decays asymptotically to zero as the period number increases.

2.3 A Permanent Change in Expected Arrivals

Again, assume that the network is in steady-state, so that expected production equals E(**P**) at time zero. Now suppose that the vector of expected arrivals changes by $\boldsymbol{\delta}$ in period zero, and that this change remains for all successive periods. Using (6), we have that:

$$\begin{aligned} E(\mathbf{P}_1) &= \mathbf{B} \cdot E(\mathbf{P}) + \mathbf{D}(\boldsymbol{\mu} + \boldsymbol{\delta}), \\ &= E(\mathbf{P}) + \mathbf{D}\boldsymbol{\delta}, \end{aligned} \tag{14}$$

where the second equation follows from the fact that $E(\mathbf{P}) = \mathbf{B} \cdot E(\mathbf{P}) + \mathbf{D}\boldsymbol{\mu}$. Iterating (14), and noting that $E(\mathbf{P}) = (\mathbf{I} - \mathbf{B})^{-1} \mathbf{D}\boldsymbol{\mu}$, we find that:

$$\begin{aligned} E(\mathbf{P}_t) &= E(\mathbf{P}) + \sum_{s=0}^{t-1} \mathbf{B}^s \mathbf{D}\boldsymbol{\delta}, \\ &= E(\mathbf{P}) + (\mathbf{I} - \mathbf{B}^t)(\mathbf{I} - \mathbf{B})^{-1} \mathbf{D}\boldsymbol{\delta}, \\ &= (\mathbf{I} - \mathbf{B})^{-1} \mathbf{D}(\boldsymbol{\mu} + \boldsymbol{\delta}) - \mathbf{B}^t (\mathbf{I} - \mathbf{B})^{-1} \mathbf{D}\boldsymbol{\delta}. \end{aligned} \tag{15}$$

where the second equation follows from (8). Thus, the transient expected production tends towards a new equilibrium, given by $E(\mathbf{P}) + (\mathbf{I} - \mathbf{B})^{-1} \mathbf{D}\boldsymbol{\delta}$. The spectral radius of $\mathbf{B}^t$ dictates the rate of change to this new state.

2.4 Linear Growth in Expected Arrivals

As in Section 2.2, suppose that the expected production in period zero is E(**P**). Suppose that, starting in period 1, the expected arrival vector is given by $\boldsymbol{\mu} + t\boldsymbol{\delta}$. Using (6), and iterating the resulting equation, we find that:

$$E(\mathbf{P}_t) = \mathbf{B}^t E(\mathbf{P}) + \sum_{s=0}^{t-1} \mathbf{B}^s \mathbf{D}(\boldsymbol{\mu} + (t-s)\boldsymbol{\delta}). \tag{16}$$

By the definition of steady-state expected production, we have that:

$$E(\mathbf{P}) = \mathbf{B}^t E(\mathbf{P}) + \sum_{s=0}^{t-1} \mathbf{B}^s \mathbf{D}\boldsymbol{\mu}, \forall t. \tag{17}$$

Then, (16) can be rewritten as:

$$E(\mathbf{P}_t) = E(\mathbf{P}) + \sum_{s=0}^{t-1} t\mathbf{B}^s \mathbf{D}\boldsymbol{\delta} - \sum_{s=0}^{t-1} s\mathbf{B}^s \mathbf{D}\boldsymbol{\delta} \tag{18}$$

Using equation (8), the second term of (18) is expanded to:

$$\sum_{s=0}^{t-1} t\mathbf{B}^s \mathbf{D}\boldsymbol{\delta} = t(\mathbf{I} - \mathbf{B}^t)(\mathbf{I} - \mathbf{B})^{-1} \mathbf{D}\boldsymbol{\delta} \tag{19}$$

Using equation (10), the second term of (18) is expanded to:

$$-\sum_{s=0}^{t-1} s\mathbf{B}^s\mathbf{D}\boldsymbol{\delta} = -\left[(\mathbf{I}-\mathbf{B})^{-1}(\mathbf{I}-\mathbf{B})^t - \mathbf{I} - (t-1)\mathbf{B}^t\right](\mathbf{I}-\mathbf{B})^{-1}\mathbf{D}\boldsymbol{\delta} \tag{20}$$

Combining (19) and (20), we find:

$$\begin{aligned} E(\mathbf{P}_t) &= E(\mathbf{P}) + \left[t\left(\mathbf{I}-\mathbf{B}^t\right) - (\mathbf{I}-\mathbf{B})^{-1}(\mathbf{I}-\mathbf{B}^t) + \mathbf{I} + (t-1)\mathbf{B}^t\right](\mathbf{I}-\mathbf{B})^{-1}\mathbf{D}\boldsymbol{\delta} \\ &= E(\mathbf{P}) + \left[(t+1)\mathbf{I} - (\mathbf{I}-\mathbf{B})^{-1} + \left((\mathbf{I}-\mathbf{B})^{-1} - \mathbf{I}\right)\mathbf{B}^t\right](\mathbf{I}-\mathbf{B})^{-1}\mathbf{D}\boldsymbol{\delta} \end{aligned} \tag{21}$$

Again, $\mathbf{B}^t$ has a spectral radius less than one, so the term with $\mathbf{B}^t$ within it asymptotically decays to zero. Thus, we can divide (21) into a long-term, "steady-state" growth path,

$$\lim_{t\to\infty} E(\mathbf{P}_t) = E(\mathbf{P}) + \left[(t+1)\mathbf{I} - (\mathbf{I}-\mathbf{B})^{-1}\right](\mathbf{I}-\mathbf{B})^{-1}\mathbf{D}\boldsymbol{\delta}\,, \tag{22}$$

and a transient component in the expected growth path,

$$\mathbf{P}_{T,t} = \left((\mathbf{I}-\mathbf{B})^{-1} - \mathbf{I}\right)\mathbf{B}^t(\mathbf{I}-\mathbf{B})^{-1}\mathbf{D}\boldsymbol{\delta}\,. \tag{23}$$

2.5 Regular Oscillations in Expected Arrivals

Finally, Parrish (1987) considers a cyclical pattern of changes in the expected arrivals. Here, the expected arrivals increase by δ for c periods, then returns to μ for d periods, and the cycle then repeats Figure 33 shows the pattern of expected arrivals.

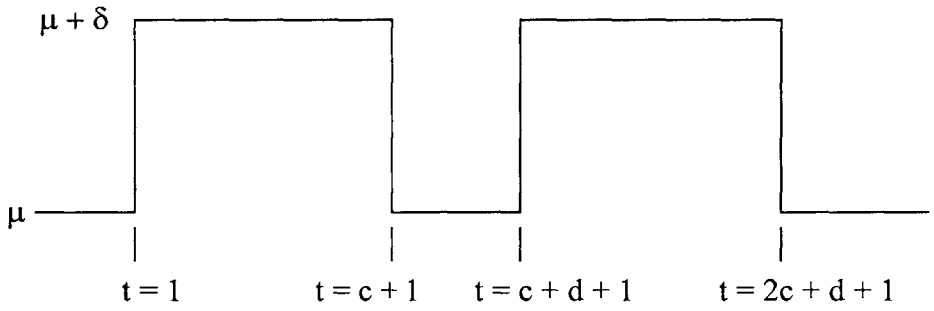


Figure 33 -- Cyclical Changes in Expected Arrivals

The pattern of expected arrivals, $E(\mathbf{A}_t)$, is given by:

$$\begin{aligned} &E(\mathbf{A}_t) = \boldsymbol{\mu}_t, && t < 1, t = (i+1)c + id + 1, \ldots, (i+1)(c+d), && \text{for } i = 0,1,2,\ldots \\ &E(\mathbf{A}_t) = \boldsymbol{\mu}_t + \boldsymbol{\delta}, && t = i(c+d)+1, \ldots, i(c+d)+c, && \text{for } i = 0,1,2,\ldots \end{aligned} \tag{24}$$

We examine $E(\mathbf{P}_t)$ over the intervals $t = 1$ to $t = c$, $t = c + 1$ to $t = c + d$, and $t = c + d + 1$ to $t = 2c + d$; all subsequent transients can be determined by repeating these.

In the first interval, the system responds to a steady increase in the expected arrival rate. In particular, at time $t = c$,

$$E(\mathbf{P}_t) = E(\mathbf{P}) + (\mathbf{I} - \mathbf{B}^c)(\mathbf{I} - \mathbf{B}^{-1})\mathbf{D}\boldsymbol{\delta}\,. \tag{25}$$

Thus, expected production equals the original E(**P**) plus a steadily-increasing transient showing the impact of the δ-increase to E(**A**).

In the second interval, the expected arrival vector returns to μ. Thus, we have that:

$$\begin{aligned} E(\mathbf{P}_{c+1}) &= \mathbf{B}\cdot E(\mathbf{P}_c) + \mathbf{D}\boldsymbol{\mu}, \\ &= (\mathbf{B}\cdot E(\mathbf{P}) + \mathbf{D}\boldsymbol{\mu}) + \mathbf{B}(\mathbf{I} - \mathbf{B}^c)(\mathbf{I} - \mathbf{B}^{-1})\mathbf{D}\boldsymbol{\delta}, \\ &= E(\mathbf{P}) + \mathbf{B}(\mathbf{I} - \mathbf{B}^c)(\mathbf{I} - \mathbf{B}^{-1})\mathbf{D}\boldsymbol{\delta}. \end{aligned} \tag{26}$$

Iterating (26) yields:

$$E(\mathbf{P}_{c+j}) = E(\mathbf{P}) + \mathbf{B}^j(\mathbf{I} - \mathbf{B}^c)(\mathbf{I} - \mathbf{B}^{-1})\mathbf{D}\boldsymbol{\delta}, \quad 1 \le j \le d \tag{27}$$

Thus, in the second interval, we have that expected production is the original E(**P**) plus a steadily-decreasing transient.

