On Control of Queueing Networks and the Asymptotic Variance Rate of Outputs

Yoni Nazarathy

A THESIS SUBMITTED FOR THE DEGREE
"DOCTOR OF PHILOSOPHY"

University of Haifa
Faculty of Social Sciences
Department of Statistics

November, 2008

This page is back of cover - throw away.

On Control of Queueing Networks and the Asymptotic Variance Rate of Outputs

By: Yoni Nazarathy

Supervised by: Professor Gideon Weiss

A THESIS SUBMITTED FOR THE DEGREE
"DOCTOR OF PHILOSOPHY"

University of Haifa
Faculty of Social Sciences
Department of Statistics

November, 2008

Recommended by: _____ Date: _____
(Advisor)

Approved by: _____ Date: _____
(Chairman of Ph.D Committee)

ברצוני להקדיש עבודה זו לזכרה של סבתה סופיה, האישה המתוקה שזרעה בי זרעים של חוזק, הן באמצעות האהבה הטהורה אשר סיפקה לי עד ימייה האחרונים והן בדרך עקיפה דרך גידולה המסור של אמי לאה ובאופן מסוים גם אבי משה, זוג ההורים הכי נהדרים שאפשר לדמיין, לפעמים כה נהדרים שאפילו קשה לדמיין.

סבתה שרדה באומץ את שואת היהודים של המאה ה 20, הצליחה בדרך נס לפגוש את סבא נחום האהוב, וביחד גידלו שתי בנות מופלאות אשר הביאו שבעה נכדים לעולם, כולם אנשים איכותיים וטובים אשר יישארו קרובים לליבי כל עוד הוא פועם. בנוסף, עד היום נולדו לסבתא שבעה נינים, את רובם אמנם לא פגשה, שתיים מהם הן אמילי וקיילי, ילדות הזהב אשר הגדירו בשבילי מחדש את המושג אהבה, כמשהו אבסולוטי וללא התניות, והרבה מכך תודות לאימם, אשת הברזל והפרחים שאני אוהב כל כך, כרמל.

סבתה לעולם לא פגשה את אמילי, נסיכת הנסיכות שלי, אשר נולדה מספר חודשים לאחר מותה, בתחילת תקופת הדוקטורט, וגם לא את קיילי בת השנה, כוכב הזהב המתוקה עלי האדמות. סבתא גם לא הייתה מאמינה שאסיים את הדוקטורט, ולאור מה שהכירה בעודה בחיים כנראה שצדקה, לא הייתי עושה זאת ללא הרוגע והנחת אשר קיבלתי מאמילי וקיילי ומכרמל וגם לא הייתי עושה זאת ללא מטריית העזרה רחבת ההיקף אשר קיבלתי מהורי טובי הלב ומאחי נדב ואחיותיי נעמה וענת.

אז יללה, לאחר כל הפוצי שמוצי הזה: החיים זה דבר קצר, צריך לתת גז עד ההקדשה הבאה.

ACKNOWLEDGMENTS

I thank Gideon Weiss for his guidance during my Ph.d studies. Having known Gideon previously, from my Master's studies, I already knew that he is an extraordinary teacher, a hard working scientist and most importantly a genuinely good caring person. But I did not yet have the full set of tools to realize what a unique researcher he is. Only now, that this Ph.d is complete, I believe to understand the true importance of some of Gideon's scientific contributions and appreciate his unique approach towards problem solving. I deeply thank him for guiding me during my academic quest, and I feel very lucky for knowing him.

I would also like to thank the staff members of the statistics department at the University of Haifa. What a wonderful period this was! First, the administrative staff was extremely helpful and always made me feel at home, Shelly, Ofra and Hana. Ofra has become a true friend and Hana is incredible in how she knows how to always get things moving efficiently and with a smile. In addition Esti Frostig , Ehud Makov, Shmuel Gal, Zinoviy Landsman, Ori Davidov and Alex Goldenshluger were all wonderful teachers of mine during different periods. Noya Galai, Nitza Barkan, Yonit Barron and Benjamin Reiser were very helpful with regards to teaching related issues. David Faraggi and Shaul Bar-Lev were very helpful in obtaining financial support. Yuval Nov served as a role model for how to do things right. It was also very enriching to have many discussions with David Perry which was always offering genuine help where needed. I also wish to thank my fellow Ph.d students for their support, specifically Itai Dattner and Michal Daloya. I also deeply thank Avner Halevy, the department chair, for his openness, friendliness and professional support during this period.

I would also like to thank Ward Whitt for pointing out some important results which were heavily used in this research. In addition Sergey Foss helped clarify some puzzling issues regarding Markov processes and I thank him for that. In addition, the following persons have inspired my research during this period and discussions with them were very useful and helpful: Itai Gurvich, Wolfgang Stadje, Brian Fralix, Yoav Kerner, Avi Mandelbaum, Nahum Shimkin and Moshe Haviv. I also thank Uri Yechiali for very helpful guidance and inspiration.

Last but not least, I would like to thank the higher studies authority at the University of Haifa for their vast financial support through a variety of grants and scholarships. I also thank the management at the Systems Division at Rafael Industries for their flexibility in allowing me to take a leave for the studies, and hope they accept my apology for not returning to the engineering world. Finally, I thank Gideon Weiss once more for his unparalleled financial support through the Israel Science Foundation and the European Network of Excellence, Euro-NGI.

CONTENTS

On Control of Queueing Networks and the Asymptotic Variance Rate of Outputs

Yoni Nazarathy

Abstract

In this thesis we study several topics related to the control of queueing networks and analysis of the asymptotic variance rate of output processes. We first address the problem of optimal control of a multi-class queueing network over a finite time horizon with linear holding costs. Our method for control and its analysis was published in Nazarathy and Weiss (2008b). We then analyze the stability properties of an example network with infinite virtual queues which we call the push-pull network. This network can be controlled in a way such that the servers operate all of the time while the queues remain stochastically bounded as in Kopzon *et al.* (2008). Our analysis generalizes the memoryless processing time results of that paper to the case of general processing durations. We utilize the fluid stability framework for showing positive Harris recurrence of Markov processes associated with queueing networks. These results were published in Nazarathy and Weiss (2008c).

The sample path behavior of the push-pull network has motivated us to analyze the variability of its output processes. A first measure for such variability is the asymptotic variance rate: the linear increase of the variance function of a counting process over time. Experimenting with this performance measure, we observe an interesting phenomena that occurs in simple finite capacity birth-death queues and obtain a closed formula for the asymptotic variance rate for such systems. These results have been published in Nazarathy and Weiss (2008a). Returning to the Push-Pull system, we obtain expressions for the asymptotic variance rate, by means of a diffusion limit whose proof relies on our positive Harris recurrence result.

Finite Horizon Control

Our method for control of a multi-class queueing network over a finite time horizon integrates several ideas: Separated continuous linear programs, infinite virtual queues and rate stable maximum pressure policies. We approximate the multi-class queueing network by a fluid network and formulate a fluid optimization problem which we solve as a separated continuous linear program. The optimal fluid solution partitions the time horizon to intervals in which constant fluid flow rates are maintained. We then use a policy by which the queueing network tracks the fluid solution. To that end we model the deviations between the queuing and the fluid network in each of the intervals by a multi-class queueing network with some infinite virtual queues. We then keep these deviations stable by an adaptation of a maximum pressure policy. We show that this method is asymptotically optimal when the number of items that are processed, and the processing speed increases.

Full Utilization Control

As summarized above, our second topic on control deals with the push-pull queueing network. This network is composed of two servers and two types of jobs which are processed by the two servers in opposite order, with stochastic generally distributed processing times. This push-pull network is similar to the Kumar-Seidman Rybko-Stolyar (KSRS) multi-class queueing network, with the distinction that instead of random arrivals, there is an infinite supply of jobs of both types. Thus each server can either process jobs of one of the types, which it pulls from the other server, or jobs of the other type which it pushes out of the infinite supply towards the other server. Unlike the KSRS network, we can find policies under which our push-pull network works at full utilization, with both servers busy at all times, and without being congested. We perform an asymptotic analysis of the push-pull network under these policies to quantify its behavior: We show that under fluid scaling the fluid model of the network is stable. We adapt the proofs of Dai, to show that as a result the queues of jobs waiting for pull operation are positive Harris recurrent.

Asymptotic Variance Rate of Outputs

With regards to the output process of finite capacity birth-death Markovian queues, we develop a formula for the asymptotic variance rate of the form $\lambda^* + \sum v_i$ where λ^* is the rate of outputs and v_i are expressions based on the birth and death rates. We show that if the birth rates are non-increasing and the death rates are non-decreasing (as is common in many queueing systems) then the values of v_i are strictly negative and thus the limiting index of dispersion of counts of the output process is less than unity. In the M/M/1/K case, our formula evaluates to a closed form expression that shows a rather surprising phenomena: When the system is balanced, i.e. the arrival and service rates are equal, $\frac{\sum v_i}{\lambda^*}$ is minimal. The situation is similar for the M/M/c/K queue, the Erlang loss system and some PH/PH/1/K queues: In all these systems there is a pronounced decrease in the asymptotic variance rate when the system parameters are balanced.

Moving to the output processes of the push-pull network, we are interested in the asymptotic variance rate as well as the covariance rate between the two processes. We do so by means of diffusion limits. Our results show that the two output streams are highly negatively correlated and that the asymptotic variance rate of outputs is the same for all fully utilizing stable policies.

We apply the same diffusion limit methodology to a general re-entrant line with infinite supplies and obtain a simple expression for the asymptotic variance rate of outputs.

Background

We also present an extensive but elementary background chapter about queueing networks that is written with the non-queueing theorist in mind. In addition we present a survey chapter on previous results of networks with infinite virtual queues. An additional chapter introduces concepts related to the asymptotic variance rate of outputs. Hope you enjoy reading.

LIST OF TABLES

LIST OF FIGURES

OVERVIEW

In this thesis we summarize our research with regards to several topics related to the control of queueing networks and the analysis of the asymptotic variance rate of output processes. We now briefly overview our main results and discuss the organization of the text.

Brief Overview

The study of queueing networks probably began with the discovery of the big miracle of product form Jackson networks in the 1950's and ever since it has been an exciting research area finding applications in manufacturing, communications and service engineering but also motivated by the mathematical elegance and charm of some results and problems. Controlled queueing networks have only received serious attention in the past 20 years, possibly due to the fact that typically an exact analysis of a network under a given control is not possible and very rarely is one able to find an optimal control. In this respect, one usually seeks to find controls that are either asymptotically optimal or at least sensible in the sense that they maintain the network stable. The latter issue of stability has become a pressing issue in the late 90's due to the discovery of some simple network examples that are unstable under controls that seem quite innocent at first sight. In general, it appears that the theory of queueing network control still has a long way to go until reaching maturity.

Our contributions to this field involve queueing networks in which some classes have an infinite supply of work which we term *infinite virtual queues*. First we handle the problem of finite horizon control of a standard multi-class queueing network with respect to minimization of linear holding costs. We propose a policy which we prove to be asymptotically optimal when the number of items processed and the processing speed increase. The problem of finite horizon control is typically a more complicated problem than infinite horizon since one can not assume that steady state is reached. In this case we approach the problem by assuming that there is a large amount of items that need to be produced and the processing time of each item is small. This allows us to approximate the problem by a finite horizon fluid optimization problem, which may be solved optimally as a separated continuous linear program. We then show how to track the optimal fluid solution in a way that achieves asymptotic optimality of

the discrete stochastic problem. This tracking procedure involves infinite virtual queues and the maximum pressure policy.

Secondly, we explore an example of a queueing network in which some buffers have infinite supplies (infinite virtual queues), this is the push-pull network. We believe that this type of model may offer an attractive alternative to the Brownian network models which have received a lot of attention in the past 20 years. We show that the specific model that we analyze, the push-pull network is positive Harris recurrent under a control that fully utilizes the resources. This behavior is very different from the typical heavy traffic networks in which congestion increases when utilization nears 100%.

Following our results regarding network control we focus on analysis of the asymptotic variance rate of outputs. Classical queueing theory, typically attempts to evaluate performance measures that are important from the customers point of view, e.g. sojourn times. This is possibly due to the fact that queueing theory research was initially motivated by problems of human customer service. Alternatively, when manufacturing systems and some types of communication networks are considered, it is reasonable to shift attention to other performance measures related to the output stream. One such important attribute is the variability of the output stream of jobs which may be measured by the asymptotic variance rate of outputs: the asymptotic rate of increase of the variance function of the output counting process. When the output process obeys some sort of central limit theorem then the asymptotic variance rate of outputs may be used to estimate the distribution of the number of outputs during long time intervals.

Our interest in the variability of outputs stems from the push-pull network. In this network, the behavior of the output streams under the proposed controls typically follows an alternating on-off type behavior: The production of one type of job by the network operates continuously at a fast rate while the other type of job idles or slows down and the situation is reversed after a random duration whose length is similar to the order of a busy period. This behavior motivates us to analyze the asymptotic variance rate of outputs with hope that it can be quantified so that one can search within the class of fully-utilizing stable policies for a policy that yields low variability of outputs. Analysis of the asymptotic variance rate is typically not relevant to queueing networks without infinite virtual queues and without losses because in such networks the asymptotic variance rate of the outputs usually equals that of the inputs. As a result of our research we now believe that a similar phenomena appears in networks with infinite virtual queues: the asymptotic variance of the outputs is not amenable to control under stable fully utilizing policies.

As a "warm up" of the analysis we are led to explore methods for calculating the asymptotic variance of point processes that are generated by simple queueing systems. In this respect we have explored explicit matrix-analytic results related to Markovian arrival processes (MAPs) and their variance function. For the elementary M/M/1/K queue, we show that the asymptotic variance of outputs is minimized when the system is balanced (the arrival rate and service rate are equated). We call this phenomenon BRAVO (Balancing Reduces Asymptotic Variance of Outputs). It appears to carry over to other finite capacity birth death queues. We also present a

useful simple formula for calculating the asymptotic variance rate of finite capacity birth death queues.

Returning to the push-pull network we present a diffusion approximation of the output processes which yields expressions for the asymptotic variance rate of outputs (as well as the covariance between output streams). In addition, we derive the asymptotic variance rate of outputs of an infinite supply re-entrant line. These results confirm the fact that all fully utilizing stable policies of the push-pull network exhibit the same asymptotic variance rate of outputs.

Organization of the Thesis

Part I is composed of Chapters 1 and 2 which contain supporting material for the results of the thesis. Reading this material is helpful for following the later chapter but may also be skipped by a brief reader.

Chapter 1 contains a brief survey of the ideas of queueing network theory. It is written with the non-specialist in mind but also attempts to entertain readers that are well within the field. Landmark results and methods of queueing networks are surveyed with only a minor bias towards results that are relevant to our research.

Chapter 2 introduces the concept of infinite virtual queues. Incorporating this idea in a queueing network yields interesting network models such as the push-pull network which we later analyze. In this chapter we survey the work that has been done on this subject until now, excluding our contribution which appears in later chapters.

Part II is composed of Chapters 3 and 4. Here we present our results with regards to control of queueing networks.

Chapter 3 handles finite horizon optimal control. Here we present our method for controlling a multi-class queueing network over a finite horizon that is asymptotically optimal with regards to holding costs. This method and its analysis employs three different concepts: separated continuous linear programs, multi-class queueing networks with infinite virtual queues and maximum pressure policies. The results of this chapter were published in Nazarathy and Weiss (2008b).

Chapter 4 presents our analysis with regards to stability of the push-pull network, showing that it is possible to control a network under full utilization while keeping the queues stable. This has actually been shown previously for a memory less system in Kopzon and Weiss (2002) and in our joint submitted publication Kopzon *et al.* (2008). Here we expand the results to the case of general processing times, employing an asymptotic analysis. We show that under certain policies, the network is positive Harris recurrent. The results of this chapter were published in Nazarathy and Weiss (2008c).

Part III is composed of Chapters 5, 6 and 7. This part deals with the variance of output processes.

Chapter 5 introduces the performance measure of asymptotic variance rate of outputs. We look at the M/G/1 queue as an example and also at a specific case of the push-pull analyzed in the previous chapter.

Chapter 6 contains our results with regards to the asymptotic variance rate of the output

process of finite capacity queues. As opposed to the results in the previous chapters, this is a more "classic" queueing result which deals with the most fundamental queueing systems studied. The results of this chapter were published in Nazarathy and Weiss (2008a).

Chapter 7 analyzes the asymptotic variance rate of outputs of the push-pull network of Chapter 4 and of a general re-entrant line with infinite supply. Our results are obtained by means of diffusion limit theorems for both types of networks.