Finally, in the third interval, starting from $c + d + 1$, the expected arrival vector returns to μ + δ. We have that:

$$\begin{aligned} E(\mathbf{P}_{c+d+1}) &= \mathbf{B}\cdot E(\mathbf{P}_{c+d}) + \mathbf{D}(\boldsymbol{\mu} + \boldsymbol{\delta}), \\ &= E(\mathbf{P}) + \mathbf{B}^d(\mathbf{I} - \mathbf{B}^c)(\mathbf{I} - \mathbf{B}^{-1})\mathbf{D}\boldsymbol{\delta} + \mathbf{D}\boldsymbol{\delta}. \end{aligned} \tag{28}$$

Iterating this equation yields:

$$\begin{aligned} E(\mathbf{P}_{c+d+j}) &= E(\mathbf{P}) + \mathbf{B}^j\left[\mathbf{B}^d(\mathbf{I} - \mathbf{B}^c)(\mathbf{I} - \mathbf{B}^{-1})\mathbf{D}\boldsymbol{\delta}\right] + \sum_{s=0}^{j-1}\mathbf{B}^s\mathbf{D}\boldsymbol{\delta}, \\ &= E(\mathbf{P}) + \mathbf{B}^{j+d}(\mathbf{I} - \mathbf{B}^c)(\mathbf{I} - \mathbf{B}^{-1})\mathbf{D}\boldsymbol{\delta} + (\mathbf{I} - \mathbf{B}^j)(\mathbf{I} - \mathbf{B})^{-1}\mathbf{D}\boldsymbol{\delta}, \qquad 1 \le j \le c. \end{aligned} \tag{29}$$

Here, expected production equals the original E(**P**), plus a steadily-increasing transient showing the impact of the current δ-increase, but also plus a steadily-decreasing transient showing the remaining impact of the first interval, in which E(**A**) was increased by δ.

Therefore, in general, during intervals where E(**A**) = μ + δ, expected production will be E(**P**) plus a steadily increasing transient for the current increase in E(**A**), plus a decreasing transient showing the remaining impacts of all previous intervals. Similarly, during intervals where E(**A**) = μ, expected production will be E(**P**) plus a decreasing transient showing the remaining impacts of all previous intervals.

2.6 Discussion of Transients

If we review the decaying transients discussed in Sections 2.1 – 2.5, we see that they all have the form of **B** multiplied by some power times a constant vector. The exponent of **B** is a function of the number of periods elapsed since the change creating the transient entered the system. Consequently, the nature of the decay of the transients is governed by the powers of **B** through its eigenvalues.

We have already noted that, assuming **B** is positive and has a spectral radius less than one, B's maximum eigenvalue will be real, positive, and less than one. Thus, the transients always decay, asymptotically, to zero, ensuring that the network returns to a new equilibrium. The rate of decay depends on the size of the maximum eigenvalue of **B**. For example, Parrish has calculated the number of periods required for a transient to decay to 1% of its original value, given a variety of maximum eigenvalues:

Eigenvalue λ	Periods n for λ^n to decay to 0.01λ
.50	8
.60	10
.70	14
.80	22
.90	45
.95	90

The exact nature of the decay depends on the other eigenvalues of B, which may cause the nature of the decay to be fairly complex. Negative and complex eigenvalues tends to produce oscillatory patterns; negative eigenvalues generate period-by-period oscillations, while complex eigenvalues generate sinusoidal oscillations (Luenberger, 1979, pp. 154-170). Such behavior is not surprising. The **B** matrix represents the transfer of work between stations each period, so taking powers of **B** tracks the progress of a fixed amount of work through the network over multiple periods.

Notably, the magnitude of the maximum eigenvalue depends heavily on the smoothing factors in the **D** matrix. For example, in Tactical Planning Models, decreasing the smoothing factors correspond to increasing the lead times at stations. Doing so increases the queue levels, which corresponds to more work staying in the network for longer amounts of time. Thus, increasing lead times creates higher maximum eigenvalues. Parrish demonstrated this effect by showing how increasing the lead times in the mainframe subcomponents shop considered by Fine and Graves (1989) increased the maximum eigenvalue of the shop's **B** matrix.

3. Numerical Analysis of the Transient Behavior of LLS-MR Models

In the previous section, we derived algebraic formulas for certain simple changes that only affected expected production (and expected queue lengths). However, we are not limited to analyzing the impact

of these simple changes. Instead, we can use the recursion equations to calculate the transient moments resulting from any changes to any of the LLS-MR model parameters. We can examine the impacts of changing expected arrivals, variances of arrivals, variances of production fluctuations, and smoothing parameters in any pattern. This generality does imply that we cannot find simple algebraic expressions for most of the transient moments resulting from these changes. Nonetheless, we can calculate the moments numerically exactly, on a period-by-period basis.

3.1 The LLS-MR Recursion Equations

In Chapter 2, we found that the production recursion equation for a LLS-MR model is:

$$\mathbf{P}_t = \mathbf{B}\mathbf{P}_{t-1} + \mathbf{D}\boldsymbol{\varepsilon}_t + (\boldsymbol{\gamma}_t - \boldsymbol{\gamma}_{t-1}), \tag{30}$$

where $\mathbf{B} = \mathbf{I} - \mathbf{D} + \mathbf{D}\Phi$ (as discussed in Section 2), $\mathbf{D}$ is the matrix of smoothing parameters, ε_t is the vector of new work arrivals, and γ_t is the vector of noise in the production quantities. Taking the moments of (30) yields the transient moments of production:

$$E(\mathbf{P}_t) = \mathbf{B} \cdot E(\mathbf{P}_{t-1}) + \mathbf{D}\boldsymbol{\mu}_t, \text{ and} \tag{31}$$

$$\text{var}(\mathbf{P}_t) = \mathbf{B} \cdot \text{var}(\mathbf{P}_{t-1}) \cdot \mathbf{B}' + \mathbf{D}\boldsymbol{\Sigma}_t\mathbf{D}' + (\boldsymbol{\Gamma}_t + \boldsymbol{\Gamma}_{t-1}), \tag{32}$$

where $\mu_t = E(\varepsilon_t)$, $\boldsymbol{\Sigma}_t = \text{var}(\varepsilon_t)$, and $\Gamma_t = \text{var}(\gamma_t)$. Using the relationship (also in Chapter 2) that $\mathbf{Q}_t = \mathbf{D}^{-1}(\mathbf{P}_t - \boldsymbol{\beta} - \boldsymbol{\gamma}_t)$, we immediately find the transient moments of the queue lengths:

$$E(\mathbf{Q}_t) = \mathbf{D}^{-1}(E(\mathbf{P}_t) - \boldsymbol{\beta}), \text{ and} \tag{33}$$

$$\text{var}(\mathbf{Q}_t) = \mathbf{D}^{-1}(\text{var}(\mathbf{P}_t) - \boldsymbol{\Gamma}_t)(\mathbf{D}^{-1})'. \tag{34}$$

Using these equations, we can track the transient moments in response to any changes to the model parameters, including changes to the input moments, the variances of the production fluctuations, the smoothing parameters, and even the workflow matrix Φ.

For example, calculating the transient behavior of a network, starting from known inventory levels (and known production levels,) is quite simple. Let the known production at time zero be $\mathbf{P}_0$. Then we simply set $E(\mathbf{P}_0) = \mathbf{P}_0$ and $\text{var}(\mathbf{P}_0) = \mathbf{0}$, and find the transient production moments in successive periods by iterating (31) and (32). The transient moments of the queue lengths follow immediately from (33) and (34).

In addition to the transient moments in individual periods, we may track the moments of aggregate production across multiple periods. The expected production of multiple periods is the sum of the expected production in those periods; we have:

$$E\left(\sum_{t=1}^{T}\mathbf{P}_t\right)=\sum_{t=1}^{T}E(\mathbf{P}_t)\,. \tag{35}$$

The variance of the production of multiple periods is more complicated. The production variances are correlated across periods, so the variance across multiple periods is not the sum of the variances. Instead, we take advantage of the fact that the work arrivals and the production fluctuations are independent across periods. For the sake of simplicity, assume that the production fluctuations (the γ_t's) are zero as in the original Tactical Planning Model. Then, iterating (30), we have that:

$$\begin{aligned}
&\mathbf{P}_1=\mathbf{B}\mathbf{P}_0+\mathbf{D}\boldsymbol{\varepsilon}_1,\\
&\mathbf{P}_2=\mathbf{B}\mathbf{P}_0+\mathbf{B}\mathbf{D}\boldsymbol{\varepsilon}_1+\mathbf{D}\boldsymbol{\varepsilon}_2,\\
&\vdots\\
&\mathbf{P}_T=\mathbf{B}\mathbf{P}_0+\mathbf{B}^{T-1}\mathbf{D}\boldsymbol{\varepsilon}_1+\mathbf{B}^{T-2}\mathbf{D}\boldsymbol{\varepsilon}_2+\ldots+\mathbf{D}\boldsymbol{\varepsilon}_T.
\end{aligned} \tag{36}$$

Then, if we add the expressions for $\mathbf{P}_1$ through $\mathbf{P}_T$ together, group the results by the ε_t's, and take the variance of each term, we find:

$$\mathrm{var}\left(\sum_{t=1}^{T}\mathbf{P}_t\right)=\sum_{t=1}^{T}\left(\mathbf{I}+\ldots+\mathbf{B}^{t-1}\right)\mathbf{D}\boldsymbol{\Sigma}_t\mathbf{D}'\left(\mathbf{I}+\ldots+\mathbf{B}^{t-1}\right)'. \tag{37}$$

3.2 Example: Transient Behavior of a Job Shop Producing Spindles

Graves (1986) considered a Tactical Planning Model of a job shop that produces spindles for ultra-precision grinding machines. We consider the transient behavior of a variant of this job shop, with parameters given below. In this shop, new orders for spindles enter the first work station, and completed spindles exit the tenth work station.

Workflow Matrix Φ'	From Work Station									
To Work Station	1	2	3	4	5	6	7	8	9	10
1 (lathe)			0.11		0.68					
2 (copy lathe)	0.15									
3 (drill press)	0.04	0.01		0.71		0.6			0.07	
4 (milling)	0.01	0.41								
5 (rough grinder)	0.03	0.37	1.36							
6 (internal grinder)	0.24				0.15				0.13	
7 (thread cutting)					0.10					
8 (hole abrading)	0.01					0.22	1.00			
9 (precision grinder)								3.43		
10 (ultra-precision grinder)									1.16	
Input μ_i	Varies									
Covariance matrix Σ_{ii}	Varies	0.01	0.01	0.01	0.04	0.04	0	0.01	0.04	0.04

Lead time $1/\alpha_i$	4	1	1	1	1	2	1	1	3	3

Arrivals to the first station are assumed to come from an exponential distribution, so that the expectation and standard deviations of the arrivals are always the same.

We track the behavior of the model in two scenarios. In the first scenario, the orders for grinders grow linearly. The shop is empty at time zero. In period one, the expected arrival to station 1 is 4 hours; in successive periods, the expected arrivals to station 1 increase by 0.2 hours.

Figure 34 charts the growth in the expected arrivals to the first station over 100 periods. It also tracks the expected production at the last station, along with 2-sigma confidence intervals for the production at the last station. (The lower-bound intervals are set to zero if $E(P_{10}) - 2\sigma_{10} < 0$.)

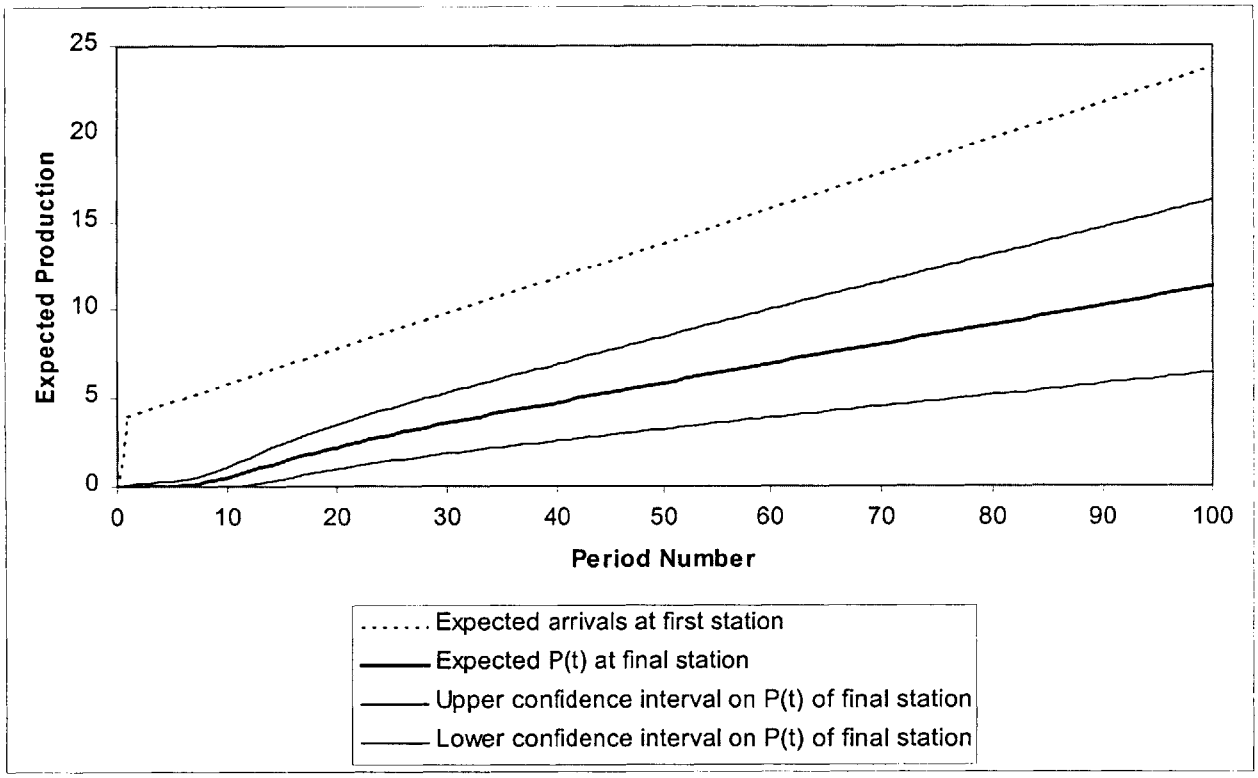


Figure 34 -- Transient Moments of Grinder Production With Linear Order Growth

After an initial ramp-up, the moments of production at the tenth station grow linearly in response to increasing arrivals at the first station.

For reference, the following table shows the moments of production for all stations across the entire 100 periods.

Station	Expected Aggregate Production	Standard Deviation of Production
1 (lathe)	1600.3	171.1
2 (copy lathe)	235.8	25.4
3 (drill press)	207.6	22.4

4 (milling)	110.6	12
5 (rough grinder)	409.9	44.4
6 (internal grinder)	495.5	53.4
7 (thread cutting)	40.2	4.4
8 (hole abrading)	162.2	17.6
9 (precision grinder)	525.9	57.7
10 (ultra-precision grinder)	575.7	63.9

In the second scenario, the distributions of arrivals to the first station vary randomly every period. In particular, the expected arrival in each period comes from a uniform distribution between 0 hours and 40 hours.

Figure 35 charts the expected arrivals to the first station over 100 periods. It also tracks the expected production at the last station, along with 2-sigma confidence intervals for the production at the last station.

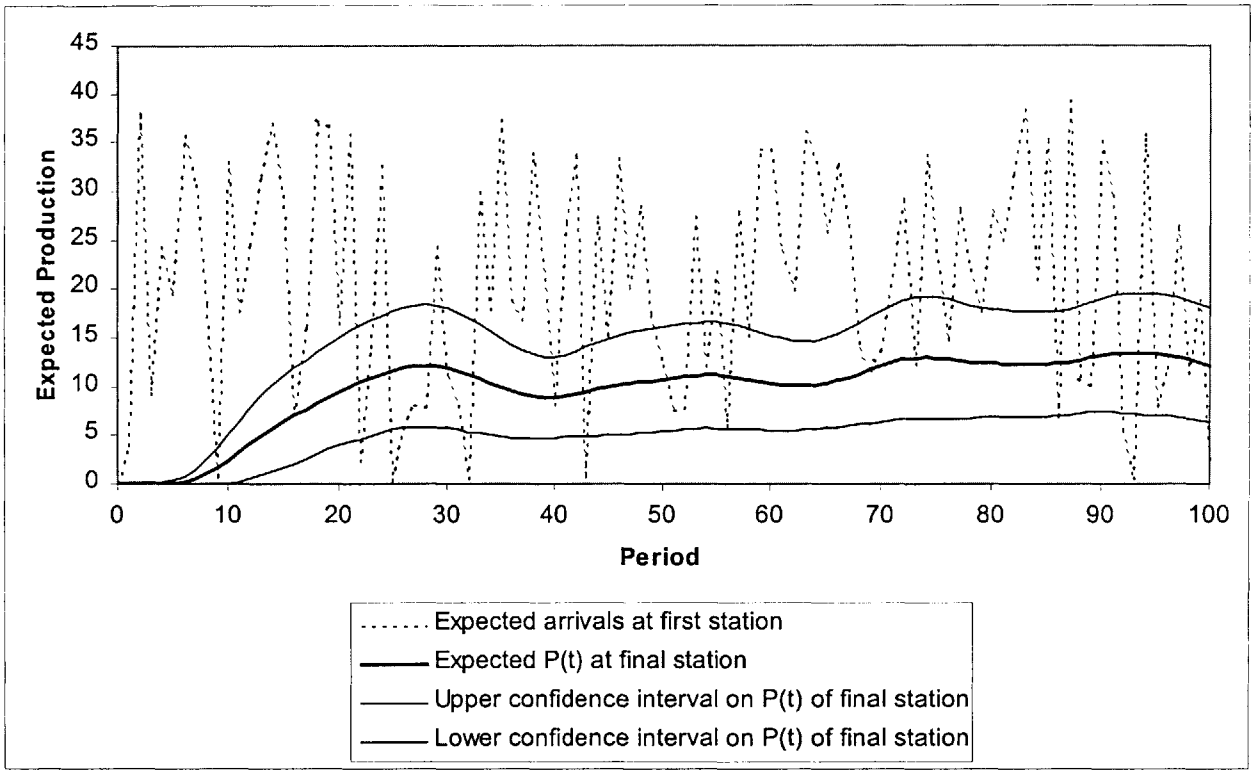


Figure 35 -- Transient Moments of Grinder Production With Random Expected Orders

As with the first scenario, the tenth station has an initial ramp-up period. After this period, the moments of production at the tenth station resemble a moving average of the arrival stream to the first station. This is not surprising, given that one of the major motivations of LLS-MR models is production smoothing.

For reference, the following table shows the moments of production for all stations across the entire 100 periods.

Station	Expected Aggregate Production	Standard Deviation of Production
1 (lathe)	2505.6	287.0
2 (copy lathe)	373.0	42.9
3 (drill press)	335.6	38.9
4 (milling)	176.4	20.4
5 (rough grinder)	663.3	76.9
6 (internal grinder)	800.4	92.7
7 (thread cutting)	65.7	7.7
8 (hole abrading)	264.3	30.7
9 (precision grinder)	878.0	103.3
10 (ultra-precision grinder)	982.9	117.0

We can draw several conclusions from these examples. First, we have demonstrated that we can calculate the transient production moments for LLS-MR models under a wide array of network changes, As shown, these changes can be extremely complicated, such as the second example's random changes to the work arrival distributions. We have also calculated the moments of aggregate production across a number of periods.

Next, we see that the numerical results are consistent with the analytic formulas given in the previous section. In the first example, expected production at the tenth station experienced a transient "ramp-up" period, and then grew linearly, matching the analytic formulas in Section 2.4. Further, the production standard deviations also grew linearly with the arrival standard deviations. In the second example, expected production also experienced a "ramp-up" period, and then acted as a moving average of the arrival distribution changes. This behavior is similar to the analytic formulas for expected production in networks with oscillating work arrivals, given in Section 2.5.

Finally, the results demonstrate the value of LLS-MR models in smoothing production variations. In the second example, despite substantial changes to the arrival distributions, the production distribution at the last station remained remarkably constant.