Appendix A outlines some details regarding a simulation software package that we developed and used to obtain most of the simulation results and illustrations in this thesis.

Part I

Background

CHAPTER 1

QUEUES AND NETWORKS

This chapter serves as an introduction to the subject matter of the thesis: queueing networks. The purpose is to present some basic concepts of queueing theory and queueing networks.

We begin very informally by introducing the basic phenomena of queues, congestion and stability in Section 1.1. We do this by demonstrating a simulation experiment that uses some real data set. We then discuss how queueing theory attempts to describe such phenomena and outline some important measures of performance. We continue our presentation in Section 1.2 where we briefly overview some key results and directions of "classic queueing theory". This is not a very well defined term, nevertheless we use it to refer to results having to do mainly with explicit (usually steady state) solution of stochastic systems, often related to the Poisson process.

Once the queueing setting has been established, we move on to discuss the notion of a queueing network in section 1.3. Here we describe several variations of queueing network models. We also explicitly define a multi-class queueing network (MCQN), a concept that re-appears in future chapters. A very basic queueing network is the 2 station tandem queue, we use it as an example for analysis in the following three sections.

In Section 1.4, we survey results regarding queueing networks that have a product form. These networks are quite miraculous in the fact that their steady state behavior may be explicitly calculated. Arguably, research with regards to finding such exact solutions of queueing networks peaked in the early 80's (at least for the mean time) with the publication of the book, "Reversibility and Stochastic Networks", Kelly (1979). This in no means implies that "all is known" but rather indicates that at least for the mean time, approximate solutions should be attempted. We thus, survey two major approximation paradigms, both of which have enjoyed great popularity.

The first approximation paradigm is based on a heuristic argument (without any theoretical justification) and uses the concept of network decomposition. Probably the most notable item in this line of approximations is Ward Whitt's Queueing Network Analyzer (QNA). The general idea is to use approximations for the traffic processes between the nodes of the network. We demonstrate the method of the QNA on a 2 station tandem queue and briefly describe how it

can be applied to general networks.

The second type of approximation paradigm is based on diffusion approximations of queues in heavy traffic. As opposed to the QNA scheme, this method is usually backed by theoretical limit theorems that justify the validity of the approximation as some network parameter reaches a limiting value. We attempt to introduce the flavor of such limit theorems by discussing the diffusion approximation of the 2 station tandem queue in Section 1.6.

While obtaining performance measures related to the congestion of the network are of interest, sometimes a much more fundamental question is to be asked: Is the network stable? Analysis of this question became popular in the 90's due to some amazingly simple yet surprising discoveries of queueing networks that have enough capacity to handle the offered load yet are not stable under some quite sensible policies. We demonstrate this in Section 1.7 where we also define several notions of stability and instability. Some of the instability phenomena which we survey, emphasized the need to find criteria and methods for analyzing the stability of queueing networks. A key method that is utilized in our research is the fluid stability framework, advanced by Jim Dai (Dai, 1995).

1.1 Demonstration of the Basic Phenomena of Queues

As a prelude to the material of this background chapter and to the contents of this thesis we choose to present a small queueing experiment that does not implicitly involve any probabilistic assumptions, yet demonstrates the typical behavior of a queue and the phenomena of congestion, stability and the steady state distribution.

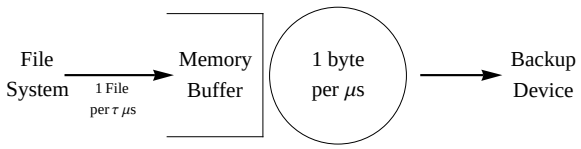

Figure 1.1: Memory buffer fed by files. A file is dumped to the memory buffer every τ micro seconds. The backup device consumes 1 byte per micro second from the buffer.

We imagine that there is some file backup device that is connected to our lap-top computer and that there is an allocated memory buffer in the computer's RAM that is dedicated to the backup device. We further assume that whenever the memory buffer is not empty, the device draws bytes from the memory buffer at a constant rate of exactly 1 byte per micro second. When the buffer is empty, the device idles. In addition there is some software mechanism that fills up the memory buffer at a constant rate with the data of the files that are on the lap-top. One file is dumped to the memory buffer every τ micro seconds. Assume that the copying time of a file from the file system to the memory buffer is instantaneous (or negligible). Note that the variability in the system is due to the variability of the file sizes. A schematic representation of this setup is in Figure 1.1.

We shall now experiment with the behavior of this memory buffer (or queue) over time. Our experiment consists of non-fictitious (real) data drawn from the file system on our lap

top. Specifically we obtained a list of $76,904$ file sizes by scanning the whole file system. The maximal size is quite big: $471,593,369$ bytes (about half a gigabyte). The mean is much smaller: $109,255$ bytes. It should be noted that $6,324$ of these are actually directories and have 0 size, this is fine, we treat those as files of size 0.

It is easy to simulate the number of bytes in the memory buffer at the times $0, \tau, 2\tau, \ldots$. These are the times at which a new file is put in the buffer. We shall actually look at the time instances immediately after a new file is put in the buffer. These are calculated by the recursion:

$$X_{n\tau} = \max[X_{(n-1)\tau} - \tau,\ 0] + \sigma_n,\ \ n = 1, 2, \ldots, 76904 \tag{1.1}$$

where $\sigma_1, \ldots, \sigma_{76,904}$ are the file sizes and we assume $X_0 = 0$. The process $\{X_t, t \geq 0\}$ is the memory buffer (or queue) process over time. Assuming the file sequence is fixed, the realization of this process depends only on the value of τ. We plot the initial part of the realization for several values of τ in Figure 1.2.

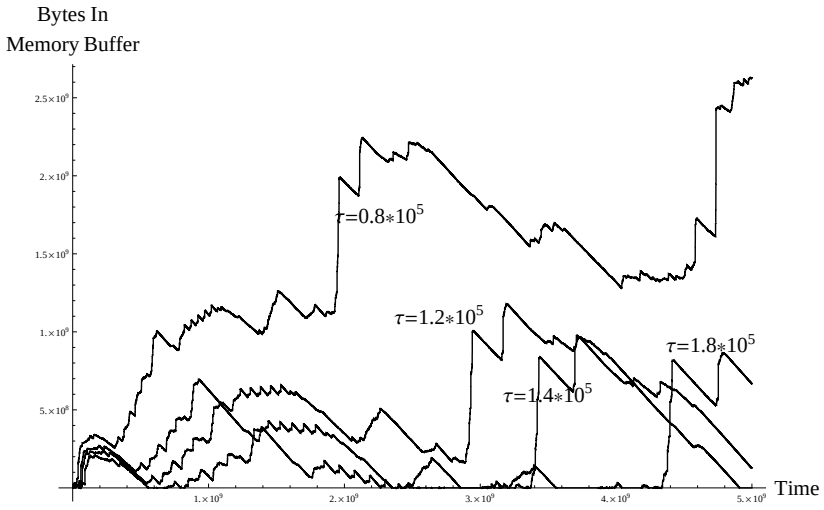

Figure 1.2: Realizations of the buffer content as a function of time for various values of τ.

Note that we have chosen values of τ that are pretty close to the mean files size, which is approximately 1.1×10^5. It is quite evident that as τ increases, the realization of the buffer size process decreases. This simple phenomenon is one of the main issues which we wish to explore in this section, more on it shortly. A second observation is that the realizations are quite smooth except for several very big instantaneous jumps. These correspond to some extremely big files that are occasionally encountered. This behavior may be viewed as an attribute of our file system: the distributions of the files contains a "heavy tail" meaning that there actually exist some very big files in the presence of a lot of very small files.

While the "heavy tail" phenomenon that we observe is extremely interesting and is actually related to some very contemporary research on queues, it is not the focus of our thesis and the current discussion. Thus we choose to avoid it and simplify things. We do this by assuming that our device does not accept files with more than 5×10^5 bytes. After removing these "excessively big files" from our data (a total of $2,175$ files), the new mean file size is $27,475$.

A further issue which is extremely interesting, yet not relevant to our main discussion is the fact that our list of file sizes is most probably quite structured. This list was obtained by

traversing through the file system in some lexicographic order. Files from the same directory were obtained one after the other. It is most probable that some directories contain many small files and others contain many big files and thus the data may be quite correlated. Actually by carefully observing Figure 1.2, some correlations in the consecutive file sizes are apparent. For example, the path for $\tau = 1.8 \times 10^5$ at around the time 1×10^9 appears to contain some "repetitive behavior". This is most probably due to having several directories with one big file and a lot of small files. Where the directories are almost exact copies of each other. To remove this "correlation" from our discussion, we assume that the file backup mechanism traverses the file system in random order. We simulate this by shuffling the file size list according to a random permutation.

Before we continue the discussion using our modified "uncorrelated" and "without heavy tails" file list, let us define,

$$\rho = \frac{27,475}{\tau},$$

(remember that the mean file size is 27,475 bytes and that bytes are drained at rate 1). In queueing theory, ρ usually symbolizes the *offered load* or *traffic intensity* of the system under study. When $\rho < 1$, the rate of inflow to the queue is less than the potential outflow. When $\rho = 1$ the queue is balanced. When $\rho > 1$ there is more coming into the queue then can be handled and the system is said to be *overloaded*. We now repeat the queue simulation (with the modified data set), for several values of ρ. The results are in Figures 1.3, 1.4 and 1.5.

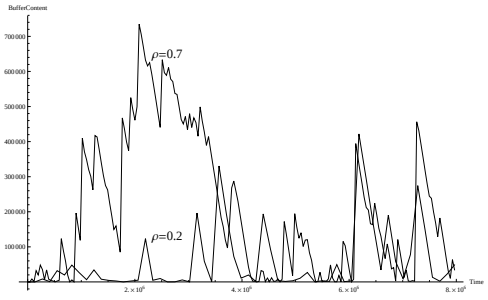

Figure 1.3: $\rho < 1$.

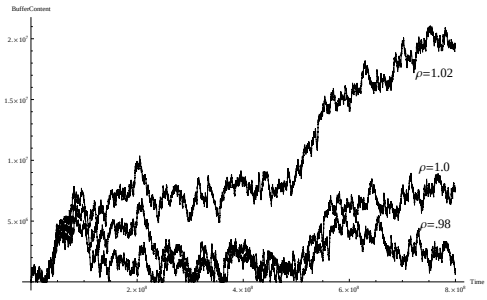

Figure 1.4: $\rho \approx 1$

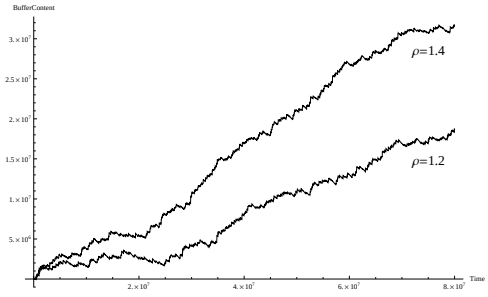

Figure 1.5: $\rho > 1$

The behavior observed in these figures is quite typical of queueing systems. When $\rho < 1$ the buffer level process alternates between periods of being empty and periods of being full (these are respectively called the *idle period* and the *busy period*). When $\rho > 1$ there is simply not enough capacity to match the arrivals and thus the size of the buffer increases at a rate which is

pretty much linear with time. When $\rho \approx 1$ the system is said to be balanced and the realization of the memory buffer actually looks like a diffusion process (more on that is in Section 7.2). Without variability, a system with $\rho \leq 1$ will stay empty. But in the presence of variability (due to variable file sizes in our example), queues build up.

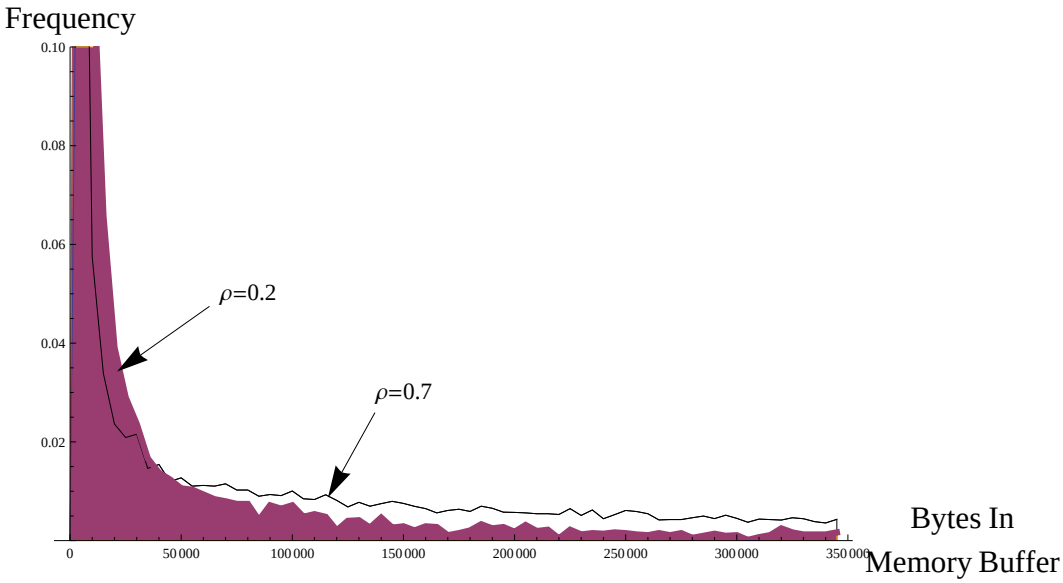

Figure 1.6: File experiment: Observed steady state distributions for $\rho = 0.2$ and $\rho = 0.7$

Our purpose in the above discussion was to show how the mode of operation ($\rho < 1$, $\rho = 1$ or $\rho > 1$) affects the general behavior of a queue (stable, marginally stable and unstable). We now wish to concentrate on the case where $\rho < 1$. Observe in Figure 1.3 that the realization of $\rho = 0.2$ appears to be generally lower than the realization of $\rho = 0.7$. How can we quantify this? The most common measure of performance in this respect is to look at the distribution of buffer level throughout the realization and further at the mean. We hope to be able to assume that when $\rho < 1$ this distribution "settles" to some *stationary distribution* and a further assumption is that the distribution has a finite mean. Assuming both assumptions to be true, we can look at histograms of the buffer content distribution sampled at equal points in time and further look at the mean. Such distributions for $\rho = 0.2$ and $\rho = 0.7$ are in Figure 1.6. Indeed the observed mean for $\rho = 0.2$ is $37,200$ while the mean for $\rho = 0.7$ is much higher: $174,567$.

Queueing theory usually attempts to describe such observations as above by assuming a stochastic model. In our example, the simplest model is to assume that the files sizes $\{\sigma_n, n = 1, 2, \ldots\}$ are drawn independently from some identical distribution and the recursion (1.1) continues indefinitely. The i.i.d. assumption is a fair modeling assumption since we have stated that files are drawn at random (according to some random permutation) and not in lexicographic order.

Once we have a stochastic model, we can attempt to analyze several performance measures either analytically or by simulation as we have done here. For example an ambitious task would be to determine the distribution of the memory buffer at any time t given some initial buffer level, X_0. This is sometimes called the *transient* or finite time behavior of the system and is typically much harder to analyze than the behavior at time $t \to \infty$ (as is empirically

estimated in Figure 1.6). We will not discuss transient behavior any further.

Let us now repeat the file size experiments for different values of ρ and calculate the sample mean, $\frac{\sum_{n=1}^{N} X_{n\tau}}{N}$ where $N = 74,729$ is the number of files in the modified data set. In Figure 1.7 we show the observed means for values of ρ less than unity[1]. It is apparent that as the offered load increases the mean number of bytes in the queue increases. This type behavior of the mean work in system is typical of queueing systems and reflects the typical trade-off that exists between utilization and congestion. We shall now see it in the simplest and most fundamental queueing model analyzed: The M/M/1 queue.