4. Optimal Control of MR Models

In this section, we consider the optimal control of MR models, using dynamic programming. We find a sequence of optimal control rules (or, equivalently, finding a sequence of optimal production quantities) that minimize the expectation of a quadratic cost function of the production and queue levels over the next T periods. As an example, we might consider minimizing a weighted sum of the variances of production and inventory levels over the next T periods. This gives rise to a *rolling-horizon* control problem; in each period, we generate the control rule for each period by computing the rule that minimizes the variances over the next T periods. Critically, we find that the optimal control rules are linear functions of the queue lengths; thus, an "optimal" MR model will always be a LLS-MR model. These results provide an important justification for LLS-MR models.

4.1 Rule Development

Most of the following development is based on Bertsekas (1995a, pp. 130-132). Suppose we have the following quadratic cost function of the queue lengths and production quantities over the next T periods:

$$\min_{P_t} E\left\{\mathbf{Q_T}'\mathbf{Y}\mathbf{Q_T} + \mathbf{y}'\mathbf{Q_T} + c + \sum_{t=0}^{T-1}\left(\left(\mathbf{Q_t}'\mathbf{Y}\mathbf{Q_t} + \mathbf{y}'\mathbf{Q_t} + c\right) + \left(\mathbf{P_t}'\mathbf{Z}\mathbf{P_t} + \mathbf{z}'\mathbf{P_t} + d\right)\right)\right\} \tag{38}$$

Here, $\mathbf{Y}$ and $\mathbf{Z}$ are constant matrices, $\mathbf{y}$ and $\mathbf{z}$ are constant vectors, and $\mathbf{c}$ and $\mathbf{d}$ are constants. We assume that the quadratic cost functions are convex, so that $\mathbf{Y}$ and $\mathbf{Z}$ are positive definite symmetric matrices. As an example quadratic cost function, note that:

$$\min_{P_t} E\left\{\left(\mathbf{Q_T} - \mathbf{Q}^*\right)' \mathbf{Y}\left(\mathbf{Q_T} - \mathbf{Q}^*\right) + \sum_{t=0}^{T-1}\left(\left(\mathbf{Q}_t - \mathbf{Q}^*\right)' \mathbf{Y}\left(\mathbf{Q}_t - \mathbf{Q}^*\right) + \left(\mathbf{P}_t - E(\mathbf{P})\right)' \mathbf{Z}\left(\mathbf{P}_t - E(\mathbf{P})\right)\right)\right\} \tag{39}$$

minimizes a weighted sum of the production variances and the variances of the queue lengths around target levels. When multiplied out, (39) has the same form as (38).

For the sake of simplicity, we restrict our discussion to models in which there are no production fluctuation terms. Rather than have $\mathbf{P}_t = f(\mathbf{Q}_t) + \gamma_t$, where γ_t is a random variable with zero mean and a finite variance, we assume that $\mathbf{P}_t = f(\mathbf{Q}_t)$. We also assume the LLS-MR work arrival structure, so that $\mathbf{A}_t = \mathbf{\Phi}\mathbf{P}_{t-1} + \boldsymbol{\varepsilon}_{t-1}$, where ε_{t-1} is the vector of random work arrivals with mean μ and covariance matrix Σ. (In chapter 2, we assumed that the vector of random work arrivals appeared at the start of period t; we simply shift the subscript here. This shift simplifies the DP analysis.) Note that we do not assume that production is a linear function of the queue lengths; we will derive that the optimal functions are linear using dynamic programming. Then, using the standard inventory balance equation, we have the following relationship relating $\mathbf{Q}_t$ to $\mathbf{Q}_{t-1}$ and $\mathbf{P}_{t-1}$:

$$\mathbf{Q}_t = \mathbf{Q}_{t-1} + \mathbf{F}\mathbf{P}_{t-1} + \boldsymbol{\varepsilon}_{t-1}, \text{ where } \mathbf{F} = (\mathbf{\Phi} - \mathbf{I}). \tag{40}$$

Using this relationship, our complete dynamic programming problem is:

$$\begin{aligned} &\min_{P_t} E\left\{\mathbf{Q_T}'\mathbf{Y}\mathbf{Q_T} + \mathbf{y}'\mathbf{Q_T} + c + \sum_{t=0}^{T-1}\left(\left(\mathbf{Q_t}'\mathbf{Y}\mathbf{Q_t} + \mathbf{y}'\mathbf{Q_t} + c\right) + \left(\mathbf{P_t}'\mathbf{Z}\mathbf{P_t} + \mathbf{z}'\mathbf{P_t} + d\right)\right)\right\} \\ &s.t. \qquad \mathbf{Q_t} = \mathbf{Q_{t-1}} + \mathbf{F}\mathbf{P_{t-1}} + \boldsymbol{\varepsilon}_{t-1}. \end{aligned} \tag{41}$$

Using the dynamic programming algorithm, we have that:

$$J_T(\mathbf{Q_T}) = \mathbf{Q_T}'\mathbf{Y}\mathbf{Q_T} + \mathbf{y}'\mathbf{Q_T} + c, \tag{42}$$

$$J_t(\mathbf{Q_t}) = \min_{P_t} E\left\{\left(\mathbf{Q}_t'\mathbf{Y}\mathbf{Q}_t + \mathbf{y}'\mathbf{Q}_t + c\right) + \left(\mathbf{P}_t'\mathbf{Z}\mathbf{P}_t + \mathbf{z}'\mathbf{P}_t + d\right) + J_{t+1}\left(\mathbf{Q}_t + \mathbf{F}\mathbf{P}_t + \boldsymbol{\varepsilon}_t\right)\right\}, \tag{43}$$

where the J_t's are the cost-to-go functions. We show that these cost-to-go functions are always quadratic; thus, the control rules are linear functions of the queue lengths. We do by induction.

We first write (43) for $t = T-1$:

$$J_{T-1}(\mathbf{Q}_{T-1}) = \min_{P_t} E\left\{\begin{array}{l}\left(\mathbf{Q}_{\mathbf{T}-1}'\mathbf{Y}\mathbf{Q}_{\mathbf{T}-1} + \mathbf{y}'\mathbf{Q}_{\mathbf{T}-1} + c\right) + \left(\mathbf{P}_{T-1}'\mathbf{Z}\mathbf{P}_{T-1} + \mathbf{z}'\mathbf{P}_{T-1} + d\right) \\ + \left(\mathbf{Q}_{T-1} + \mathbf{F}\mathbf{P}_{T-1} + \boldsymbol{\varepsilon}_{T-1}\right)'\mathbf{Y}\left(\mathbf{Q}_{T-1} + \mathbf{F}\mathbf{P}_{T-1} + \boldsymbol{\varepsilon}_{T-1}\right) + \mathbf{y}'\left(\mathbf{Q}_{T-1} + \mathbf{F}\mathbf{P}_{T-1} + \boldsymbol{\varepsilon}_{T-1}\right) + c\end{array}\right\} \tag{44}$$

Expanding the quadratic form on the right hand side of (44), and taking the expectation of the resulting expression, yields:

$$\begin{aligned} J_{T-1}(\mathbf{Q}_{T-1}) = {} & \left(\mathbf{Q}_{\mathbf{T}-1}'\mathbf{Y}\mathbf{Q}_{\mathbf{T}-1} + \mathbf{y}'\mathbf{Q}_{\mathbf{T}-1} + c\right) \\ & + \min_{\mathbf{P}_{T-1}}\left\{\begin{array}{l}\left(\mathbf{P}_{T-1}'\mathbf{Z}\mathbf{P}_{T-1} + \mathbf{z}'\mathbf{P}_{T-1} + d\right) + \mathbf{Q}_{\mathbf{T}-1}'\mathbf{Y}\mathbf{Q}_{\mathbf{T}-1} + \mathbf{P}_{\mathbf{t}-1}'\mathbf{F}'\mathbf{Y}\mathbf{F}\mathbf{P}_{T-1} + \boldsymbol{\mu}'\mathbf{Y}\boldsymbol{\mu} \\ + 2\mathbf{Q}_{\mathbf{T}-1}'\mathbf{Y}\mathbf{F}\mathbf{P}_{T-1} + 2\boldsymbol{\mu}'\mathbf{Y}\mathbf{F}\mathbf{P}_{T-1} + 2\boldsymbol{\mu}'\mathbf{Y}\mathbf{Q}_{\mathbf{T}-1} \\ + \left(\mathbf{y}'\mathbf{Q}_{T-1} + \mathbf{y}'\mathbf{F}\mathbf{P}_{T-1} + \mathbf{y}'\boldsymbol{\mu} + c\right)\end{array}\right\} \end{aligned} \tag{45}$$

As noted, all the quadratic functions are convex, so we can find the optimal $\mathbf{P}_{\mathbf{T}-1}$ by differentiating (45) with respect to $\mathbf{P}_{\mathbf{T}-1}$, and setting the resulting expression to $\mathbf{0}$. Doing so yields:

$$0 = \left(2\mathbf{Z} + 2\mathbf{F}'\mathbf{Y}\mathbf{F}\right)\mathbf{P}_{T-1} + 2\mathbf{F}'\mathbf{Y}\mathbf{Q}_{\mathbf{T}-1} + 2\mathbf{F}'\mathbf{Y}\boldsymbol{\mu} + \mathbf{F}'\mathbf{y} + \mathbf{z} \tag{46}$$

The matrix multiplying $\mathbf{P}_{\mathbf{T}-1}$ is positive definite since $\mathbf{Z}$ is positive definite (hence invertible) and $\mathbf{F'YF}$ is positive semidefinite. Then, solving for $\mathbf{P}_{\mathbf{T}-1}$ yields the optimal control rule:

$$\begin{aligned} \mathbf{P}_{T-1}^{*} &= \mathbf{D}_{T-1}\mathbf{Q}_{T-1} + \boldsymbol{\beta}_{T-1}, \text{ where:} \\ \mathbf{D}_{T-1} &= -\left(\mathbf{Z} + \mathbf{F}'\mathbf{Y}\mathbf{F}\right)^{-1}\mathbf{F}'\mathbf{Y}, \text{ and} \\ \boldsymbol{\beta}_{T-1} &= -\left(\mathbf{Z} + \mathbf{F}'\mathbf{Y}\mathbf{F}\right)^{-1}\left(\mathbf{F}'\mathbf{Y}\boldsymbol{\mu} + \mathbf{F}'\mathbf{y}/2 + \mathbf{z}/2\right). \end{aligned} \tag{47}$$

Substituting (47) into the expression for J_{T-1}, we have:

$$J_{T-1}(\mathbf{Q}_{T-1}) = \mathbf{Q}_{T-1}'\mathbf{K}_{T-1}\mathbf{Q}_{T-1} + \mathbf{k}_{T-1}'\mathbf{Q}_{T-1} + C_{T-1}, \tag{48}$$

where $\mathbf{K}_{T-1}$ is a matrix, $\mathbf{k}_{T-1}$ is a vector, and C_{T-1} is a constant such that:

$$\mathbf{K}_{T-1} = \mathbf{Y} + \mathbf{D}_{T-1}'\mathbf{Z}\mathbf{D}_{T-1} + \left(\mathbf{I} + \mathbf{F}\mathbf{D}_{T-1}\right)'\mathbf{Y}\left(\mathbf{I} + \mathbf{F}\mathbf{D}_{T-1}\right), \tag{49}$$

$$\mathbf{k}_{T-1} = \mathbf{y} + 2\boldsymbol{\beta}_{T-1}'\mathbf{Z}\mathbf{D}_{T-1} + \mathbf{z}'\mathbf{D}_{T-1} + 2\left(\mathbf{F}\boldsymbol{\beta}_{T-1} + \boldsymbol{\mu}\right)'\left(\mathbf{I} + \mathbf{F}\mathbf{D}_{T-1}\right) + \mathbf{y}'\left(\mathbf{I} + \mathbf{F}\mathbf{D}_{T-1}\right) \tag{50}$$

$$C_{T-1} = 2\mathbf{c} + \mathbf{d} + \boldsymbol{\beta}'_{T-1}\mathbf{Z}\boldsymbol{\beta}_{T-1} + \mathbf{z}'\boldsymbol{\beta}_{T-1} + \boldsymbol{\beta}'_{T-1}\mathbf{F}'\mathbf{Y}\mathbf{F}\boldsymbol{\beta}_{T-1} + 2\boldsymbol{\beta}'_{T-1}\mathbf{F}'\mathbf{Y}\boldsymbol{\mu} + E\{\boldsymbol{\varepsilon}_{t-1}\mathbf{Y}\boldsymbol{\varepsilon}_{t-1}\} + \mathbf{F}\boldsymbol{\beta}_{T-1} \tag{51}$$

Importantly, $\mathbf{K}_{T-1}$ is a positive definite symmetric matrix, since $\mathbf{Y}$ and $\mathbf{Z}$ are positive definite symmetric matrices. Thus, like $\mathbf{J}_T$, we have that $\mathbf{J}_{T-1}$ is a positive definite quadratic function.

We can iterate the above process to find $\mathbf{J}_{T-2}$, etc. through $\mathbf{J}_0$, which is the optimal control rule for the current period. At each stage, through an analysis identical to that for $t = T - 1$, we find:

$$\begin{aligned} &\mathbf{P}^*_{t-1} = \mathbf{D}_{t-1}\mathbf{Q}_{t-1} + \boldsymbol{\beta}_{t-1}, \text{ where:} \\ &\mathbf{D}_{t-1} = -\left(\mathbf{Z} + \mathbf{F}'\mathbf{K}_t\mathbf{F}\right)^{-1}\mathbf{F}'\mathbf{K}_t, \text{ and} \\ &\boldsymbol{\beta}_{t-1} = -\left(\mathbf{Z} + \mathbf{F}'\mathbf{K}_t\mathbf{F}\right)^{-1}\left(\mathbf{F}'\mathbf{K}_t\boldsymbol{\mu} + \mathbf{k}'_t\mathbf{F}/2 + \mathbf{z}/2\right). \end{aligned} \tag{52}$$

Further, $\mathbf{K}_t$ and $\mathbf{k}_t$ are generated recursively by the following algorithms:

$$\mathbf{K}_{t-1} = \mathbf{Y} + \mathbf{D}'_{t-1}\mathbf{Z}\mathbf{D}_{t-1} + \left(\mathbf{I} + \mathbf{F}\mathbf{D}_{t-1}\right)'\mathbf{K}_t\left(\mathbf{I} + \mathbf{F}\mathbf{D}_{t-1}\right) \tag{53}$$

$$\mathbf{k}_{t-1} = \mathbf{y} + 2\boldsymbol{\beta}'_{t-1}\mathbf{Z}\mathbf{D}_{t-1} + \mathbf{z}'\mathbf{D}_{t-1} + 2\left(\mathbf{F}\boldsymbol{\beta}_{t-1} + \boldsymbol{\mu}\right)'\left(\mathbf{I} + \mathbf{F}\mathbf{D}_{t-1}\right) + \mathbf{k}'_t\left(\mathbf{I} + \mathbf{F}\mathbf{D}_{t-1}\right) \tag{54}$$

4.2 Discussion of Optimal Control Rules

Using the above dynamic programming formulation should be fairly easy to use in practice. At the start of each time period t, we calculate $\mathbf{D}_t$ and β_t. Then, we find the optimal production using the formula $\mathbf{P}^*_t = \mathbf{D}_t\mathbf{Q}_t + \boldsymbol{\beta}_t$. Bertsekas (1995a, pp. 133-141) shows that the $\mathbf{K}_t$'s have spectral radii less than one, guaranteeing that the resulting system will be stable.

Note that if none of the parameters involved in calculating $\mathbf{D}_t$ and β_t ever change, and we always look ahead the same number of periods, $\mathbf{D}_t$ and β_t do not change from period to period (i.e. we only have to calculate them once). Further, if $\mathbf{D}_t$ and β_t remain constant, we can immediately find the steady-state expected queue lengths, as well. We have the recursion relationship $\mathbf{Q}_t = \mathbf{Q}_{t-1} + \mathbf{F}\mathbf{P}_{t-1} + \boldsymbol{\varepsilon}_{t-1}$, where $\mathbf{F} = \left(\boldsymbol{\Phi} - \mathbf{I}\right)$, and $\mathbf{P}_{t-1} = \mathbf{D}\mathbf{Q}_{t-1} + \boldsymbol{\beta}$. Iterating this recursion equation infinitely, and taking the expectation of both sides, yields:

$$E(\mathbf{Q}) = \sum_{s=0}^{\infty}(\mathbf{I} + \mathbf{F}\mathbf{D})^s(\mathbf{D}\boldsymbol{\beta} + \boldsymbol{\mu}) = \left(-\mathbf{F}\mathbf{D}\right)^{-1}(\mathbf{D}\boldsymbol{\beta} + \boldsymbol{\mu}). \tag{55}$$

The preceding analysis yields the optimal production quantities when production quantities do fluctuate randomly (i.e. $\mathbf{P}_t = f(\mathbf{Q}_t) + \boldsymbol{\gamma}_t$). Since the means of these fluctuations are assumed to be $\mathbf{0}$, all terms with both γ_t and $\mathbf{P}_t$ drop out of the J_t formulas, which results in the same formulas for $\mathbf{D}_t$ and β_t

given above. The expected costs will be higher, however, since the J_t formulas will contain $E(\gamma_t'\mathbf{Z}\gamma_t)$ terms. (This is not surprising, considering that the γ_t terms increase the production variances.)

For the sake of simplicity, in the above analysis we assumed that only the queue levels and production quantities change over time (due to random fluctuations). However, one can calculate the optimal production quantities given any known changes to the control parameters over the next T periods, including changes to the cost matrices (**Y**, **y**, **Z**, and **z**), changes to flow routing (i.e. changes to **F**), and changes in the pattern of expected arrivals (μ). The cost, however, is that the **D** and β matrices will need to be completely recalculated over most periods. In addition, E(**Q**) cannot be calculated analytically using (55) when model parameters change over time.

We can use an analysis similar to the above to find optimal policies for *discounted-horizon* problems. In this class of problems, the objective function costs decrease by a factor of α in each successive period, where $0 < \alpha < 1$. These problems reflect the idea that analysts care more about short-term costs than long term costs, given that the state of the network many periods from now is largely unknown, and given the time value of money. The discounted-horizon problem is:

$$\min_{P_t} E\left\{\alpha^T\left(\mathbf{Q}_T'\mathbf{Y}\mathbf{Q}_T + \mathbf{y}'\mathbf{Q}_T\right) + c + \sum_{t=0}^{T-1}\left(\alpha^t\left(\mathbf{Q}_t'\mathbf{Y}\mathbf{Q}_t + \mathbf{y}'\mathbf{Q}_t + c\right) + \alpha^t\left(\mathbf{P}_t'\mathbf{Z}\mathbf{P}_t + \mathbf{z}'\mathbf{P}_t + d\right)\right)\right\} \qquad (56)$$
$$s.t. \qquad \mathbf{Q}_t = \mathbf{Q}_{t-1} + \mathbf{F}\mathbf{P}_{t-1} + \boldsymbol{\varepsilon}_{t-1}.$$

Through an analysis similar to that used for the undiscounted problem, we find that at each state the optimal policy is:

$$\begin{aligned} &\mathbf{P}_{t-1}^* = \mathbf{D}_{t-1}\mathbf{Q}_{t-1} + \boldsymbol{\beta}_{t-1}, \text{ where:} \\ &\mathbf{D}_{t-1} = -\alpha\left(\mathbf{Z} + \alpha\mathbf{F}'\mathbf{K}_t\mathbf{F}\right)^{-1}\mathbf{F}'\mathbf{K}_t, \text{ and} \\ &\boldsymbol{\beta}_{t-1} = -\alpha\left(\mathbf{Z} + \alpha\mathbf{F}'\mathbf{K}_t\mathbf{F}\right)^{-1}\left(\mathbf{F}'\mathbf{K}_t\boldsymbol{\mu} + \mathbf{k}_t'\mathbf{F}/2 + \mathbf{z}/2\right). \end{aligned} \qquad (57)$$

Further, $\mathbf{K}_t$ and $\mathbf{k}_t$ are generated recursively by the following algorithms:

$$\mathbf{K}_{t-1} = \mathbf{Y} + \mathbf{D}_{t-1}'\mathbf{Z}\mathbf{D}_{t-1} + \alpha\left[\left(\mathbf{I} + \mathbf{F}\mathbf{D}_{t-1}\right)'\mathbf{K}_t\left(\mathbf{I} + \mathbf{F}\mathbf{D}_{t-1}\right)\right]. \qquad (58)$$

$$\mathbf{k}_{t-1} = \mathbf{y} + 2\boldsymbol{\beta}_{t-1}'\mathbf{Z}\mathbf{D}_{t-1} + \mathbf{z}'\mathbf{D}_{t-1} + \alpha\left[2\left(\mathbf{F}\boldsymbol{\beta}_{t-1} + \boldsymbol{\mu}\right)'\left(\mathbf{I} + \mathbf{F}\mathbf{D}_{t-1}\right) + \mathbf{k}_t'\left(\mathbf{I} + \mathbf{F}\mathbf{D}_{t-1}\right)\right]. \qquad (59)$$

As in the undiscounted case, Bertsekas (1995b, pp. 150-152) shows that the $\mathbf{K}_t$'s have spectral radii less than one, guaranteeing that the resulting system will be stable.