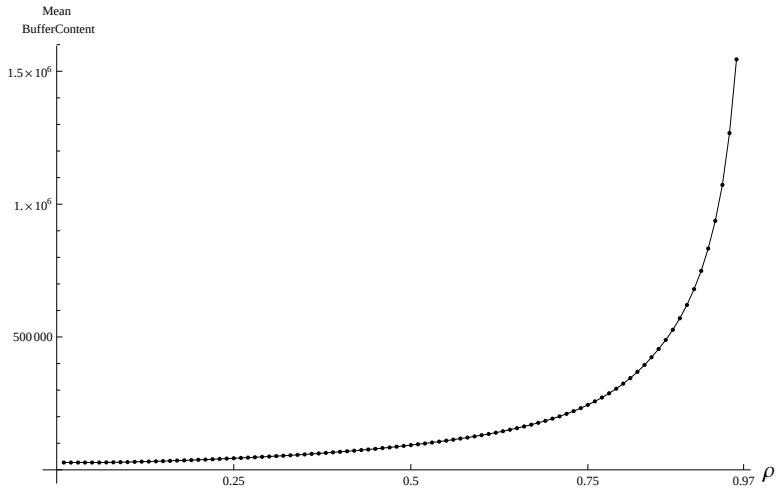

Figure 1.7: File experiment: Estimated mean buffer size over time as a function of ρ for $\rho < 1$.

The M/M/1 queue is a single server queue, see Figure 1.8. This queue is modeled by assuming that there is some arrival process of discrete jobs that queue up. The server works on each job for some random duration and then releases it and moves to the next job or idles if the queue is empty. Note that it differs from the memory buffer simulated above, in that material flow is discrete (as opposed to an almost continuous byte out flow). The mean number of jobs arriving per unit time is usually denoted λ and the mean service time is μ^{-1}. The traffic intensity is denoted by

$$\rho = \frac{\lambda}{\mu},$$

and it can be shown that a necessary and sufficient conditions for the existence of a stationary distribution for the number of jobs in the system is that $\rho < 1$. In the M/M/1 queue, the simplest assumptions are assumed on the arrival and processing times: It is assumed that the arrival process is Poisson with rate λ and the service times are i.i.d. exponential with mean μ^{-1}. In this case, the number of jobs in the system at time t, denoted by $Q(t)$ is a countable state space Markov chain and it is easy to show that

$$\lim_{t\to\infty} P\{Q(t) = k\} = (1-\rho)\rho^k, k = 0, 1, 2, \ldots \tag{1.2}$$

[1]Note that this is the mean sampled at the points of arrivals of new files and it is not necessarily equal to the time average over continuous time. For some systems, these two means are the same, e.g. when the arrivals are according to a Poisson process, but in general they may be different.

This implies that the steady state mean queue size is $\rho/(1-\rho)$ and indeed a graph of this function is similar in nature to the graph we obtained in Figure 1.7: As ρ increases to 1 the mean queue length increases rapidly. More on queueing models of this sort is in the next section.

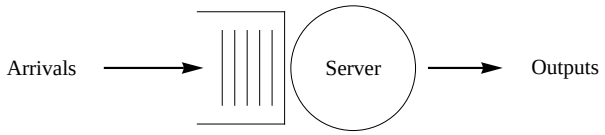

Figure 1.8: A schematic representation of a single server queue.

In summary, this is the basic queueing behavior that was demonstrated in this section[2]:

- Queue levels are typically "stable" when $\rho < 1$, "marginally stable" when $\rho = 1$ and "unstable" when $\rho > 1$.
- Stable systems with $\rho < 1$ typically have a stationary distribution for quantities such as the queue length.
- The mean steady state number of units in the system typically increases as ρ increases to 1 in a manner similar to the M/M/1 queue.

It is always important to keep in mind, that it is variability of the file sizes that caused the queues to build up (when $\rho < 1$). This is further demonstrated through steady state formulas of the M/G/1 queue presented in the next section.

1.2 Classic Analysis of Queues

Queueing theory primarily deals with analysis of stochastic models of queues. This formal analysis began in the beginning of the previous century with the works of the telephone traffic engineer A. K. Erlang who set the tone of the theory by analyzing queues that may be represented as continuous time Markov chains. These are today called (following Kendall's notation, see any elementary book on Queueing theory) the M/M/1, M/M/c, M/M/1/K, M/M/K/K (Erlang loss system) and others. The nice thing about these types of queues, is that one can quite easily obtain the stationary distribution of the number of jobs in the system. Performing these types of calculations has been termed "Elementary Queueing Theory" (Kleinrock, 1974) and is usually covered in introductory courses on stochastic performance analysis. An example steady state result is given for the M/M/1 queue (1.2) above and the general case is summarized later in this thesis in Section 6.2 of Chapter 6.

Elementary queueing theory models assume a Poisson arrival process which is a reasonable assumption in many settings because the Poisson process occurs in nature quite frequently when arrivals are due to independent "micro-Bernoulli" trials or when the arrivals are a super position of many point processes. The models also assume that processing times follow an

[2]Some further demonstrations of a similar nature may be found in a web-site which we are now developing: *The Queueing Science Exploratorium*: `http://www.stat.haifa.ac.il/~yonin/qsm/main.html`.

exponential distribution, i.e. they have constant hazard rates and are thus memory-less. For many applications, this assumption is often too restrictive.

Some types of systems such as the so-called Erlang loss system (M/M/K/K queue) enjoy a property called insensitivity which essentially means that the steady state solution of M/M/K/K is the same as that of an M/G/K/K system with same service mean but some arbitrary service distribution. Here the 'G' in Kendall's notation stands for "General Distribution" while the 'M' stands for "Markovian" assumptions, i.e. Poisson arrival processes and exponential processing times. In general systems are not insensitive: changing the shape of the service or inter-arrival distribution (even when the mean does not change) affects the behavior of the system.

The next step up, after analysis of Markovian systems is analysis of systems where one of the driving components is based on an exponential memory-less distribution and the other is based on general service times (from some given distribution). The two typical examples are the M/G/1 queue and the G/M/1 queue. Here the 'G' stands for either the service times (in M/G/1) or inter-arrival times (in G/M/1) being based on a sequence of i.i.d. random variables from some arbitrary distribution. Obviously M/M/1 is a special case of both. M/G/1 is of particular practical importance because it removes the sometimes unreasonable assumption of memory-less service times. For example, as is often encountered in time slotted communication systems, one can analyze a system with Poisson arrivals and deterministic service times using known results for M/G/1.

The steady state behavior of the M/G/1 queue is described by the Pollaczek-Khintchine (P-K) formulas and we present them below. Arguably, these formulas are the most outstanding success story of classic queueing theory. And further (also arguably) when one tries to go further than M/G/1 there are quite a bit of hardships and explicit useful results are very hard or impossible to obtain. For example, it is not known how to obtain the steady state solution for the number in queue for the M/G/2 queue (this is a 2 server queue). Note though that the G/M/m with $m \geq 1$ queue has been solved. Further, the behavior of the $G/G/1$ queue can only be approximated (sometimes the approximation is accompanied by rigorous limit theorems). This does not mean that research regarding explicit queueing theory results has settled, there are still many variations and combinations that may be obtained, sometimes employing quite sophisticated reasoning and analysis. A landmark manuscript of classic queueing theory is "The Single Server Queue", of Cohen (1982).

For illustration we shall now summarize the (P-K) formulas. Assume that jobs arrive according to a Poisson process with rate λ and each job performed with service times that are taken from an i.i.d. sequence with a mean μ^{-1} where $\lambda < \mu$. Denote the distribution function of the service times $H(t)$ and assume that it has a Laplace-Stieltjes transform denoted by

$$H^*(s) = \int_{0^-}^{\infty} e^{-st} dH(t).$$

Further assume that jobs are performed according to a FCFS (first come first served) policy. In this case the (P-K) formula for the mean waiting time in queue of a job is:

$$\frac{\rho}{1-\rho}\mathbb{E}\left[R\right],$$

where $\rho = \lambda/\mu$ and $\mathbb{E}[R]$ is the mean residual service time of a randomly observed job in operation. From renewal theory it is well known that

$$\mathbb{E}[R] = \frac{1}{2\mu}(c_S^2 + 1),$$

where c_S^2 is the squared coefficient of variation of the service time. For the special case of the M/M/1 queue we have $c_S^2 = 1$ and thus the mean waiting time in queue is

$$\frac{\rho}{\mu - \lambda}.$$

It is instructive to also look at the case of deterministic service times and see that the mean waiting time in the queue is half of the exponential service time case. Indeed, as a general rule in queueing theory, variability increases waiting times and queue sizes. A strong feature of the M/G/1 queue is that the second moment of the service time distribution is all that is needed to quantify this.

The above mean waiting time formula of M/G/1 can be derived in several ways, the most instructive being based on renewal theory (see almost any standard reference on queueing theory). It can also be obtained by differentiation of the more general (P-K) formula for the Laplace-Stieltjes transform of the mean waiting time:

$$W^*(s) = \frac{s(1-\rho)}{s - \lambda(1 - H^*(s))}.$$

By inverting $W^*(s)$, one can obtain the distribution function of the waiting time and answer questions of practical importance such as: "What percentage of the jobs wait in queue more than 5 minutes". A similar formula exists for the probability generating function of the number of jobs in the system (it is also sometimes called the (P-K) formula).

In general, many results of "classic queueing theory" are stated in terms of transforms of the quantities of interest. The results we present in this thesis are not of this nature.

1.3 Queueing Network Models

What is a queueing network[3]? It is a collection of service stations (nodes) that are interconnected so that the output of some stations are fed into the input of others. Queueing networks are used to model manufacturing systems, certain communication systems, patients in hospitals, law cases in the justice system and a variety of other applications areas and natural phenomena. We do not elaborate on applications any further in this thesis.

In this section we introduce several variations of queueing network models. We shall describe the following notions: open networks, closed networks, single-class networks and multi-class networks. We shall also briefly comment on networks with infinite virtual queues. Other types of network concepts such as loss networks, finite buffers networks with blocking, G-networks and fork-join networks are briefly commented on.

Typical terms in the queueing network setting are: *Jobs* these are the entities that move through the queueing networks (also called customers or packets). *Buffers* (or queues), these are

[3]Also sometimes refereed to as *stochastic network*.

the place holders of jobs as they move around. *Activities* are the operations that are performed, servicing the jobs and once completed moving them between buffers (also called steps). *Resources*, sometimes called servers or processors or machines, are the entities that are used to perform the activities. Another term is a *node* or station. This is a collection of one or more buffers and one or more resources.

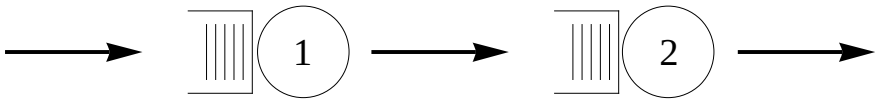

Figure 1.9: The 2 station tandem queue.

The simplest example of a queueing network is in Figure 1.9. We shall refer to it as the *2 station tandem queue*, it will be very useful in explaining some of the methods of analysis which we present in the following sections. Assume customers arrive from the outside world first into station 1, queue up, receive service and then move to station 2 receive service and then depart. Further assume that the servers service the customers in a FCFS non-idling fashion. This is an *open*, single-class, infinite buffer queueing network. We say it is open because it is connected to the outside world. We shall explain the term single-class shortly. An alternative to the open network is in Figure 1.10. This network is closed an contains a fixed finite customer population, it is an example of a *closed queueing network*. We shall not be concerned with such networks in this thesis.[4]

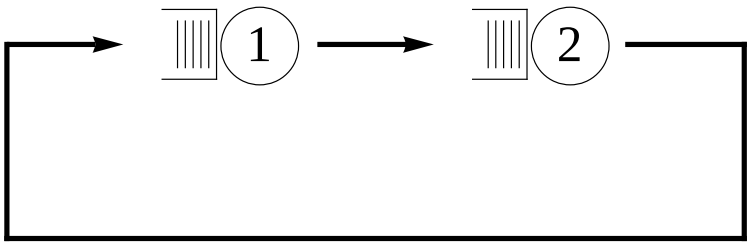

Figure 1.10: The closed version of the 2 station tandem queue.

We now wish to formulate mathematical models that describe the evolution of queueing networks over time. For example, we would like to describe the number of jobs in each buffer (including the job which is in service) at time t by the processes $Q_k(t)$ where k is the buffer index. We shall do so formally soon, but for starters let us look at the 2 station tandem queue example. In the next section we show how the above network is solved when the primitive sequences are exponential and in the following two sections we shall discuss approximations for the case of general processing times.

Assume that the processing times in the 2 station tandem queue of Figure 1.9 are given by the two sequences $\{X_k(\ell), \ell = 1, 2, \ldots\}$, $k = 1, 2$. Here, $X_k(\ell)$ is the processing time of the ℓ'th job in station k. For the example, we shall assume that processing times are constant: $X_1(\ell) = 1$,

[4] Actually, when the processing times are assumed to be i.i.d. exponential, the network in Figure 1.10 is equivalent to the M/M/1/K queue which we analyze in Chapter 6. An extension of this network is analyzed in Boxma (1988).

$X_2(\ell) = 1.5$ for all ℓ. Further data that is required is the arrival process. Here assume that the arrival times of jobs to the first node (from the outside) are at times,

$$0, 1, 3, 4, 6, 7, 9, 10, 12, 13, 15, 16, 18, 19, 21, 22, \ldots$$

These can be described by an arrival process $A_1(t)$ which counts how many arrivals have occurred up to time t and by inter-arrival times $\{X_0(\ell),\ \ell = 1, 2...\} = \{0, 1, 2, 1, 2, 1, 2, 1, 2, \ldots\}$ (note that $X_0(0)$ is the time of the first arrival and not an inter-arrival time), and the relation

$$A_1(t) = \sup\{n | \sum_{\ell=1}^{n} X_0(\ell) \leq t\}.$$

Finally, assume that at time 0 the network is empty: $Q_1(0) = Q_2(0) = 0$. We now have enough data to determine the evolution of $Q_k(t), k = 1, 2$ for all time t. We do so systematically in Figure 1.11. We first plot $A_1(t)$, the arrival process into the first node. We then take into account the fact that the server 1 requires 1 unit of time to service each job and obtain the queue realization $Q_1(t)$. Now we look at the output of the first node and use it to construct the process $A_2(t)$. We then use this process to build $Q_2(t)$. The resulting network evolution (or realization) is in Figure 1.11.

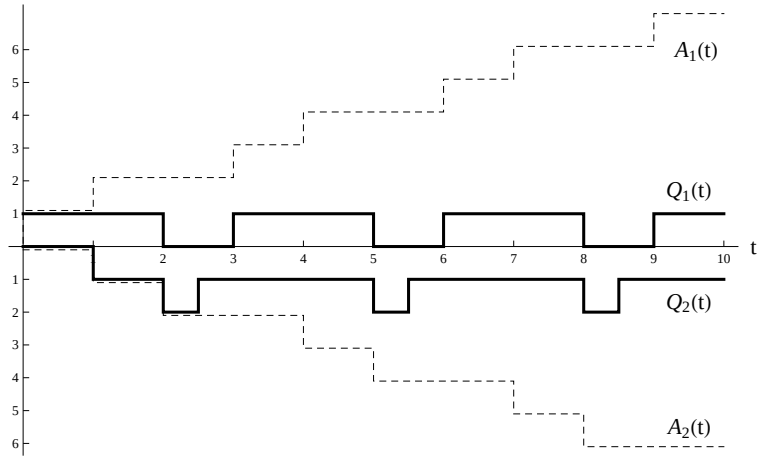

Figure 1.11: Deterministic realization of 2 station tandem queue. The processes $A_1(t), Q_1(t)$ are shown above the x-axis and the processes $A_2(t), Q_2(t)$ are below the x-axis. Note that at times $1, 4, 7$ and 10 the queues experience 2 simultanious events (arrival and service completion).

We shall often refer to the sequences $X_k(t)$ (including the inter-arrival times) as *primitive sequences*. In a sense, a queueing network is a mapping that transforms these primitive sequences into realizations of the form $Q_k(t), A_k(t)$. As in the analysis of single server queueing systems, one usually employs stochastic primitive sequences which are usually assumed to be i.i.d. A further simplifying assumption is the so-called "Kleinrock assumption" which states that processing times of jobs at different nodes depend (possibly) on the node but not on the job. This is a pretty strong assumption when one considers packet communication networks because usually the packet size is directly proportional to the transfer time on links (processing times) and varies from packet to packet but does not change from link to link. Nevertheless, without this simplifying assumption, the analysis is typically intractable. In the continuation of this section we explicitly define this "mapping" for a quite general class of networks

In general we are interested in queueing networks that are arbitrarily complex (yet have a finite amount of nodes and buffers). For an illustration look at Figure 1.12. This network has 4 nodes and the arrows indicate that customers may take several possible routes. For example, the 3 arrows that point out from node 1 indicate that customers that complete service at this node can move to either nodes 2, 3 or 4. Customers that complete service at node 2 may either leave the network or more to further service at node 3. These customers later return to node 2. This type of "re-entrant" behavior often yields interesting analysis. In certain cases when one assumes its absence (in which case the network is called *feed-forward*) the analysis simplifies.