There is one important drawback to finding the optimal policies using dynamic programming. If one looks at the dynamic programming formulation, one sees that there are no restrictions on either the

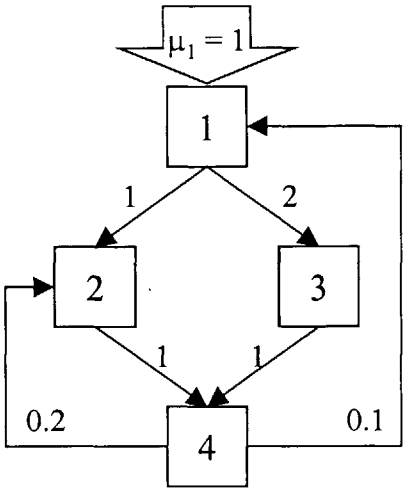
production quantities or queue-length quantities being negative in any particular period. As a result, situations may arise in which a control rule either tells a station to produce a negative amount or produce more than a station has in its queue.

This problem is common to LLS-MR models with production rules that are functions of the work in more than one queue. In Chapter 2, for instance, we considered models using the Denardo-Tang rules; recall these rules tell stations to produce an amount needed to counteract inventory fluctuations at downstream stations. If a Denardo-Tang shop is short of inventory at downstream stations, the rule may tell a station to produce more than it has in its own queue. Conversely, if the shop has a large excess of inventory at downstream stations, the rule may tell a station to produce a negative amount of work.

Thus, in practice, we might use truncated policies. Should a particular control rule tell us to produce a negative amount at a station, we produce nothing at that station; similarly, should a control rule tell us to produce more than is in the queue we process the queue. Over time, these truncated policies should produce the desired equilibrium, as more work arrives to the station. Alternately, there are environments in which it may be possible to add work to a station's queue or process more than a station has in its queue. The former might correspond to introducing additional work orders to the shop, while the latter might correspond to stations having access to additional parts depots.

4.3 An Optimal Control Example

Figure 36 shows an example four-station network. Station 1 receives an average of one unit of work each period, and completed work exits from station 4; the workflow (Φ_{ij}) entries are shown in the figure. Figure 36 also shows the network's expected production.



Station	E(P)
1	1.6
2	2.8
3	3.2
4	6.0

Figure 36 -- A Four-Station Network

We are to solve an undiscounted rolling horizon problem. The objective in each period is to minimize a sum of the production variances and a sum of the queue-length variances around the smallest possible expected inventory levels (ideally, we would like E(**Q**) = E(**P**)). The resulting dynamic programming problem is:

$$\min_{P_t} E\left\{\begin{array}{l}\left(\mathbf{Q}_T - E(\mathbf{P})\right)' \mathbf{I}\left(\mathbf{Q}_T - E(\mathbf{P})\right) \\ +\sum_{t=0}^{T-1}\left(\left(\mathbf{Q}_T - E(\mathbf{P})\right)' \mathbf{I}\left(\mathbf{Q}_T - E(\mathbf{P})\right) + \left(\mathbf{P}_t - E(\mathbf{P})\right)' \mathbf{I}\left(\mathbf{P}_t - E(\mathbf{P})\right)\right)\end{array}\right\}, \qquad (60)$$

$$s.t. \quad \mathbf{Q}_t = \mathbf{Q}_{t-1} + \mathbf{F}\mathbf{P}_{t-1} + \boldsymbol{\varepsilon}_{t-1}.$$

which, when written in the form of equation (41), becomes

$$\min_{P_t} E\left\{\begin{array}{l}\mathbf{Q}_T' \mathbf{I} \mathbf{Q}_T - 2E(\mathbf{P})'\mathbf{Q}_T + E(\mathbf{P})'E(\mathbf{P}) \\ +\sum_{t=0}^{T-1}\left(\left(\mathbf{Q}_t' \mathbf{I} \mathbf{Q}_t - 2E(\mathbf{P})'\mathbf{Q}_t + E(\mathbf{P})'E(\mathbf{P})\right) + \left(\mathbf{P}_t' \mathbf{I} \mathbf{P}_t - 2E(\mathbf{P})'\mathbf{P}_t + E(\mathbf{P})'E(\mathbf{P})\right)\right)\end{array}\right\} \qquad (61)$$

$$s.t. \quad \mathbf{Q}_t = \mathbf{Q}_{t-1} + \mathbf{F}\mathbf{P}_{t-1} + \boldsymbol{\varepsilon}_{t-1}.$$

The following table compares the optimal **D**, β, and steady-state E(**Q**) when we look ahead one period, and when we look ahead an infinite number of periods (here, "infinite" is approximated by looking ahead 500 periods).

One-Period Results						Infinite-Period Results					
D				β	E(Q)	D				β	E(Q)
0.206	-0.164	-0.231	-0.110	3.128	1.600	0.323	-0.130	-0.191	-0.079	-0.375	16.701
0.054	0.437	-0.164	-0.161	2.982	2.800	0.292	0.528	-0.093	-0.092	-4.604	8.058
0.174	-0.276	0.285	-0.273	4.416	3.200	0.432	-0.209	0.376	-0.208	-3.196	8.084
0.088	0.049	0.068	0.303	3.686	6.000	0.600	0.217	0.234	0.444	-12.326	10.493

The one-period β is much larger than the infinite-period β (the latter has the station add work to the queue rather than subtract it). Further, the one-period expected queue lengths are significantly smaller than the infinite-period queue lengths. Indeed, the one period E(**Q**) exactly equals E(**P**), implying that minimizing production fluctuations was significantly less important than minimizing queue-length fluctuations in the one-period problem. These results imply that the more periods are added to the horizon, the more significant minimizing production fluctuations becomes in comparison to minimizing queue-length fluctuations.

It should be noted that the one-period results are fundamentally no more or less optimal than the infinite period results; both are optimal solutions to their respective problems. The very different natures of the solutions, though, illustrate the importance of determining the correct objective function for a particular network.

We can quantify the relative change in importance between minimizing queue-length and production fluctuations. Figure 37 graphs the steady-state total expected inventory, $\sum_i E(Q_i)$, against the number of periods included in the objective function.

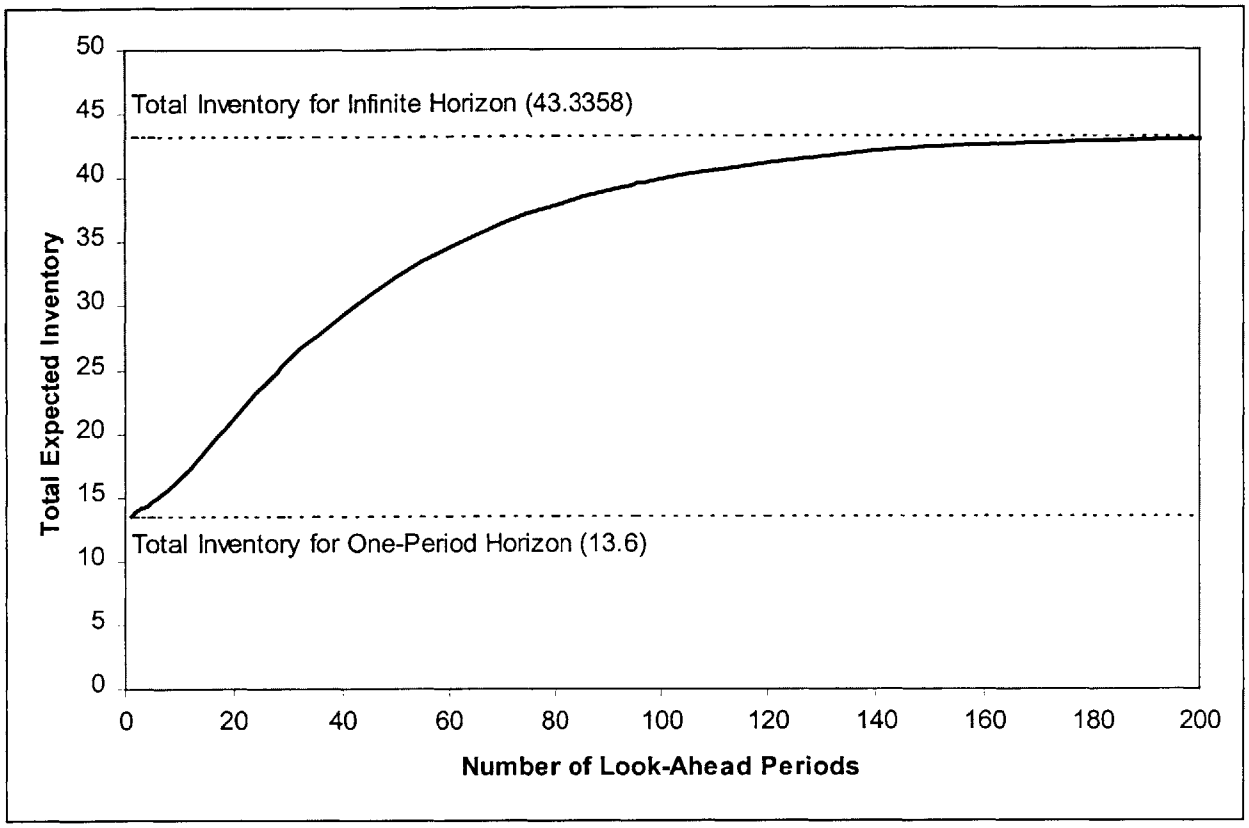


Figure 37 -- Optimal Expected Inventory v. Look-Ahead Periods

In this case, as the number of periods in the horizon increases, the total expected inventory corresponding to the optimal control rules increases in accordance with an S-shaped curve.