One way to specify the routing that actually takes place is by introducing additional primitive sequences. This time we shall use indicator sequences $\{\phi_{ij}(n), n = 1, 2, \ldots\}$ to indicate the routing that occurred in the network. Specifically $\phi_{ij}(n) = 1$ if the n'th customer out of node i moved to node j, otherwise we have $\phi_{ij}(n) = 0$. Note that $\sum_j \phi_{ij}(n) = n$. In the next section we shall assume probabilistic assumptions regarding these sequences. An alternative to the primitive sequences is to say that the routing is part of the network policy. This is sometimes called *discretionary routing*.

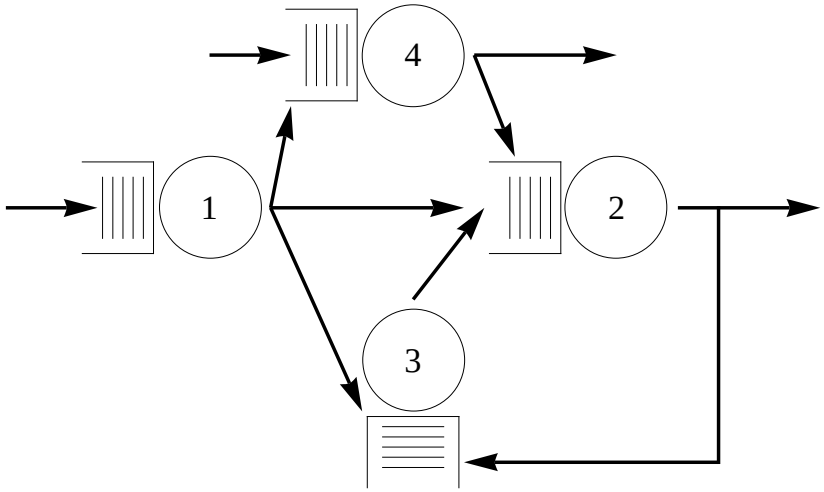

Figure 1.12: A single-class queueing network.

Suppose we want to add the following characteristic to our network: Jobs that arrive to node 2 from nodes 1 or 3 are to depart the network after being serviced while jobs that arrive from node 4 are to move to node 3 after being serviced. To do so we may label the jobs as 'a' type jobs and 'b' type jobs and keep track of this labeling of the jobs in the queue of node 4. This is illustrated in Figure 1.13.

This type of network is called a classed network or *multi-class queueing network* (MCQN) because customers are grouped into classes (in this case 'a' and 'b'). For clarity, we shall also assume that all jobs in nodes 1 and 3 are 'a' type and all jobs in node 4 are 'b' type, but this is not critical, since we have not specified different behaviors for jobs in 1, 3 or 4. The important thing is that we allow jobs to change their classes, for example, when a job moves from node 2 to node 3 we say it changes from being a 'b' type job to being an 'a' type job. This is opposed to *single-class* (also sometimes called Jackson-type) networks which do not distinguish between

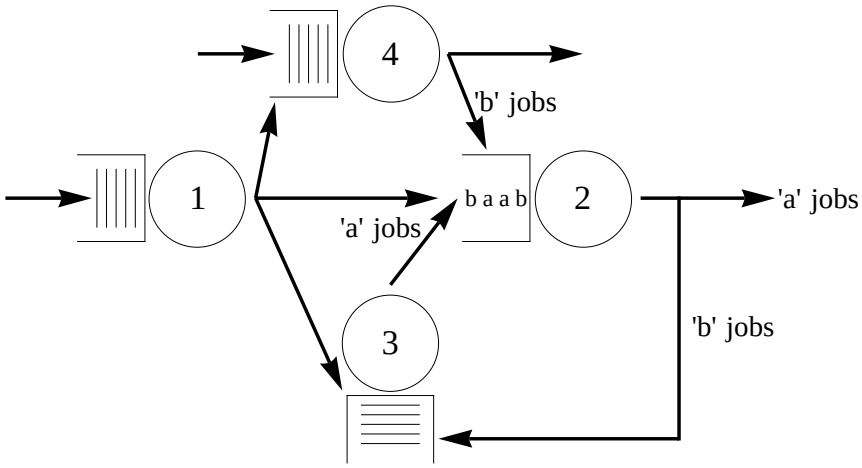

Figure 1.13: A multi-class queueing network

jobs in a node. A second distinction is that in a multi-class network, we may assume different probabilistic assumptions on the primitive sequences of different classes.

The illustration of the jobs in node 2 in figure 1.13 hints that the order of the jobs in the queue is important. This is indeed the case if the scheduling policy of node 4 is FCFS. As an alternative, suppose that we wish to give full priority to 'a' type jobs over 'b' type jobs. In this case a more illustrative drawing of the node 2 is as in right side of Figure 1.14. Here we actually associate a queue with every class and group the two classes 'a' and 'b' by the resource, or node which is drawn as a rectangle. We read this figure as implying that at any time, work can only be performed on a single job at a time and the server has to choose 'a' or 'b'. In general, this is the type of representation which we shall use for queueing networks in this thesis.

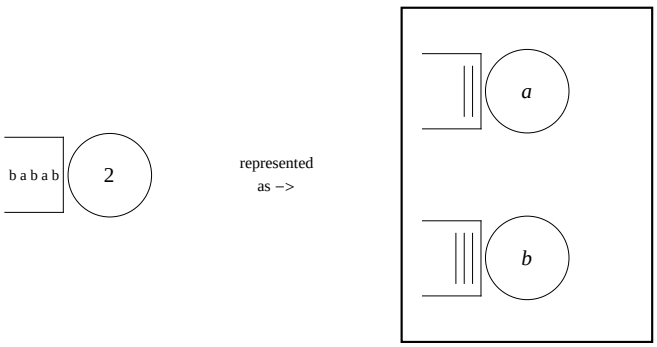

Figure 1.14: An alternative representation of a multi-class node

An example of a simple multi-class queueing network is in Figure 1.15. This is a type of a *re-entrant line* network, i.e. a network composed of one route in which jobs may re-enter nodes. We shall use this example for the analysis of our method of finite horizon control in Chapter 3.

Control Policies

The illustration of the node on the right hand side of Figure 1.14 shows that a multi-class queueing network requires a specification of a *control policy*: at every time instance, which class

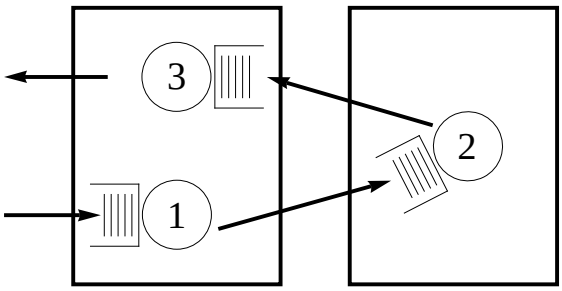

Figure 1.15: A Simple 3 Buffer Re-entrant line

should be served: 'a' or 'b'? One can distinguish between control policies that are *local*, each node makes information only based on local information, e.g. number of jobs in each queue. A *global* control takes into account information regarding the whole network, e.g. queue levels at other nodes.

A recent book which summarizes some of the advances in the field is Meyn (2008), another reference which mainly deals with control using diffusion approximations (Section 1.6) is Kushner (2001). A more classic approach is covered in Chapter 8 of Walrand (1988). Our results of Chapter 3 make heavy use of the *maximum pressure* control policy, see that chapter for details. In other chapters, we analyze a specific network under some simple specified control rules. In this respect, there is no attempt to show that the control is optimal, but rather show that it has some desired properties. In general there are still many un-answered questions regarding control of queueing networks and how to do so effectively.

Formulation of a Multi-class Queueing Network Model

We shall now formally define a MCQN. This model was essentially introduced in Harrison (1988). The network consists of $k \in \mathcal{K} = \{1, \ldots, K\}$ job-classes and $i \in \mathcal{I} = \{1, \ldots, I\}$ servers. Jobs of class k queue up in buffer k, and we let the queue length $Q_k(t)$ be the number of jobs of class k in the system at time t. We let $Q_k(0),\ k \in \mathcal{K}$ be the initial queue lengths. Buffer k is served by server $\sigma(k)$, and the constituency of server i is $C_i = \{k \mid \sigma(k) = i\}$. In general a server may serve several classes, i.e. $|C_i| > 1$, hence the term multi-class. The topology of the network is described by the $I \times K$ constituency matrix $\mathbf{A}$ with elements $A_{ik} = 1$ if $k \in C_i$, $A_{ik} = 0$ otherwise.

For $\ell = 1, 2, \ldots$, the ℓ's job out of buffer k requires processing amount $X_k(\ell)$, after which the job may either leave the system or move to another buffer. $S_k(t) = \sup\{n \mid \sum_{\ell=1}^{n} X_k(\ell) \leq t\}$ counts the number of jobs completed at buffer k by processing for a total time t. $\phi_{kk'}(\ell)$ is the indicator of the event that the ℓ's job out of buffer k moved into buffer $k' \in \mathcal{K} \setminus k$. Let $\Phi_{kk'}(n) = \sum_{\ell=1}^{n} \phi_{kk'}(\ell)$, this is a count of the number of jobs routed from buffer k to k' out of the first n jobs served at buffer k.

We further assume an inter-arrival sequence $X_0(\ell),\ \ell = 1, 2, \ldots$, with,

$$E(t) = \sup\{n \mid \sum_{\ell=1}^{n} X_0(\ell) \leq t\}.$$

This sequence is partitioned to the nodes according to the Bernulli sequences $\phi_{0k}(\ell)$ such that $E_k(t) = \phi_{0k}(E(t))$.

The MCQN is controlled by allocating processing times to the buffers. Let $T_k(t)$ be the total time allocated to buffer k by server $\sigma(k)$, during $[0, t]$. Then the dynamics of the queues are described by:

$$Q_k(t) = Q_k(0) + E_k(t) - S_k(T_k(t)) + \sum_{k' \in \mathcal{K} \setminus k} \Phi_{k'k}(S_{k'}(T_{k'}(t))).$$

The allocated times have to satisfy:

$$T_k(0) = 0, \quad T_k(t) \text{ non decreasing}, \quad \sum_{k \in C_i} T_k(t) - T_k(s) \leq t - s \text{ for all } s < t \text{ and each } i \in \mathcal{I}.$$

In particular each $T_k(t)$ is Lipschitz continuous with constant 1, and its derivative $\dot{T}_k(t)$ exists for almost all t, and satisfies:

$$\mathbf{A}\dot{T}(t) \leq \mathbf{1}, \quad \dot{T}(t) \geq 0, \quad 0 < t < T, \tag{1.3}$$

where $\mathbf{1}$ denotes a vector of 1's.

Additional constraints have to be satisfied by $T_k(t),\ Q_k(t),\ k \in \mathcal{K}$. First and foremost, $Q_k(t) \geq 0$ and no processing can occur when $Q_k(t) = 0$. We also assume that servers cannot be split so each server can work on only one job at a time. In addition we assume, that jobs are not preempted. Hence for almost all t, $\dot{T}_k = 0$ or $\dot{T}_k(t) = 1$, and $\dot{T}_k(t)$ can only change from 1 to 0 when $S_k(T_k(t))$ has a jump of 1, that is when the processing of a job is completed.

The above network formulation is now quite standard, see (Bramson, 2008, Chapter 1) for an introduction. The idea of using accumulating processes is quite novel and is especially useful since the allocation process T is Lipschitz. It is especially important to realize that for a given primitive sequence, different network controls yield different realizations of $T(\cdot)$ and thus different network realizations.

Other Types of Queueing Networks

There are many variations that one can employ for modeling queueing networks. One such variation is to assume finite buffers. In this case we need to specify the behavior of the network when a job arrives to a buffer that is full. One option is for the job to get lost (this is common in the Internet where routers have finite capacity). A survey regarding such networks is Kelly (1991). A further option is for blocking to occur. In this case congestion in downstream buffers causes upstream servers to stop working. These types of models are often appropriate for manufacturing systems. A survey of these types of results is in Balsamo *et al.* (2001). Other types of networks are also possible. For example one can assume fork-join type behavior as in Baccelli *et al.* (1989). Another alternative is for "negative cutomers" - such customers enter queues and wipe out other customers, cf. (Artalejo, 2000). Such networks are also considered in the algebraic study of queueing networks: Dao-Thi and Mairesse (2006).

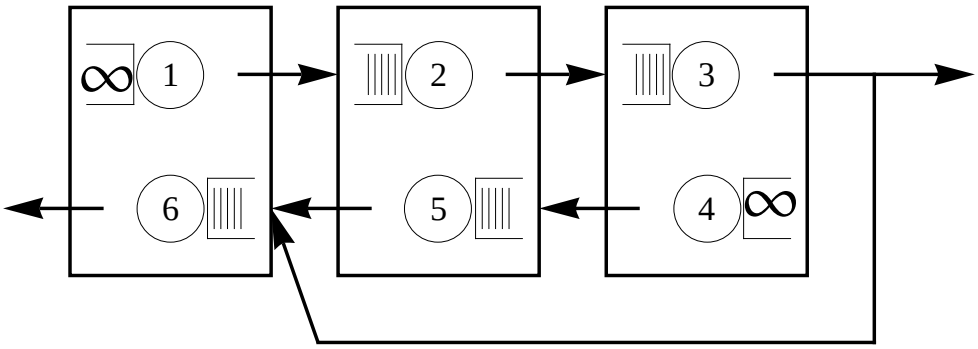

Figure 1.16: An example of a multi-class queueing network with infinite virtual queues.

A major extension to the MCQN model is the ability to perform discretionary routing. Here the policy does not only sequence the jobs but also decides which routes should be taken. An early study of such networks is in Kelly and Laws (1993). A rather general model which employs some of the above characteristics is called a *stochastic processing network*. This network model proposed in Harrison (2000), Harrison (2002) and Harrison (2003) is currently receiving a lot of attention. A stochastic processing network generalizes the multi-class queueing network by allowing job splitting and merging, discretionary routing and the joint use of resources to perform activities. The idea is to define 4 types of entities: jobs, buffers, activities and resources. Activities contend for resources by operating on jobs that are placed in buffers. Once an activity is complete jobs move from the "input buffers" of the activity onto the "output buffers" of the activity. A control is a rule for allocation of resources to activities. The generality is due to the allowed many-to-many relationship between activities and buffers and between activities and resources.

Yet another alternative to queueing network modeling is to assume that there is an infinite supply of jobs for some as implied for classes 1 and 4 in Figure 1.16. We call such networks, networks with *infinite virtual queues*. These networks play a major role in this thesis and are further reviewed in Chapter 2.

1.4 Product Form Miracles

In this section we shall outline the results and main ideas of what we call "classic queueing networks". We use this term to refer to queueing networks that exhibit a so-called product form solution. These queueing networks enjoy the incredible attribute that their steady state probability vector decomposes into a product of probabilities, each for a different queue. Thus, in some steady state time, t_0, the queue size random variables are mutually independent.

Up to the early 80's most queueing network research has concentrated on these types of networks with the state of the art probably being the release of Kelly's book in 1979 "Reversibility and Stochastic Networks" (Kelly, 1979). Other notable and useful sources are Serfozo (1999), Walrand (1988), Chen and Yao (2001, Chapters 2 and 4) and Bramson (2008, Chapter 2).

As an example, we shall start by analyzing the 2 station tandem network presented in the

previous section. Assume now that arrivals to the network are according to a rate α Poisson process and further assume that the service times at station/queue $k = 1, 2$ are i.i.d. sequences with exponential distribution and mean μ_k^{-1}. The arrival process, and service sequences are assumed independent. We also assume that $\alpha < \mu_k$, $k = 1, 2$. This turns out (as may be expected) to be the necessary and sufficient condition for stability, more on that in Section 1.7.

The first station is a stable $M/M/1$ queue and is not affected by the second station. Now in steady state, it is an elementary exercise to show that the process $Q_1(\cdot)$ is a time reversible Markov process. This implies that its reversed process is stochastically equivalent to $Q_1(\cdot)$. As a consequence the arrival process into the reversed process is also a Poisson process with rate α. But each arrival in the reversed process actually corresponds to a departure of the its reversal (a non-reversed process) so this implies that the output process is a Poisson process! This remarkable miracle is known as Burke's theorem (Burke, 1956)[5] and is a fundamental result of queueing theory. A further consequence of the reversibility is that departures prior to any time t_0 are independent of $Q_1(t_0)$. This is because these departures are matched by the arrivals of the reversed process.

And now for a "corollary to the miracle": Let us denote the output process from the first node by $A_2(t)$, it is also the arrival process into the second node. We now know it is a Poisson process. Thus the queue at the second queue is also an M/M/1 queue. Further, the arrivals it experiences up to time t_0 are the departures of the first queue and are independent of $Q_1(t_0)$. This immediately implies that $Q_2(t_0)$ and $Q_1(t_0)$ are independent and they are each distributed as in equation (1.2) with $\rho_k = \alpha/\mu_k$, $k = 1, 2$. Thus, we have:

Theorem 1.1. *The stationary distribution of the 2 station tandem network with rate α Poisson arrivals and i.i.d. exponential processing times having rates $\mu_k > \alpha$, $k = 1, 2$ is given by $P(n_1, n_2)$ as follows:*

$$P(n_1, n_2) = \prod_{k=1}^{2} (1 - \frac{\alpha}{\mu_k})(\frac{\alpha}{\mu_k})^{n_k}, \ n_k = 0, 1, 2, \ldots$$

Classic queueing network theory is all about extending the theorem above as far as possible. A major early achievement in this respect is the *Jackson network* (Jackson, 1957, 1963). In brief: a Jackson network is a single-class network of exponential single-class nodes with Bernoulli routing (the nodes may actually have state dependent service rates), an example is illustrated in Figure 1.17.

A *Jackson network* is a special case of the much more modern MCQN presented in the previous section. There are K buffers (each buffer is also a node). We assume that the processing time primitives are exponential with mean μ_k^{-1} and the arrival process to the network is a Poisson with rate α (when any of these assumption are violated the network is sometimes called a *generalized Jackson network*). In the context of a MCQN, we have:

$$X_k(\ell) \sim \begin{cases} \exp(\alpha) & k = 0 \\ \exp(\mu_k), & k = 1, \ldots, K \end{cases}, \qquad \text{i.i.d. for } \ell = 1, 2, \ldots$$

[5] Burke did not use the simple reversibility argument to prove this.

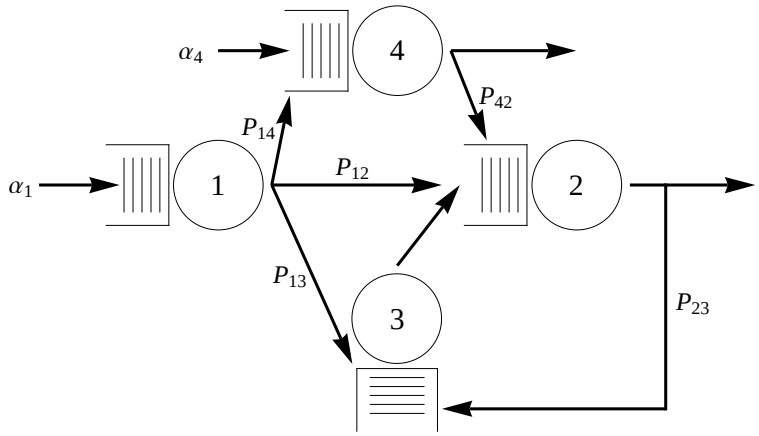

Figure 1.17: A Jackson Queueing Network.

We further assume that the routing primitives are Bernoulli i.i.d. with probabilities P_{ij}. To analyze this network we need to formulate the *traffic equations*. These are equations for the unknowns, $\lambda_1, \ldots, \lambda_k$ which symbolize the net input and output rates of jobs to each of the nodes:

$$\lambda_k = \alpha P_{0k} + \sum_{j \neq k} \lambda_j P_{jk}. \tag{1.4}$$

If we represent the net input rates as the vector λ, The routing probabilities between nodes in the square matrix, $\mathbf{P}$, and the exogenous input rates in the vector α (with $\alpha_k = \alpha P_{0k}$) then the above equations have a unique solution, $\lambda = (\mathbf{I} - \mathbf{P}')^{-1}\alpha$, if and only if $\mathbf{P}$ has a spectral radius less than 1, we shall assume this holds. The miracle of Jackson networks is that in steady state, the joint distribution of the queue length of all nodes is:

$$P(n_1, \ldots, n_k) = \prod_{k=1}^{K} (1 - \frac{\lambda_k}{\mu_k})(\frac{\lambda_k}{\mu_k})^{n_k}, \ n_k = 0, 1, 2, \ldots.$$

The above holds if and only if $\rho_k = \frac{\lambda_k}{\mu_k} < 1$ for each node. Thus, as in the special case of the 2 station tandem queue, at the instant of steady state, the queue lengths of all nodes are independent and are distributed as if they were M/M/1 queues. Note that this is a different type of "miracle" (and not a corollary to the previous one) because the decomposition proof of Theorem 1.1 no longer holds and typically the flows between nodes are not Poisson processes (they are Poisson when the network is acyclic).

The Jackson network was greatly extended to a multi-class setting with several types of policies in Baskett *et al.* (1975). Today these networks are called BCMP after their authors. A further generalization which appears to be nearly the state of the art was in Kelly (1975) and Kelly (1976), and the book Kelly (1979). These types of networks are generally called *Kelly type networks*. A Kelly network consists of a set of customer classes and a set of nodes. Nodes are multi-class and may be of two forms, *homogeneous* or *symmetric*.

A homogeneous node essentially models stations in which the service policy is FCFS and processing times are exponentially distributed with the same mean for all customer classes.

Variations of the FCFS are allowed but these do not include policies such as priority policies based on class.

A symmetric node allows phase-type distributions (cf. Breuer and Baum (2005)) which essentially approximate any distribution. Several types of policies and configurations are possible: $M/G/1-PS$ (processor sharing) queue, $M/G/1-LCFS$ (last come first served), $M/G/\infty$ infinite servers (good for modeling a delay line in the network). In this case different customer classes may have different means. Again, variations are allowed but these do not include FCFS service or priority based policies. Symmetric nodes satisfy a condition known as *quasi-reversibility* which is characterized by the property that the input process into a node up to time t_0 is independent of the current state and the output process out of the node after time t_0. It can be shown that in this case the arrival and departure processes are Poisson.

The main result (miracle) of Kelly networks is that homogeneous and symmetric nodes may be combined with arbitrary customer classes and job routes to obtain a multi-class network that is modeled by a Markov process with a product form stationary distribution that is explicitly known. See Bramson (2008, Chapter 2) and Walrand (1988, Chapter 3).

1.5 Network Decomposition Heuristics

If one attempts to upgrade the models of the previous section only slightly, explicit exact results are usually unobtainable. For example, take the 2 station tandem queue that we analyzed in the previous section and set the service distribution of the first server to some non-exponential distribution. This will result in the first queue being an M/G/1 FCFS queue (solvable), but the second queue now has an arrival process that makes it more complicated than the solvable GI/M/1 queue[6] because the inter-departure times of the first queue are typically no-longer independent. Further, if one replaces the service time of the second server also by some 'G' then the second server is typically more complicated than a $GI/G/1$ queue and it appears that finding an explicit exact solution is hopeless.

Another type of enrichment to the previous models is to look at a multi-class queueing network under policies other than the ones described in the previous section. In particular, one common choice is a static buffer priority policy. Going in this direction, again one ventures into the land of models that typically lack an explicit steady state solution. For example, the simple re-entrant line in Figure 1.15 is currently unsolved under the LBFS or FBFS policies, even when processing times are assumed to be exponential, cf. Kumar (1993). Similarly if one considers finite capacity buffers with or without blocking then the analysis is typically intractable except for some special cases[7].

As a consequence of these hardships, one now has to resort to approximations. Actually, most of modern queueing theory deals with approximations that are justified by some type of limit theorem. For example, stochastic process limits as presented in Whitt (2002) or large deviation principles as in Ganesh *et al.* (2004). In this respect one judiciously decides to use an

[6] 'GI' in Kendall's notation stands for arrivals according to a renewal process.

[7] If all buffers are finite and processing times are phase type, the network may in principle be explicitly solved. But then one quickly meets the "curse of dimensionality".

approximation, knowing that it is exact in some asymptotic sense (some times the asymptotic result is only conjectured), we outline such approximations in the next section. An alternative viewpoint regarding approximations that is somewhat less scientific and often more engineering oriented is to use heuristic approximations that are not fully justified by theory, yet seem to yield good results. We now briefly survey a major branch of such network approximations: *Network Decomposition and Traffic Process Approximations*. This line of queueing network approximations is usually associated with the phrase, QNA (Queueing Network Analyzer) which is a method and software package that popularized them, (Whitt, 1983b,a).

The general idea behind network decomposition and traffic process approximations is to decompose the network into subsets of nodes (often with a single node in each subset), characterize or approximate the traffic processes between these decomposed subsets and approximate the queue levels at each node assuming that it is fed by the approximating traffic processes. By traffic processes we mean the flows of jobs into and out of nodes (also sometimes overflows from the nodes in the case of finite queues). The approximation thus relies on 3 types of assumptions:

1. An assumption that the network may be decomposed.
2. An assumption regarding the traffic processes that originate from the node.
3. An assumption regarding the performance of each individual node, given some input traffic processes and service times.

An Example

We shall now illustrate one variant of this approximation approach on the 2 server tandem queue. Assume that arrivals are according to a renewal process with inter-arrival times uniformly distributed on the range $(0.5, 2.05)$. Let the processing times of the first server be i.i.d. exponential with mean 1.0 and let the processing times of the second server be uniform on the range $(0.1, 1.9)$. Note that we have chosen the parameters such that the offered load on both servers is $\rho = 1/1.05 = 0.952$.

As a first approximation to this example, we can "relax" the uniform distributions and assume that the arrival process is Poisson and processing times of the second server are also exponential. Under this assumption, Theorem 1.1 characterizes the steady state distribution of the queue sizes as:

$$P(n_1, n_2) = 0.002268 \times 0.952^{n_1+n_2}, \;\; n_1, n_2 = 0, 1, 2, \ldots.$$

The resulting mean number of jobs in each queue is easily computed to be 20 thus the total mean is $L_{Jackson} = 40$. Here we used an "exact analysis" that wrongfully assumed that the "rather deterministic" arrival process was Poisson and that the processing time were exponential. A measure that shows how strong these assumptions are, is to look at the squared coefficients of variation (SCV) which is the variance divided by the mean of the inter-arrival and processing times. Specifically, we have

$$C_a^2 = 0.3023, \quad C_1^2 = 1.0 \quad C_2^2 = 0.27,$$

where C_a^2 is the inter-arrival SCV and C_k^2 is the SCV of processing times at node k. As two of these are quite far from the exponential random variable SCV which is 1, it is quite probable that modeling this network as a Jackson network may yield results that are far from realistic.

We shall now employ the QNA method as described in Whitt (1983b) to this example. A key phrase in the introduction of that paper is *"A natural alternative to an exact analysis of an approximate model is an approximate analysis of a more exact model"*. Our more exact model will take all 3 SCVs into account. QNA assumes that the network may be decomposed in the sense that each node may be analyzed separately once the traffic processes into it have been characterized. For our simple network this assumption is actually valid. For more complex networks with re-entrant traffic flows or conflicting routes, this may be a very strong assumption. Further, the QNA assumes that all traffic processes in the network are renewal processes (see also Whitt (1982)). In our simple example this is exact for the first node but an approximation for the second. In more complicated networks one further needs to assume that when these renewal processes are merged, the resulting process is again renewal (see also Albin (1984)). QNA models the expected waiting time (in queue), $\bar{W}$, in each node as

$$\bar{W} = \frac{\bar{x}\rho}{1-\rho}\frac{C_{in}^2 + C_s^2}{2}g(\rho, C_{in}^2, C_s^2),$$

where $\bar{x}$ is the mean service time, ρ is the offered load at the node as calculated by the set of equations (1.4), C_{in}^2 is the SCV of the inter-arrival time, C_s^2 is the SCV of the service times and,

$$g(\rho, C_{in}^2, C_s^2) = \begin{cases} \exp[-\frac{2(1-\rho)}{3\rho}\frac{(1-C_{in}^2)^2}{C_{in}^2+C_s^2}], & C_{in}^2 < 1 \\ 1, & C_{in}^2 \geq 1 \end{cases}$$

This is an approximation of a GI/G/1 queue. Note that it is consistent with the (P-K) formula when $C_{in}^2 = 1$ as presented in Section 1.2, for further details see Whitt (1982).

Applying this approximation to the first node, we obtain $\bar{W}_1 = 12.86$. Since the mean inter-arrival time is 1.05, by Little's law we obtain that the mean number of jobs waiting in Q_1 is 12.25. We add to this, the mean number of jobs in service which is 0.952, to obtain a mean number of jobs in node 1 of 13.2.

Now we continue and analyze the second node, for that we first need to approximate the traffic process departing from node 1. This is a process with mean rate 0.952. We approximate it as a renewal process having SCV of inter-arrival times C_d^2 using the following approximation:

$$C_d^2 = \rho^2 C_s^2 + (1-\rho^2)C_{in}^2,$$

where ρ is the traffic intensity on the first node. In our example we obtain $C_d^2 = 0.935$. We now again employ the GI/G/1 approximation to the second node as we did for the first and obtain mean number of jobs in the node (queue + service) of 12.38. As a result, our approximation for the mean number of jobs in the system is $L_{QNA} = 25.6$.

Compare the QNA result to $L_{Jackson}$: Taking the SCV's into consideration almost halved our assessment of the mean number of jobs in the system. Note also that when SCVs are set to 1, the QNA yields the exact results of a Jackson network.

Since we are not able to perform an exact analysis of this network, we turn to simulation (see Appendix A) to asses what is the exact value. We simulated this network for 10^6 time units and obtained $L_{Sim} = 22.3$. It thus appears that the QNA approximations is quite good.

More on Network Decomposition Heuristics

We have only shown how to use the QNA approximation on a simple example that does not involve, merging and splitting of traffic streams. In general the idea of QNA is to first solve (exactly) the traffic equations (1.4) and then to solve a second set of linear equations for the SCV's of the traffic processes. Briefly: The idea is to approximate the SCV of a renewal process that results from Bernoulli thinning with probability p as $C^2_{out} = pC^2_{in} + (1-p)$ and to approximate the SCV of a renewal process that is a superposition of renewal processes as a weighted average of the SCVs of the input processes where the weights are given by the traffic rates.

In general, there is agreement that QNA seems to give good results for Jackson type networks but performs much more poorly on general MCQNs. This is especially obvious when one considers the behaviour of simple MCQNs using priority policies. Some proposed refinements to QNA that still make use of the renewal approximation are in Whitt (1994), Whitt (1995), Caldentey (2001), and Araghi and Balcıoglu (2008). Further refinements to the network decomposition approach that use a more complicated traffic process than a renewal process are Bitran and Dasu (1993) and Balcıoglu *et al.* (2008).

Network decomposition of networks with finite queues is considered in Haverkort (1995), Sadre *et al.* (1999), Heindl and Telek (2002) and Mitchell and van de Liefvoort (2003). Here the traffic processes are known to be quite correlated and thus the approximations resort to approximating them as processes that may capture some of the correlation structure of the traffic processes. It is possible that some of the results of our thesis (BRAVO effect presented in Chapter 6), may be relevant to this line of work. We have not explored this any further.

1.6 Diffusion Approximations

Diffusion approximations provide a more theoretically robust alternative to the heuristic approximations of the previous section. Such approximations were first applied in the queueing context in the late 60's and early 70's by Kingman, Borvokov, Iglehart and Whitt and since then have enjoyed great popularity. A diffusion process is a Markov process having continuous sample paths, the most typical of which is Brownian motion. Diffusion approximations of queueing systems often assume some sequences of queueing systems, each with its own stochastic queue length or workload process indexed by $n = 1, 2, \ldots$ and employ a weak convergence of these processes to some limiting process which is shown to be diffusive. The limiting process is often a mapping of Brownian motion. Thus a diffusion result is usually some sort of weak functional limit theorem. Such results (in their modern form) first appeared in Iglehart and Whitt (1970) for the context of single station queues. We now briefly outline this type of result.

A Diffusion Limit for Single Server Queues

Consider a a sequence of standard GI/G/1 systems indexed by $n = 1, 2, \ldots$. Inter-arrival times in the n'th system are distributed as a random variable U_n having mean $1/\lambda_n$ and SCV $c_{a,n}^2$. Service times in the n'th systems are distributed as a random variable V_n with mean $1/\mu_n$ and SCV $c_{s,n}^2$. Define $\rho_n = \lambda_n/\mu_n$. Now consider the case where $\rho_n \to 1$ as $n \to \infty$. Look at the sequence of stochastic process $\{Q_n(\cdot), n \geq 1\}$, where $Q_n(t)$ is the number of jobs in the system at time t in the n'th process. A "heavy-traffic" diffusion is performed by letting $n \to \infty$ and looking at a limiting diffusive process of this sequence.