Intuitively, for a fairly small number of look-ahead periods, the cost of minimizing queue-length fluctuations tends to outweigh the costs of minimizing production fluctuations by a significant amount. A possible explanation for this effect comes from the formulation of the objective functions. If we look at the formulation, (41) we see that the costs of queue-length fluctuations are considered one additional time compared to production fluctuations. In particular, we account for queue-length fluctuations at time T, but not production fluctuations.

As the number of periods increases, however, the costs of the production fluctuations required to minimize queue-length fluctuations grow, and grow faster than the costs of the queue length fluctuations. This effect causes the optimal policy to change from one of minimizing queue-length fluctuations by keeping queue lengths low (and production fluctuations comparatively high), to one of minimizing

production fluctuations by keeping queue lengths comparatively high. The difference between the growth rates explains the geometric growth in expected inventories shown on the first part of the curve.

Recall that Bertsekas (1995a, pp. 133-141) showed that the class of problems including the example will be stable, such that the optimal policy will converge to a single policy as the number of look-ahead periods approaches infinity. This convergence is shown in the second part of the curve. The costs of production fluctuations are still growing faster than the costs of queue-length fluctuations, but the resulting policies are now converging asymptotically to the infinite horizon solution (with expected inventory equal to 43.3358).

Chapter 8: Conclusions

1. Contributions

In this final chapter, we review the contributions of this thesis. We also present some opportunities for future research.

First, and importantly, we have defined the class of MR models. Previously, work on this class of models had been limited to the Tactical Planning Model (TPM) and direct extensions to the TPM. Consequently, all research assumed that production is a fixed fraction of work-in-queue, and that work arrivals are stationary fluid arrivals. We have shown that the TPM is one model in a much larger class of models that may all be analyzed through similar techniques.

We began by generalizing the current TPM to include linear control rules of multiple queues along with production fluctuation terms. We showed how this larger subclass (LLS-MR) encompasses a variety of common network and job-shop control techniques, including pull-based systems such as Kanban networks, Basestock networks, and more complicated variance suppression rules, such as the proportional restoration rules of Denardo and Tang (1997).

Next, we demonstrated how to calculate approximations for the steady-state moments of MR-models with general (i.e. nonlinear) control rules (class GLS-MR). This technique allows for the modeling of a wide range of realistic machine and human behavior, including machine congestion, machine overloading, and the effects of overtime work. The drawback, however, is that the results are approximations. As shown in Chapter 3, there are situations (usually, very heavily loaded networks with a great deal of random noise) in which the approximations are inadequate. Further, the applicability of the approximations to transient analysis is limited, since estimating the transient behavior of a network with general control rules involves taking approximations of approximations. Nevertheless, GLS-MR approximations are quite accurate over the range of networks one generally finds (i.e. reasonably well-behaved networks), and opens a wide range of steady-state optimization and decision-support applications for networks whose stations have nonlinear production functions. We expect such networks (and applications) are quite common in practice, given that they incorporate realistic human behavior and machine behavior. Thus, the ability to analyze GLS-MR models is quite significant.

We then developed MR-models that maintain a weighted constant inventory constraint (class LLC-MR). This subclass permits models of shops obeying common control rules, such as CONWIP (c.f. Hopp and Spearman, 1996) and Drum-Buffer-Rope (c.f. Goldratt, 1986). In fact, this subclass of MR-models generalizes the conventional CONWIP rules, producing job shops that have less variability, and require less capacity, than shops using simple CONWIP or constant-release policies.

Next, we developed a class of MR-models that process discrete jobs (class LRS-MR). In these models, downstream work arrivals depend on completed jobs, often in complicated, probabilistic ways, not on completed fluid work. This class allows the modeling of environments in which jobs have greatly differing processing requirements, and relationships between jobs completed upstream and jobs completed downstream are given by complicated probabilistic relationships. This ability is especially important in environments such as data processing, where the completion of one upstream job may randomly trigger cascades of downstream jobs, and where the processing times of the jobs may differ by orders of magnitude.

We then showed how to set up and solve steady-state optimization problems related to MR-models, including maximum performance, maximum throughput, and minimum cost problems. In doing so, we showed how to model capacity requirements for models using linear and general control rules. Further, our example problems showed that cost-performance tradeoff curves (resulting from solving a set of optimization problems with varying constraints) often indicated a set of solutions that provided good performance at a reasonable cost.

Finally, we used the underlying recursion equations to find the transient behavior of MR-models. For simple network changes, we found analytic expressions for the transients, extending the work of Parrish (1987). We also demonstrated using the MR recursion equations directly to numerically calculate transient behavior with respect to a very broad class of changes, including changes to mean arrivals, arrival variances, routing flows, and smoothing parameters. We also found optimal control rules given a multi-period quadratic objective function, by using a dynamic programming formulation. Importantly, we found that the optimal policies were always linear functions of the work in queue – a key justification for the use of linear control rules.

2. Opportunities for Future Research

The following opportunities for future research apply to all MR models. Possibilities for extensions to particular classes of MR-models have been discussed in the corresponding chapters.

2.1 Modeling Multiple Work Flows and History-Dependent Work Flows

All of the MR-models have the disadvantage that they require Markov assumptions for the work flows. They assume that work flows in the network can be modeled such that accounting for the history of the work flow is not necessary. In the general case, it does not seem possible to relax the Markov assumptions without having models with combinatorial complexity. However, there are some limited generalizations for history-dependent work flows that are worth further study.

First, assume that the flows through the network may be decomposed into flows corresponding to a few distinct job types, each with its own routings and production requirements. Further, assume that stations can process jobs of each type each period independently (i.e. a station processes a fixed fraction of the queue of type-one jobs each period, regardless of how many jobs of other types there are). Then we can model the different job flows by having one MR model for each job flow. Provided the job flows are independent, the total expected loads and load variances on each station would be the sum of the statistics from the individual MR models.

In an alternate scenario, assume that the history dependencies are defined by a small number of sequential states (for example, work goes from station A to station B, then back to station A for reprocessing), we can create an MR model in each state as one model node. Again assuming that the stations can handle the work in each state through separate queues, the expected workload at a station is the sum of the workloads at the corresponding state nodes. The workload variance is the sum of the variances at the corresponding state nodes, plus the sum of the covariances between these state nodes.

Note that the above scenarios assume that jobs of different types, or in different states can be processed independently. In practice, this often will not be the case, as a shared station might use a *batch-processing* rule, instead. Under batch processing, the station processes an amount of work based on all the jobs currently in queue for it, regardless of the jobs' type or state. Thus, an important area of research would be to model batch processing; doing so has two requirements. First, the model must accurately assess the total load on the shared station resulting from processing multiple types of work. Second, the model must accurately depict the work flows of each type and state coming out of the station. We have done some preliminary work in this area, and expect that modeling batch processing is feasible. The batch-processing policies would be modeled through general control rules of multiple queues.

2.2 Decomposition Approaches for Large Models

All of the example MR-models considered in this thesis have been fairly small (the largest model had eighteen stations). However, there are a number of applications for MR-models which have very large numbers of stations. For example, we consider manufacturing or data-processing applications in which there are many physical stations or processing nodes, many job types, and many processing steps per job type. Modeling such an environment may require thousands of stations.

The overall complexity required to generate the steady-state moments of an MR-model is $O(n^3)$ where n is the number of stations. While this complexity is excellent for small to mid-sized networks, it is impractical for a network with thousands of stations – especially if we want to optimize the network, using the MR-model as an optimization subroutine.

Consequently, an important area of research would be methods to decompose the calculations required to analyze MR-models. Such decomposition techniques would be quite useful for "sparse" networks (networks with many stations, but comparatively few flows between them.)

2.3 Period-Sizing and Discrete Control Networks

One of the fundamental assumptions of MR networks is that they are discrete-time networks. Work transfers between stations occur only at fixed intervals (between periods). This a reasonable assumption for manufacturing plants with distinct work shifts, or other facilities involving people completing work in distinct shifts. However, there are many environments where this assumption does not apply.

In many manufacturing facilities, job components shuttle between stations continuously, not just at shift breaks. Graves (1986) suggests that MR models (the TPM, at that time) still applies to such an environment if the periods are sized carefully. He suggests the periods be large enough that a significant amount of work gets done per period, but small enough that a job is unlikely to travel through more than one station in a particular period. Additional research on sizing periods within real job shops, expanding on Graves' ideas, would be valuable.

Beyond manufacturing facilities, we might consider the application of MR-models to data processing systems. In these systems, streams of computing jobs enter and leave the network continuously. Further, the computing requirements for particular jobs may vary by several orders of magnitude, making a discrete-period assumption unrealistic. Instead, we assume that the network is a *discrete-control network*. At regular intervals, we make decisions about the capacity to allocate between the different work flows, presumably in accordance with the production quantities suggested by an MR model. To use an MR model in this setting, we will need to model the fact that particular jobs may travel to one of several stations, or leave the system, during the next period (defined by the interval between successive control decisions). We would need to calculate the distributions for where particular jobs are likely to be at the decision points (these calculations would probably be approximations). Developing such a model would be extremely valuable, as it would allow MR-models to be applied to sophisticated data-processing systems.

2.4 Continuous-Time Models

Another way to address the discrete-time issues would be to create a continuous-time MR model. Intuitively, one might consider shrinking the period lengths to zero. Doing so creates a set of equations that look like a system of linear differential equations with random process terms for new work arrivals.