For $n = 1, 2, \ldots$ and $0 \leq t \leq T$ where $T < \infty$, let

$$\hat{Q}^n(t) = \frac{Q^n(nt)}{\sqrt{n}}.$$

For each n we are thus defining a random process $\hat{Q}^n(t)$ on the time interval $[0, T]$ based upon sample paths of the process $Q^n(\cdot)$ over the expanded time interval $[0, nT]$ and the normalizing factor $n^{1/2}$. For $d \in (-\infty, \infty)$ and $\sigma^2 \in (0, \infty)$ let $\{B(t), t \geq 0\}$ be a Brownian motion process with drift parameter d and variance coefficient σ^2. Now look at the *reflected Brownian motion*:

$$R(t) = B(t) - \inf\{B(s), 0 \leq s \leq t\}.$$

And let $R_T(\cdot)$ denote the restriction of $R(\cdot)$ to $[0, T]$. The following result is due to Iglehart and Whitt (1970)[8]:

Theorem 1.2. *Suppose that as $n \to \infty$ we have $(\lambda_n - \mu_n)n^{1/2} \to d$, $\lambda_n \to \lambda$, $\mu_n \to \mu$, $c_{a,n}^2 \to c_a^2$ and $c_{s,n}^2 \to c_s^2$ where each of the limiting values λ, μ, c_a^2 and c_s^2 is positive and finite. Further assume that $\mathbb{E}\left[(U_n)^{2+\epsilon}\right]$ and $\mathbb{E}\left[(V_n)^{2+\epsilon}\right]$ are uniformly bounded in n for some $\epsilon > 0$. Then for all initial values $\{Q_n(0), n \geq 1\}$,*

$$\hat{Q}^n(\cdot) \Rightarrow R_T(\cdot) \text{ as } n \to \infty,$$

where $R_T(\cdot)$ is reflected Brownian motion on $[0, T]$ with drift coefficient d and variance parameter $\sigma^2 = \lambda c_a^2 + \mu c_s^2$.

Since the value of T is arbitrary we can interpret the result of the above theorem as saying that as $n \to \infty$, the process $\gamma_n Q^n(t/\gamma_n^2)$ converges weakly to $R(\cdot)$ where $\gamma_n = (\lambda_n - \mu_n)/d$. In particular, for each t, the distribution of $\gamma_n Q^n(t/\gamma_n^2)$ converges to the distribution of $R(t)$. Now one usually employs the following result:

$$\lim_{t \to \infty} P(R(t) \leq x) = 1 - e^{-2|d|x/\sigma^2}, \quad x \geq 0.$$

The above result is useful for approximating the behavior of a GI/G/1 queue in heavy traffic. To utilize the approximation take a GI/G/1 system with parameters λ, μ, c_a^2 and c_s^2. Now determine the value of n by the difference between λ and μ. As a result, the steady state queue size is approximately exponentially distributed with expectation:

$$\frac{1}{(1-\rho)} \frac{\rho c_a^2 + c_s^2}{2}.$$

[8] That paper actually considers more general multi-server systems.

Much more can be said about diffusion approximations of single station queues (cf. Whitt (2002)). We now move to briefly discuss networks.

A Diffusion Limit for the 2 Station Tandem Queue

By the late 70's, it was apparent that queueing networks can be approximated by diffusion processes under the heavy-traffic regime. The framework for this was laid down by a series of papers by Harrison with the most notable one being Harrison (1978) which showed that under a heavy traffic condition the 2 station tandem queue has a diffusion limit. A useful early survey is Lemoine (1978). Other more contemporary sources for approximations of queueing networks are Chen and Yao (2001) and Kushner (2001).

Harrison presented a diffusion approximation for the waiting time in the 2 station tandem queue: Let the inter-arrival times have mean a and variance u^2, and let the service times at station $i, i = 1, 2$ have mean b_i and variance v_i^2. Let W_j^i denote the waiting time at station i of the j'th arriving customer. Also denote $d = \min\{(a - b_1), (a - b_2)\}$. In order for the network to be stable we need $d > 0$. We will look at the network in heavy-traffic, i.e. d is positive but close to 0. The result of Harrison shows that under heavy traffic conditions, the vector dW is distributed approximately as a certain two-dimensional random vector W^* which is defined as a certain functional of three-dimensional Brownian motion. Further, the distribution of W^* depends only the first and second moments of the service and inter-arrival times.

The precise formulation is in terms of a limit theorem for a sequence of tandem systems with $d \to 0$. Assume that as $n \to \infty$ we have $d_n \to 0$, $u_n^2 \to \sigma_0^2$, $v_{i,n}^2 \to \sigma_i^2$ and $(a_{i,n} - b_{in})/d_n \to c_i$ for $i = 1, 2$ where $\sigma_0, \sigma_1, \sigma_2, c_1$ and c_2 is positive and finite. Now under similar uniform boundedness conditions to the theorem above, we have that as $n \to \infty$, the random vector $W(n)$ converges in distribution to a random vector W^* whose distribution depends only on the parameters $\sigma_0, \sigma_1, \sigma_2, c_1$ and c_2. The limiting process, W^*, is typically called reflected Brownian motion in an orthant (cf. Harrison and Reiman (1981); Dai and Harrison (1992); Taylor and Williams (1993)). It's analysis has posed great challenges to applied probabilists. We shall not discuss it any further.

The diffusion approximations which we present in this Thesis (Chapter 7) are of a much simpler nature: We do not look at a sequence of systems with changing parameters but rather simply scale time by n and space by $n^{1/2}$. Also, our limiting processes turn out to be simple Brownian motions as opposed to functionals of Brownian motion.

More General Networks

Following the 2 server tandem queue, the theory of diffusion approximations of queueing networks has evolved greatly in the past 30 years. A notable publication is Reiman (1984) which establishes a diffusion approximation for a generalized Jackson (single-class) network. A further pair of papers are Bramson (1998b) and Williams (1998) which lay a framework for diffusion approximations of multi-class queueing networks. Many other works which are not mentioned here are at the forefront of contemporary queueing network research today.

Another similar trend in modern queueing theory is the study of queueing networks under "many - server scaling" which has also grown to be known as the Halfin-Whitt regime (Halfin and Whitt, 1981). This type of diffusion approximation has been especially fruitfull in queueing applications related to call-centers (cf. Gans *et al.* (2003)) Our results are not of this nature.

1.7 Instability Surprises

Positive Harris recurrence is one of the notions of stability that is used in this thesis, especially in Chapter 4. It implies that the associated Markov process of the network posses a stationary distribution. Another notion of stability, *rate stability* implies that there is no linear build up of queues over time, see Chapter 3. When we say a queueing network is *stable* we imply that an associated Markov process is positive Harris recurrent. When we say a queueing network is *unstable* (not stable) we imply that it is not rate stable.

As demonstrated in Section 1.1, queues appear to be stable when $\rho < 1$, rate stable when $\rho \leq 1$ and unstable when $\rho > 1$. Furthermore, the networks analyzed in Section 1.4 are all known to posses a stationary distribution when $\rho < 1$ for each node and are thus stable. In fact, it has only recently been proven that a generalized Jackson network is positive Harris recurrent if and only if $\rho < 1$ for all nodes (Sigman, 1990; Meyn and Down, 1994; Baccelli and Foss, 1994).

For a long time it was believed that $\rho < 1$ implies stability of any queueing network under any work conserving policy. This belief was shattered with the discovery of a simple clever example of a queueing network that is not stable even when $\rho < 1$ for all nodes. This network is called the Kumar-Seidman-Rybko-Stoylar Network, (Kumar and Seidman, 1990; Rybko and Stolyar, 1992). The KSRS network has revolutionized queueing network theory because it illustrated that there exist multi-class queueing networks with quite sensible work conserving policies that are unstable, even when there are enough resources to handle all of the input. Following the discovery of KSRS there have been other examples of similar networks. An important example is a network with a FCFS policy in Bramson (1994) that is unstable. This example emphasized that in the multi-class setting, even the most naive policy can cause problems. A survey of several additional examples is in Bramson (2008).

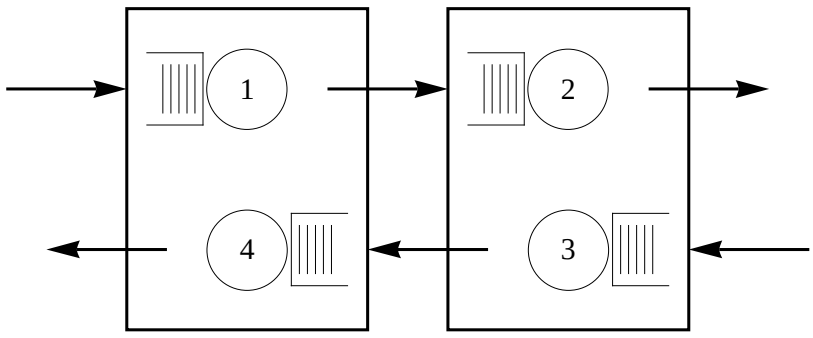

Figure 1.18: The KSRS Queueing Network.

The KSRS network is illustrated in Figure 1.18. The policy of interest is the natural policy to use a last buffer first serve (LBFS) priority rule: the left server gives priority to step 4 and

the right server gives priority to step 2. This policy is "natural" because it is a greedy policy for reducing the queue sizes. The distinction between preemptive and non-preemptive policy is not important. The surprise of KSRS is that in certain cases, this policy may lead to under-utilization of the resources and in turn to instability.

Denote the processing rates (inverse of mean) of step k by μ_k , the input rate to buffer 1 by α_1 and the input rate to buffer 3 by α_3. In that case the offered loads for the servers are $\alpha_1/\mu_1 + \alpha_3/\mu_4$ and $\alpha_3/\mu_3 + \alpha_1/\mu_2$. And a necessary condition for stability is that both offered loads are less than 1.

Observe that whenever $Q_4(t) = 0$, the stream $\cdot \to 1 \to 2 \to \cdot$ behaves like a 2 station tandem queue until buffer 2 becomes empty. This is because the right server gives priority only to 2, thus accumulating jobs in 3 at rate α_3 and not passing any jobs to 4. While it is ensured that 2 will become empty in a finite time (because the offered load is less than 1), during this time it is possible that an excessive amount of jobs will accumulate in 3. Similarly for the case where $Q_2(t) = 0$. It is thus apparent that buffers 2 and 4 will typically not operate at the same time: the network essentially alternates between periods in which stream $\cdot \to 1 \to 2 \to \cdot$ is in operation and periods in which stream $\cdot \leftarrow 4 \leftarrow 3 \leftarrow \cdot$ is in operation. The coupling between buffers 2 and 4 hints that another condition for stability is that the *virtual server* composed of these buffers needs to have an offered load less than 1:

$$\rho_v = \frac{\alpha_1}{\mu_2} + \frac{\alpha_3}{\mu_4} < 1.$$

It turns out that the virtual server really imposes a necessary condition for stability. We demonstrate with a numerical example:

$$\alpha_1 = \alpha_2 = 1, \quad \mu_1 = \mu_3 = 3, \quad , \mu_2 = \mu_4 = \frac{20}{11} \approx 1.82$$

In this case the offered loads of both servers are 0.883 (less than 1) but we have that $\rho_v = 1.1$. Figure 1.19 plots a realization of this network with deterministic processing times [9]. Indeed in this case the network appears to be unstable, alternating between "busy periods" of a 2 station tandem queue and during each period too much work accumulates in the opposite stream. Similar realizations appear with stochastic processing times [10]. We give some more details on KSRS in Chapters 4 and 7 where we compare it to the push-pull network that we analyze.

The discovery of this type of unstable queueing network behavior motivated the search for finding criteria for stability of a queueing network. A landmark paper, on which our thesis relies heavily is Dai (1995). It outlines a framework for proving stability of a queueing network based on a corresponding fluid model. A comprehensive summary of this framework is in Bramson (2008). While Dai's method is of great theoretical importance, there are still many unanswered questions regarding stability of queueing networks: There is still not an efficient method to systematically find the stability region of a queueing network with respect to an arbitrary policy.

[9] To observe the effect with deterministic processing times, the initial conditions need to be asymmetric.

[10] An animated demonstration of KSRS is in *The Queueing Science Exploratorium*: `http://www.stat.haifa.ac.il/~yonin/qsm/main.html`.

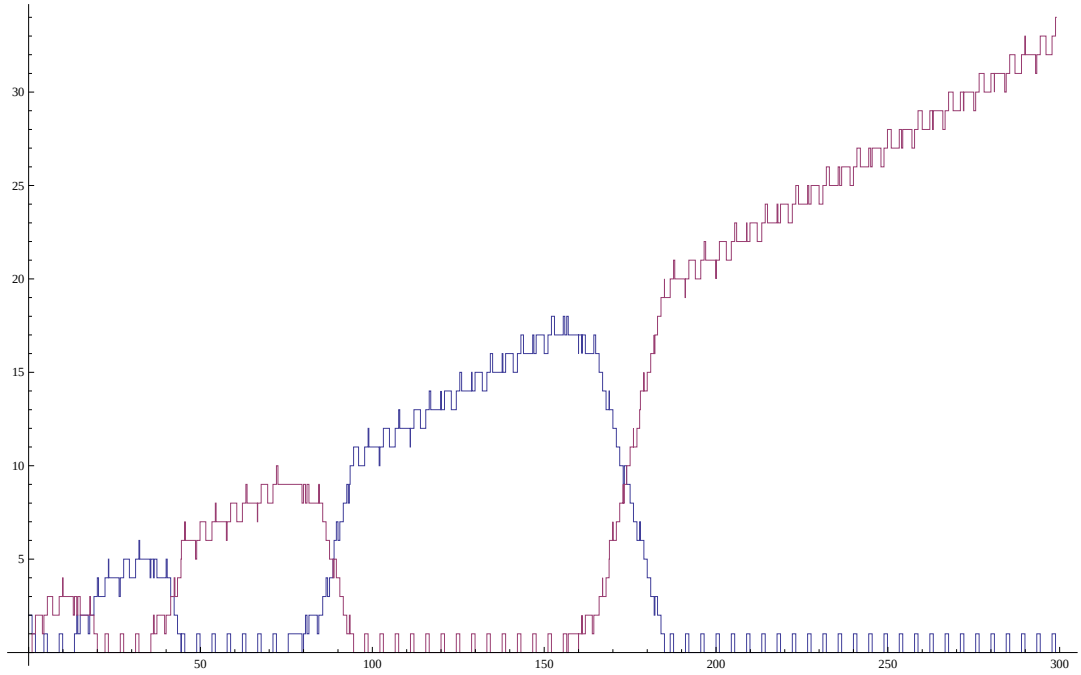

Figure 1.19: Realization of the KSRS example with deterministic processing times. The blue curve is $Q_1(t) + Q_4(t)$. The red curve is $Q_2(t) + Q_3(t)$.

CHAPTER 2

INFINITE VIRTUAL QUEUES

Typically queueing models have exogenous arrival processes. Such are the product form and multi-class queueing networks that were surveyed in the previous chapter. While this type of modeling often makes sense, incorporating an input stream in the model may sometime be unnecessary. For example, when modeling a manufacturing system that is geared to operate at full capacity, it may be extremely superficial to assume that it is driven by some exogenous stream of orders because often in the short or medium range the backlog of orders does not drain. Another example is a communication system in which a transmitter has a constant supply of messages generated on the spot in addition to serving messages in transit from other transmitters. We call such "piles" of infinite supplies, *infinite virtual queues* (IVQs). Surprisingly, queueing models with IVQs have not received much attention in the literature. In this short chapter we survey the few literature results that deal with such networks.

In Section 2.1 we motivate the concept of infinite virtual queues. In Section 2.2 we survey results regarding examples of such networks that are "Jackson-type". In Section 2.3 we survey results from the literature regarding possibly the simplest non-trivial queueing network with infinite virtual queues, the 3 buffer infinite supply re-entrant line. We continue in Section 2.4 where we briefly review the few known results regarding general re-entrant lines with infinite virtual queues. In Section 2.5 we introduce the push-pull network and survey some previous results about it. This network will be further analyzed in Chapters 4, 5 and 7.

2.1 Motivation

A schematic representation of an IVQ is in Figure 2.1(a). All that is symbolized in the figure is the fact that the server always has a job to work on. Output from the IVQ will be at rate μ if the server never idles. IVQs may be used to model exogenous arrivals into a network, as in the single server queue model in Figure 2.1(b). This semantic representation of arrival processes has been used by some authors, e.g. Dai and Lin (2005). When viewed as a controlled queueing network, models with IVQs have a potentially richer control space than models with exogenous arrivals because now the arrival processes may be controlled (e.g. turned on and off).

Figure 2.1: (a) An infinite virtual queue. (b) An infinite virtual queue feeding a regular queue.

Things start to get especially interesting with IVQs when we use models in which a control decision is to be made with regards to either serving an IVQ or serving jobs from some finite queue. This can be observed by considering the control decisions that are available for the (multi-class) network in Figure 2.2(a). Here we have two servers and jobs which are processed first by server 1 at rate μ_1, and next by server 2, at rate μ_2. An infinite supply of raw jobs will keep server 1 busy all the time, and produce a stochastic input at rate μ_1 into the second server. To fully utilize the second server we now assume that $\mu_2 > \mu_1$, and add another IVQ of jobs which need processing only on server 2, at rate μ_3.

The network will now produce two output streams of jobs, stream 1, of jobs that are processed by both servers, and stream 2 which is processed only by server 2. This network will fully utilize both servers with no idling, and produce output of stream 1 at rate μ_1 and of stream 2 at rate $\mu_3(1-\frac{\mu_1}{\mu_2})$, if we use the following type of control: Whenever the queue of jobs of stream 1 that wait for server 2 is exhausted, server 2 will switch to stream 2, and will continue processing it for a duration which is a stopping time with finite expectation. The queue of jobs of stream 1 between the two servers will in that case be positive recurrent. In fact server 2 will behave like a server with vacations (see Levy and Yechiali (1975)). The corresponding queueing network (See Figure 2.2(b)) with two random arrival streams of rates α_1, α_2 will, under a similar vacation policy, produce outputs at rates $\min(\alpha_1, \mu_1, \mu_2)$ and $\min(\alpha_2, \max(0, \mu_3(1-\frac{\mu_1}{\mu_2})))$, and be subject to idling and congestion.

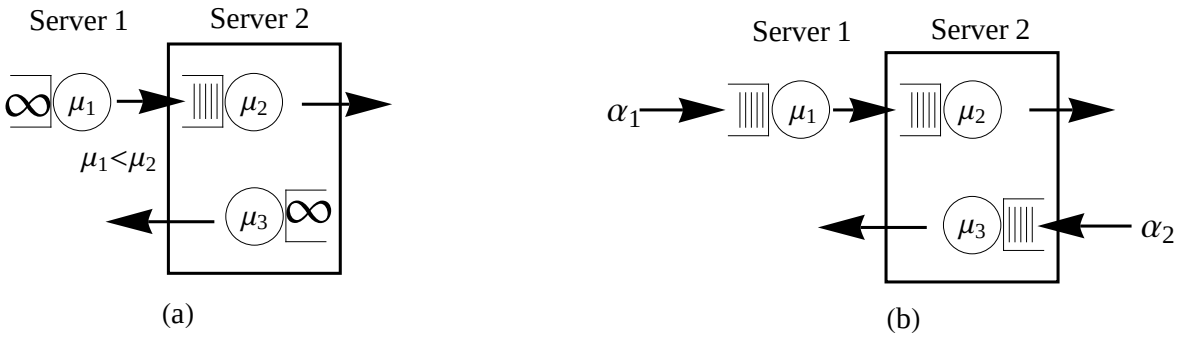

Figure 2.2: Systems with infinite supply of work versus exogenous arrivals

A notable property of the network in 2.2(a) is that the suggested policy achieves full utilization, no idling, and no congestion. We conjecture that the same can be done for a large class of general multi-class queueing networks as well as for more general stochastic processing networks: If each resource has an infinite supply of work then, under appropriate conditions, there exist policies which fully utilize all the resources and which keep all the standard queues pos-

itive recurrent, and these policies can be implemented without knowing the exact processing rates of the various activities.

To be specific, we envision the IVQs to be part of the network, and processing of items out of these IVQs consumes some of the network's resources. We assume here that each resource of the network has, among the queues which it is processing, at least one IVQ. In that case none of the resources ever needs to idle. In the operation of the network we assume that each of the IVQs has a nominal processing rate at which it introduces items into the network, and each of the output streams has an output rate at which items are produced. These determine the rates at which each input activity, intermediate activity, and output activity is performed, and the offered load of each resource. The network is *rate stable* if each of the standard queues (i.e. not IVQs) in which items are stored in intermediate stages has equal input and output rates, so that there is no linear accumulation of material in these queues. A network operates in *balanced heavy traffic* if it is rate stable, and if all the resources are fully utilized. In other words, the input and output rates are such that the offered load to all the resources is equal to $\rho = 1$.

When inputs to the network are exogenous and subject to stochastic variability, a rate stable network in balanced heavy traffic is always congested. As the offered load approaches 1 items accumulate at a rate of $\Theta(\sqrt{t})$. In this case the network may behave as a semi-martingale reflected Brownian motion on the diffusion scale.

In networks with infinite supply of work the situation is radically different: because inputs are not exogenous but are produced by processing of IVQs within the network, we have much more control to cope with stochastic fluctuations and with congestion. Hence we conjecture that such networks can be operated under balanced heavy traffic, with full utilization of all the resources, and yet show no congestion at all: Under appropriate conditions there exists a wide range of policies that achieve 0 idling in the network, and keep all the queues which are not IVQs positive recurrent.

2.2 A Jackson-Type Network with IVQs

The short paper Weiss (2005), introduced Jackson type networks with infinite virtual queues. The model consists of I nodes where each node has a service rate μ_i and exogenous input rate α_i. The routing probabilities are $P_{ij}, i, j = 1, \ldots, I, i \neq j$. A subset of the nodes E contain lower priority infinite virtual queues which means that whenever the queue at this node is empty an item from the infinite supply is processed. The priority scheme is preemptive. Weiss assumes that all processing times are independent exponential random variables so it is not necessary to specify if the preemption is preemption-resume or preemption-restart. The traffic equations of this network are:

$$\lambda_i = \alpha_i + \sum_{j \in \bar{E} \neq i} \lambda_j P_{ji} + \sum_{j \in E \neq i} \mu_j P_{ji}$$

These traffic equations are easily solved (in matrix form) and Weiss shows that a necessary and sufficient condition for stability is $\lambda_i \leq \mu_i$. A key observation regarding this network is that each of the nodes $i \in E$ works non-stop on processing items for independent and identically exponentially distributed times at rate μ_i. Thus departures from these nodes are independent

Poisson streams. A result of this "Poisson departure" property is that the subnetwork of nodes $i \in \bar{E}$ behaves like a regular Jackson network with Poisson inputs and thus has a steady state distribution as described in Section 1.4. It is further shown that arrivals to each of the nodes $i \in E$ are Poisson and each of these nodes has a marginal distribution like an M/M/1 queue. An interesting point is shown in a companion paper, Adan and Weiss (2005), where by analyzing an example, Adan and Weiss show that while the nodes in E have marginal geometric distributions, the joint distribution is not product form.

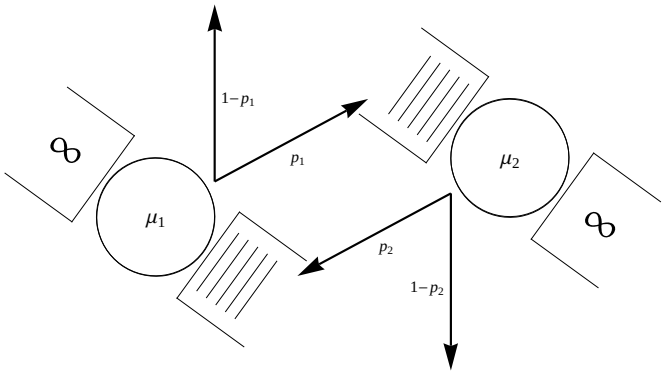

Figure 2.3: The Jackson-type network with IVQs Analyzed in Adan and Weiss (2005).

The example analyzed in Adan and Weiss (2005) is in Figure 2.3. Their main result is that when $\rho_1, \rho_2 < 1$, the stationary distribution for $(n_1, n_2) \neq (0, 0)$ is given by an infinite sum of product forms:

$$P(n_1, n_2) = \sum_{k=1}^{\infty} (-1)^{k+1} [(1-\alpha_k)\alpha_k^{n_1}(1-\beta_{k+1})\beta_{k+1}^{n_2} + (1-\alpha_{k+1})\alpha_{k+1}^{n_1}(1-\beta_k)\beta_k^{n_2}]$$

where for $k \geq 1$:

$$\alpha_{k+1}^{-1} = \frac{\mu_1+\mu_2}{\mu_2 p_2}\beta_k^{-1} - \alpha_{k-1}^{-1} - \frac{1-p_2}{p_2}, \quad \beta_{k+1}^{-1} = \frac{\mu_1+\mu_2}{\mu_1 p_1}\alpha_k^{-1} - \beta_{k-1}^{-1} - \frac{1-p_1}{p_2}$$

and we initialize the sequence with $\alpha_0 = \beta_0 = 1$ and $\alpha_1 = \rho_1, \beta_1 = \rho_2$. Further expressions are given for $P(0,0)$, for the distribution of the sum of the queues, the joint factorial moments and for the queue length correlation. Their derivation of the steady state probabilities is performed by analyzing the detailed balance equations and using the method of compensations introduced in Adan *et al.* (1993).

2.3 The 3 Buffer Infinite Supply Re-Entrant Line

The *3 buffer infinite supply re-entrant line* is pictured in Figure 2.4. It has been analyzed in the literature quite extensively with respect to the last buffer first serve (LBFS) priority policy. This policy is as follows: The first server gives priority to activity 3 whenever $Q_3(t) > 0$, otherwise it works on activity 1 (the IVQ). The second server serves jobs sequentially or idles when $Q_2(t) = 0$.

Under LBFS, server 1 operates continuously, the situation of server 2 is different and we need to distinguish between 3 cases:

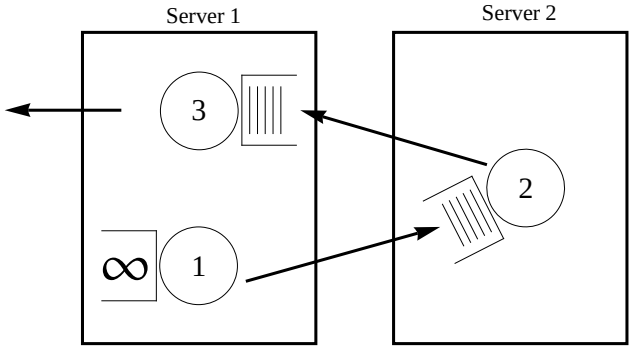

Figure 2.4: The 3 buffer re-entrant line with infinite supply

Server 1 bottleneck ($m_1 + m_3 > m_2$): The traffic intensity on server 2 is $\frac{m_2}{m_1+m_3} < 1$.

Server 2 bottleneck ($m_1 + m_3 < m_2$): The arrival rate of jobs to server 2 is $\geq \frac{1}{m_1+m_3} > \frac{1}{m_2}$. Hence the traffic intensity on server 2 is > 1.

Servers balanced ($m_1 + m_3 = m_2$): The traffic intensity on server 2 equals 1.

The stability (in the sense of positive Harris recurrence) of this model for the server 1 bottleneck case was first handled in Weiss (2004) for i.i.d. exponential processing times and later in Guo and Zhang (2006) where the fluid stability framework of Dai (1995) was employed for general processing times[1]. The instability of the server 2 bottleneck case is shown in Guo (2008) as an example of an application of a criterion for instability of a re-entrant line with an infinite virtual queue.

Further, Guo and Zhang (2006) shows that for exponential processing times, a necessary condition for stability is $m_1 + m_3 > m_2$. Guo (2008) shows that the system is not stable if $m_1 + m_3 < m_2$ for general processing times.

Adan and Weiss (2006) were able to take the stability results further and obtained the joint steady state distribution of queue 2 and queue 3 (under the exponential processing times assumption). Their expression for the steady state distribution of (n_2, n_3) is:

$$P(n_2, n_3) = \begin{cases} \frac{m_1}{m_1+m_3}(1-\alpha_2)\alpha_2^{n_2}\alpha_3^{n_3}, & n_2 > 0, n_3 \geq 0 \text{ or } (n_2, n_3) = (0,0) \\ \frac{m_1}{m_1+m_3}(1-\alpha_2)\alpha_3^{n_3-1}, & n_2 = 0, n_3 > 0 \end{cases}$$

where,

$$\begin{aligned} \alpha_2 &= \frac{\mu_1}{\mu_2} \frac{-\mu_1 - \mu_2 + \mu_3 - \sqrt{(\mu_1+\mu_2+\mu_3)^2 - 4\mu_1\mu_3}}{2\mu_3}, \\ \alpha_3 &= \frac{\mu_1 + \mu_2 + \mu_3 - \sqrt{(\mu_1+\mu_2+\mu_3)^2 - 4\mu_1\mu_3}}{2\mu_3}. \end{aligned}$$

An interesting fact that stems from this steady state distribution is that the marginal distributions are equal to stationary distributions of corresponding G/M/1 queues with the appropriate traffic intensities. This is currently unexplained. Adan and Weiss (2006) continue their analysis with some sample path properties, monotonicity results and also obtain stationary distributions for the more involved non-preemptive case (which is slightly more involved).

[1] Guo and Zhang (2006) also show stability for the case of exponential processing times using Foster Lyapounov and random walk techniques similar to Weiss (2004).

2.4 The General Infinite Supply Re-Entrant Line

The model of the previous section is a simple example of a *re-entrant line* with K consecutive steps on I servers where the first step is an IVQ. As in a MCQN, we denote the server of step k by $\sigma(k)$ and $C_i = \{k : \sigma(k) = i\}$. One sensible family of policies to use for such a network is a preemptive priority policy that only serves jobs from the IVQ at time t, if $Q_k(t) = 0$ for all $k \in C_1$. A quite general analysis of this policy is in Guo and Zhang (2007) and Guo (2008) [2]. In the first paper, the authors adapt the fluid stability framework of Dai (Dai, 1995) for this model. They then show that the fluid models for several policies are stable and thus they show that under some technical conditions the network represented as a continuous time general state space Markov process is positive Harris recurrent. The second paper (Guo, 2008) shows that under the sensible condition of $\rho_i > 1$ for some server i, the network is unstable. This paper adapts the results of Dai (1996). For this network, ρ_i is defined as follows:

$$\rho_i = \frac{\sum_{k \in C_i} m_k}{\sum_{k \in C_1} m_k}.$$

This is sensible because every job that enters the system should perform all steps on processing station 1 and requires $\sum_{k \in C(1)} m_k$ time units from it. Station 1 operates continuously at full capacity and thus the number of jobs it processes in the system per time unit is $\lambda := \frac{1}{\sum_{k \in C(1)} m_k}$. For other stations $j \neq 1$, the number of jobs that are processed when the station is fully utilized is $\mu_j := \frac{1}{\sum_{k \in C(j)} m_k}$ per time unit. Thus for the system to be weakly stable we should intuitively expect that $\rho_j = \frac{\lambda}{\mu_j} \leq 1$. An additional paper, Guo and Yang (2007), illustrates the Positive Harris recurrence results for a 5 buffer example.

In Chapter 7 of this Thesis, we present a diffusion limit for the output process of this re-entrant line.

The case of "sub-bottlenecks"

Our Master's Thesis, (Nazarathy, 2001, Chapter 5) also discusses the general infinite supply re-entrant line. In that work we discuss the operation under the LBFS policy without requiring the condition $\rho_i < 1$. In that case we give a deterministic iterative algorithm that partitions the buffer indexes into L groups as follows:

$$\{1, \ldots, K\} = \{a^{(L)}, \ldots, K^{(L)}\} \cup \ldots \cup \{a^{(1)}, \ldots, K^{(1)}\}$$

Here all of the buffers in the group $\{a^{(\ell)}, \ldots, K^{(\ell)}\}$ are upstream buffers to the *bottleneck server of the group*: K^{ℓ}. We conjecture that under LBFS, these upstream buffers are positive recurrent while the bottleneck server of each group grows without bound.