Analyzing the resulting set, however, appears extremely difficult. The natural choice for the random process terms – reflected Brownian Motion (RBM) processes – are by definition non-

differentiable everywhere. Further, the analysis of even simple, individual queues with RBM arrivals is quite complicated (ex. Karandikar and Kulkarni, 1995). Thus, it is not clear whether the creation of a continuous-time MR model is possible. The creation of such a model, however, would be quite valuable for theoretical and practical applications.

References

Abate J., and W. Whitt. 1988. Transient Behaviour of the M/M/1 Queue via Laplace Transforms. *AAP*, **20**, 4, 145-178.

Asmussen, S. 1995. Stationary Distributions for Fluid Flow Models With or Without Brownian Noise. *Stochastic Models*, **11**, 21-49.

Baker, K. 1993. "Requirements Planning." *Logistics of Production and Inventory,* S. Graves, A. H. G. Rinnooy Kan and P. Zipkin (eds.), *Handbook in Operations Research and Management Science, Vol. 4*, North Holland.

Baskett, F., K. M. Chandy, R. R. Muntz, and F. Palacios. 1975. Open, Closed and Mixed Networks of Queues with Different Classes of Customers. *J. ACM*, **22**, 248-260.

Bertsekas, Dimitri P. 1982. "Projected Newton Methods for Optimization Problems with Simple Constraints." *SIAM J. on Control and Optimization*, **20**, 221-246.

Bertsekas, Dimitri P. 1995a. *Dynamic Programming and Stochastic Control*, Vol. 1. Athena Scientific, Cambridge, Mass.

Bertsekas, Dimitri P. 1995b. *Dynamic Programming and Stochastic Control*, Vol. 2. Athena Scientific, Cambridge, Mass.

Bertsekas, Dimitri P. 1995c. *Nonlinear Programming*. Athena Scientific, Belmont, Massachusetts.

Bertsimas, Dimitris J., and Daisuke Nakazato. 1995. The Distributional Little's Law and its Applications. *Operations Research*, **43**, 298-310.

Bitran, G. R., and D. Tirupati. 1988. Multiproduct Queueing Networks with Deterministic Routing: Decomposition Approach and Notion of Inference. *Management Science*, **34**, 75-100.

Bitran, G. R., and D. Tirupati. 1993. "Hierarchical Planning." *Logistics of Production and Inventory,* S. Graves, A. H. G. Rinnooy Kan and P. Zipkin (eds.), *Handbook in Operations Research and Management Science, Vol. 4*, North Holland.

Blazewicz, Jacek. 1994. *Scheduling in Computer and Manufacturing Systems*. Springer-Verlag, New York.

Buzacott, J. A., and J. G. Shantikumar. 1985. Approximate Queueing Models of Dynamic Job Shops. *Management Science*, **31**, 870-887.

Buzacott, J. A., and D. D. Yao. 1986. Flexible Manufacturing Systems: A Review of Analytical Models. *Management Science*, **32**, 890-905.

Cohen, J. W. 1969. *The Single Server Queue*. John Wiley, New York, 1969.

Denardo, Eric V., and Y. S. Lee. 1987. Pulling a Markov Production System: I and II. Working Paper, Department of Operations Research, Yale University, New Haven, CT, 1987.

Denardo, Eric V., and Y. S. Lee. 1992. Linear Control of a Markov Production System. *Operations Research*, **40**, 259-278.

Denardo, Eric V., and Christopher S. Tang. 1997. Control of a Stochastic Production System with Estimated Parameters. *Management Science*, **43**, 1296-1307.

Duenyas, Izak, and Wallace Hopp. 1993. Estimating the Throughput of an Exponential CONWIP Assembly System. *Queueing Systems*, **14**, 135-157.

Duenyas, Izak, Wallace Hopp, and Mark Spearman. 1993. Characterizing the Output Process of a CONWIP Line with Deterministic Processing and Random Outages. *Management Science*, **39**, 975-988.

Duncan, W. R., and J. Nevison. 1996. "Mastering Modern Project Management." Seminar and Lecture Notes. Duncan-Nevison, Lexington.

Fine, Charles H., and Stephen C. Graves. 1989. A Tactical Planning Model for Manufacturing Subcomponents of Mainframe Computers. *J. Mfg. Oper. Mgt.*, **2**, 4-34.

Gallager, Robert G. 1996. *Discrete Stochastic Processes*. Kluwer Academic Publishers, Boston.

Gantmacher, F. R. 1959. *The Theory of Matrices*, Vol. 2. Chelsea Publishing Company, New York.

Goldratt, Eliyahu, M. 1986. *The Goal: A Process of Ongoing Improvement*. North River Press, New York.

Graves, Stephen C. 1986. A Tactical Planning Model for a Job Shop. *Operations Research*, **34**, 522-533.

Graves, Stephen C. 1988a. Safety Stocks in Manufacturing Systems. *J. Mfg. Oper. Mgt.*, **1**, 67-101.

Graves, Stephen C. 1988b. Determining the Spares and Staffing Levels for a Repair Depot. *J. Mfg. Oper. Mgt.*, **1**, 227-241.

Graves, Stephen C. 1988c. Extensions to a Tactical Planning Model for a Job Shop. Proceedings of the 27th IEEE Conference on Decision and Control, Austin, Texas, December 1988.

Gstettner, S., and H. Kuhn. 1996. Analysis of Production Control Systems Kanban and CONWIP. *International Journal of Production Research*, **34**, 3253-3273.

Harrison, J. M. 1988. Brownian Models of Queueing Networks with Heterogeneous Customer Populations. In *Stochastic Differential Systems, Stochastic Control Theory, and Applications*, W. Fleming and P. L. Lions (eds.). IMA Volume **10**, Springer-Verlag, New York, 147-186.

Harrison, J. M., and R. J. Williams. 1987. Brownian Models of Open Queueing Networks with Homogeneous Customer Populations. *Stochastics*, **22**, 77-115.

Hax, A.C. 1978. "Aggregate Production Planning." *Handbook of Operations Research, Vol. 2*, J. Moder and S. E. Elmaghraby (eds.), Von Nostrand, Reinhold.

Herer, Y.T., and M. Masin. 1997. Mathematical Programming Formulation of CONWIP based Production Lines; and Relationships to MRP. *International Journal of Production Research*, **35**, 1067-1076.

Hopp, Wallace J., and Mark L. Spearman. 1996. *Factory Physics: Foundations of Manufacturing Management*. Irwin, Chicago.

Jackson, J. R. 1957. Networks of Waiting Lines. *Operations Research*, **5**, 518-521.

Jackson, J. R. 1963. Jobshop-Like Queueing Systems. *Management Science*, **10**, 131-142.

Karandkar, Rajeeva L., and Vidyadhar G. Kulkarni. 1995. Second-order Fluid Flow Models: Reflected Brownian Motion in a Random Environment. *Operations Research*, 43, 77-88.

Karmarkar, U.S. 1989. "Capacity Loading and Release Planning with Work-in-Progress (WIP) and Leadtimes." *Journal of Manufacturing and Operations Management*, **2**, 105-123.

Karmarkar, U.S. 1993. "Manufacturing Lead Times, Order Release, and Capacity Loading." *Logistics of Production and Inventory,* S. Graves, A. H. G. Rinnooy Kan and P. Zipkin (eds.), *Handbook in Operations Research and Management Science, Vol. 4*, North Holland.

Keilson, J. and L. D. Servi. 1988. A Distributional Form of Little's Law. *Operations Research Letters*, **7**, 223-227.

Keilson, J., and L. D. Servi. 1994. Networks of Nonhomogeneous $M/G/\infty$ Systems. *Journal of Applied Probability*, **31A**, 157-168.

Kelly, F. P. 1975. Networks of Queues with Customers of Different Types. *Journal of Applied Probability*, **12**, 542-554.

Lancaster, Peter, and Miron Tismenetsky. 1985. *The Theory of Matrices*, Second Edition. Academic Press, Inc., New York.

Leong, Thin-Yin. 1987. A Tactical Planning Model for a Mixed Push and Pull system. Ph.D. program second year paper, Sloan School of Management, Massachusetts Institute of Technology, July.

Luenberger, David G. 1979. *Introduction to Dynamic Systems: Theory, Models, and Applications*. John Wiley and Son, Inc., New York.

Mihara, Shoichiro. 1988. A Tactical Planning Model for a Job Shop with Unreliable Work Stations and Capacity Constraints. S.M. Thesis, Operations Research Center, MIT, Cambridge MA, January.

Parrish, Scott H. 1988. Extensions to a Model for Tactical Planning in a Job Shop Environment. S.M. Thesis, Operations Research Center, Massachusetts Institute of Technology, June.

Rice, John A. 1995. *Mathematical Statistics and Data Analysis, Second Edition*. Belmont: Duxbury Press.

Solberg, J. J. 1977. A Mathematical Model of Computerized Manufacturing Systems. Proceedings of the 4th International Conference on Production Research, Tokyo, Japan.

Spearman, Mark L., Wallace Hopp, and David Woodruff. 1989. A Hierarchical Control Architecture for Constant Work-in-Progress (CONWIP) Production Systems. *Journal of Manufacturing and Operations Management*, **21**, 147-171.

Suri, R. and J. L. Sanders. 1993. Performance Evaluation of Production Networks. In *Logistics of Production and Inventory*, S. Graves, A. H. G. Rinnooy Kan, and P. Zipkin (eds.) *Handbook in Operations Research and Management Science, Vol. 4*, North Holland.

Venkatakrishnan, C.S., Arnold Barnett, and Amadeo Odoni. 1993. Landings at Logan Airport: Describing and Increasing Airport Capacity. *Transportation Science*, **27**, 211-227.

Wein, Lawrence M. 1992. Dynamic Scheduling of a Multiclass Make-to-Stock Queue. *Operations Research*, **40**, 724-735.

Whitt, W. 1983a. The Queueing Network Analyzer. *Bell Syst. Tech. J.*, **62**, 2779-2815.

Whitt, W. 1983b. Performance of the Queueing Network Analyzer. *Bell Syst. Tech. J.*, **62**, 2817-2843.