2.5 The Push-Pull Network

The *push-pull network* is pictured in Figure 2.5, it consists of two servers, numbered 1, 2 and two types of jobs numbered 1, 2 each of which is processed by both servers. Type 1 is processed by

[2] Actually those papers only claim to handle static buffer priority policies (SBP) of this type but it appears that their analysis is not restricted to SBP.

server 1 and then by server 2, while type 2 is first processed by server 2 and then by server 1. We call the first step of each type a *push activity* and the second step a *pull activity*. We denote the finite queues of the pull activities, $Q_i(\cdot), i = 1, 2$. The mean duration of the push activities are $\lambda_i^{-1}, i = 1, 2$ and the mean durations of the pull activities are $\mu_i^{-1}, i = 1, 2$. There is no arrival stream, arrivals are generated by the push activities from the IVQs.

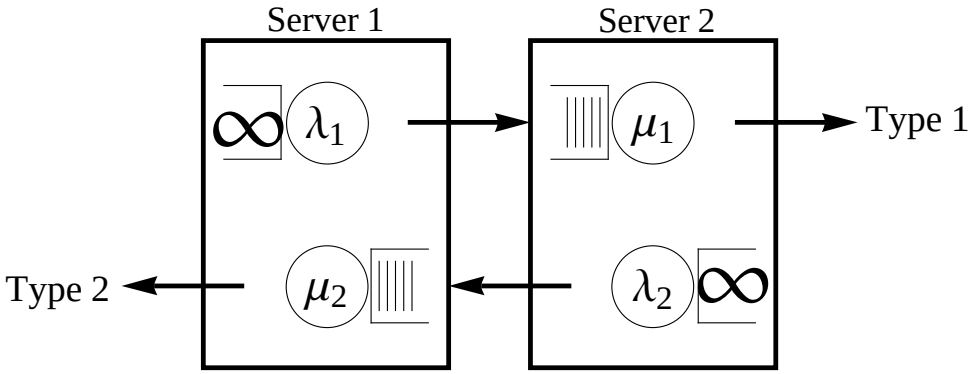

Figure 2.5: The push-pull queueing network.

This network is quite similar to the KSRS network, shown in Figure 1.18, a comparison of the two is in Chapter 4 and further detailed comparisons along with some simulations are in our publication, Kopzon *et al.* (2008), which is not fully included in this thesis. As stated in Section 2.1 a novelty of queueing networks with IVQs is that servers that have an IVQ are never forced to idle due to emptiness. The push-pull network is special in this respect because in certain cases there are simple policies that may control it under full utilization while still maintaining finite stochastically bounded queues. This was first shown in Kopzon and Weiss (2002), is further explored in Kopzon *et al.* (2008) and expanded to the case of general processing times in Nazarathy and Weiss (2008c) as covered in this thesis in Chapter 4. We shall now review the results of Kopzon and Weiss (2002) and Kopzon *et al.* (2008).

Consider some policy in which both servers are working all the time, and assume that this policy is rate stable i.e. input rates equal output rates at all the queues. Denote by $\theta_i, i = 1, 2$ the long run average fraction of time that server i is working on jobs of type i, in a push operation. Since servers are working all the time, $1 - \theta_i, i = 1, 2$ is the long run average fraction of time that server i is working on jobs of type $3 - i$, in a pull operation. If the network is to be rate stable then we have the following equations for the production rates ν_i of jobs of types $i = 1, 2$:

$$\begin{aligned} \nu_1 &= \theta_1\lambda_1 = (1-\theta_2)\mu_1, \\ \nu_2 &= \theta_2\lambda_2 = (1-\theta_1)\mu_2, \end{aligned}$$

which are solved by

$$\begin{aligned} \theta_i &= \tfrac{\mu_i(\mu_{\bar{\imath}}-\lambda_{\bar{\imath}})}{\mu_1\mu_2-\lambda_1\lambda_2}, \\ \nu_i &= \tfrac{\mu_i\lambda_i(\mu_{\bar{\imath}}-\lambda_{\bar{\imath}})}{\mu_1\mu_2-\lambda_1\lambda_2}, \end{aligned} \qquad i = 1, 2,\ \bar{\imath} = 3 - i. \tag{2.1}$$

One needs to distinguish between several cases:

Inherently stable network: When $\lambda_i < \mu_i, i = 1, 2$, service of each type of jobs alone, by its second server, is a stable single server queue.

Inherently unstable network: When $\lambda_i > \mu_i, i = 1, 2$, service of each type of jobs alone, by both servers results in an unstable single server queue.

Unbalanced network: When $\lambda_1 > \mu_1$ and $\lambda_2 < \mu_2$, then server 2 has more work to do than server 1, for both types of jobs, and the network cannot be stable unless server 1 idles some of the time. Similarly for $\lambda_1 < \mu_1$ and $\lambda_2 > \mu_2$. We do not consider this case any further.

Completely balanced network When $\lambda_i = \mu_i, i = 1, 2$ it is possible to find policies which work with full utilization of both servers, and which are rate stable, i.e. they satisfy $\nu_1 = \nu_2$ and $\nu_3 = \nu_4$, however these rates are not uniquely determined. We can choose $0 \leq \theta \leq 1$, and specify $\theta_1 = \theta_2 = 1 - \theta_3 = 1 - \theta_4 = \theta$, and use $\nu_i = \mu_i \theta_i$ as nominal rate. As shown in Nazarathy and Weiss (2008b), we can use an adaptation of the maximum pressure policy of Dai and Lin (2005) to serve jobs of types 1 and 2 at these rates, under full utilization. However, the system will become congested, with expected $O(\sqrt{T})$ jobs in the system at time T. We conjecture that this cannot be improved.

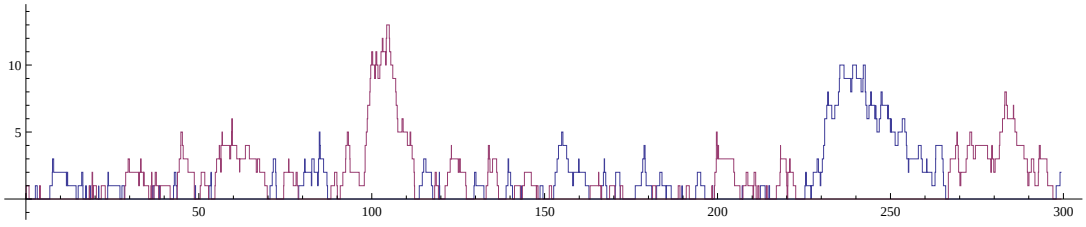

Figure 2.6: Queue sizes realization of "pull-priority" in the preemptive case: alternating single server busy periods. $\lambda_1 = \lambda_2 = 0.8$, $\mu_1 = \mu_2 = 1$. $Q_1(t)$ – Red. $Q_2(t)$ – Blue.

The results that we survey in this section assume that all processing times are i.i.d. exponential random variables[3].

Control Policies for the Inherently Stable Case

A quite sensible control for the inherently stable case is to give priority to pull over push (i.e. operate using LBFS). This method yields realizations that look like "geometrically alternating busy periods of single server queues", see Figure 2.6. If preemption is allowed then the busy periods always start with an empty system. If preemptions are not allowed then busy periods start with a random amount of customers in the queue as in classical vacation models (Wolff, 1989, Chapter 10). The steady state distribution for the simpler preemptive case is used as a motivating opening theorem in Kopzon *et al.* (2008). The more involved non-premptive case is in the earlier paper Kopzon and Weiss (2002). We now summarize.

[3]Actually, Kopzon and Weiss (2002) also analyzed the more general M/G/. case for the inherently stable network under pull priority without preemption. Similar (even simpler) analysis may also be applied to the preemptive case and has not been done yet.

Denote by (n_1, n_2, I_1, I_2) the state of the network as follows: n_i are the number of jobs in the pull queue of type i. I_i is 1 when there is a non-premptable job in service of the push activity i, it is 0 otherwise. Then the for the non-preemptive case the set of recurrent states is:

$$(+,0,0,0) \cup (+,0,0,1) \cup (0,+,0,0) \cup (0,+,1,0),$$

where we use $+$ to indicate that a coordinate is strictly positive. All these states communicate. For the preemptive case the set of recurrent states is

$$(0,0,0,0) \cup (+,0,0,0) \cup (0,+,0,0),$$

again all states communicate. The stationary distributions on the above states spaces are

$$P(n_1,0,0,I_2) = \begin{cases} \frac{\lambda_1(\mu_1-\lambda_1)(\mu_2-\lambda_2)}{(\mu_1\mu_2-\lambda_1\lambda_2)(\lambda_1+\lambda_2-\mu_2)} \times ((\frac{\lambda_2}{\mu_2})^{n_1} - (\frac{\lambda_2}{\lambda_1+\lambda_2})^{n_1}) & I_2 = 0 \\ \frac{(\mu_1-\lambda_1)(\mu_2-\lambda_2)}{(\mu_1\mu_2-\lambda_1\lambda_2)} \frac{\lambda_1}{\lambda_1+\lambda_2} (\frac{\lambda_2}{\lambda_1+\lambda_2})^{n_1} & I_2 = 1 \end{cases}$$

for the preemptive case with a symmetric expression for $P(0, n_2, I_1, 0)$. And,

$$P(n_1,0,0,0) = \frac{(\mu_1-\lambda_1)(\mu_2-\lambda_2)}{\mu_1\mu_2-\lambda_1\lambda_2} \left(\frac{\lambda_1}{\mu_1}\right)^{n_1}$$

for the non-preemptive case with a symmetric expression for $P(0, n_2, 0, 0)$.

Control Policies for the Inherently Unstable Case

Control of the inherently unstable case is not as simple, in this case pull priority policies yields transient behavior. Nevertheless, a class of fully utilizing, stable policies is proposed and analyzed in Kopzon *et al.* (2008), the analysis is largely due to the Ph.D thesis of Kopzon (2006). Only the preemptive case is considered. The class of policies is called "Generalized Threshold Policies", they are essentially threshold policies where each server observes the queue of the other server and maintains the jobs in that queue above a certain threshold. Stability of the preemptive case is considered under exponential processing times using Lyapounov functions. Further results include steady state analysis of two special cases which we detail below: *Fixed threshold policies* and the *queue balancing policy*.

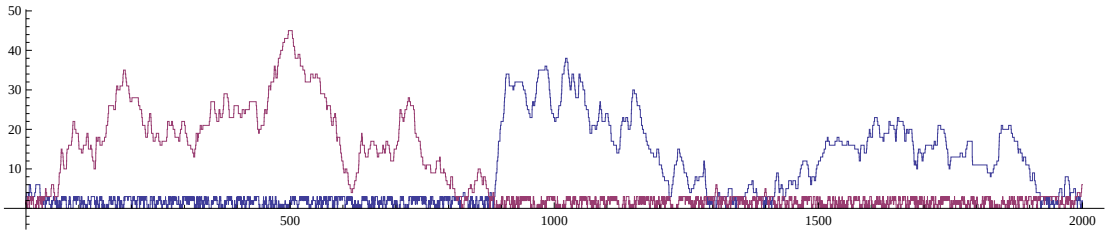

Figure 2.7: Queue sizes realization of the fixed threshold policy. $\lambda_1 = \lambda_2 = 1.25$, $\mu_1 = \mu_2 = 1$. $s_1 = s_2 = 3$, $Q_1(t)$ – Red. $Q_2(t)$ – Blue.

Fixed threshold policies: This policy is based on two fixed positive integer thresholds, s_1 and s_2 that are interpreted as follows: Server 1 only pulls (from Q_2) when Q_1 has at least s_1 jobs

and Q_2 is not empty. Similarly, server 2 only pulls (from Q_1) when Q_2 has at least s_2 jobs. The resulting CTMC has one recurrent communicating class:

$$\{(n_1, n_2) : n_1 \geq s_1, 0 \leq n_2 \leq s_2\} \cup \{(n_1, n_2) : n_2 \geq s_2, 0 \leq n_1 \leq s_1\}.$$

An example realization is in Figure 2.7. As explained in Kopzon *et al.* (2008), minimal values of the thresholds need to satisfy:

$$\frac{\lambda_2}{\mu_2}(\frac{\lambda_1}{\mu_1})^{s_1} < 1, \qquad \frac{\lambda_1}{\mu_1}(\frac{\lambda_2}{\mu_2})^{s_2} < 1.$$

An intuitive explanation of this is in Kopzon *et al.* (2008). Under these conditions, the stationary distribution, $P(n_1, n_2)$ is:

$$P(n_1,n_2) = \begin{cases} P(s_1,s_2)\dfrac{\left(\frac{\lambda_2}{\mu_2}\right)^{n_2}+\frac{\lambda_1}{\lambda_2-\mu_2}\left(\left(\frac{\lambda_2}{\mu_2}\right)^{n_2}-1\right)}{\left(\frac{\lambda_2}{\mu_2}\right)^{s_2}+\frac{\lambda_1}{\lambda_2-\mu_2}\left(\left(\frac{\lambda_2}{\mu_2}\right)^{s_2}-1\right)}, & n_1 = s_1,\ 0 \leq n_2 \leq s_2 \\ P(s_1,s_2)\dfrac{\left[\frac{\lambda_1}{\mu_1}+\frac{\lambda_1}{\lambda_2-\mu_2}\left(\left(\frac{\lambda_2}{\mu_2}\right)^{s_2}-1\right)\right]^{n_1-s_1-1}}{\left[\left(\frac{\lambda_2}{\mu_2}\right)^{s_2}+\frac{\lambda_1}{\lambda_2-\mu_2}\left(\left(\frac{\lambda_2}{\mu_2}\right)^{s_2}-1\right)\right]^{n_1-s_1+1}}\times & \\ \quad\left\{\frac{\lambda_1}{\lambda_2-\mu_2}\left(\left(\frac{\lambda_2}{\mu_2}\right)^{s_2}-\frac{\lambda_1}{\mu_1}\right)+\frac{\lambda_1}{\mu_1}\frac{\lambda_1+\lambda_2-\mu_1-\mu_2}{\lambda_2-\mu_2}\left(\frac{\lambda_2}{\mu_2}\right)^{n_2}\right\}, & n_1 > s_1,\ 0 \leq n_2 \leq s_2, \end{cases}$$

with analogous expressions for $n_2 \geq s_2, 0 \leq n_1 \leq s_1$. The expression for $P(s_1, s_2)$ is:

$$P(s_1,s_2) = \left[\frac{\lambda_1+\mu_1}{2(\lambda_1-\mu_1)} + \frac{\frac{\lambda_2}{\mu_2}\frac{\lambda_1}{\lambda_1-\mu_1}-\frac{\mu_1}{\lambda_1-\mu_1}}{\left(\frac{\lambda_1}{\mu_1}\right)^{s_1}-\frac{\lambda_2}{\mu_2}} + \frac{\lambda_2+\mu_2}{2(\lambda_2-\mu_2)} + \frac{\frac{\lambda_1}{\mu_1}\frac{\lambda_2}{\lambda_2-\mu_2}-\frac{\mu_2}{\lambda_2-\mu_2}}{\left(\frac{\lambda_2}{\mu_2}\right)^{s_2}-\frac{\lambda_1}{\mu_1}}\right]^{-1}. \tag{2.2}$$

Queue balancing policy: The fixed threshold policy described above alternates between periods in which $s_1 \leq Q_1(t)$ and $s_2 \leq Q_2(t)$, always passing through the "shifted origin", (s_1, s_2), in the process. As a consequence, it appears that the difference between Q_1 and Q_2 is often large. This is hinted in Figure 2.7 but has not been explicitly analyzed. An alternative is the *queue balancing policy*. With this policy the control attempts to balance the size of the queues. A realization of it is in Figure 2.8.

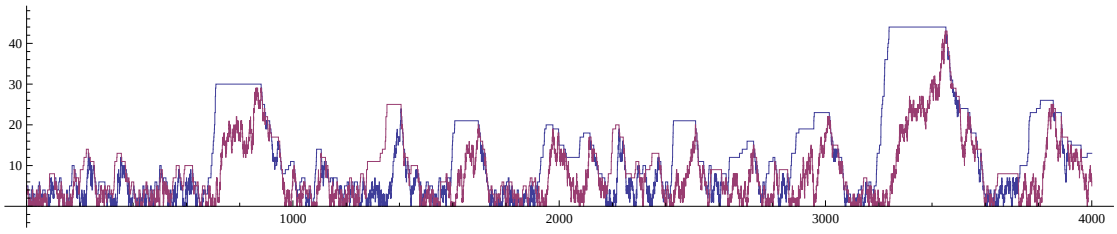

Figure 2.8: Queue sizes realization of the "Queue balancing policy". $\lambda = 1.25$, $\mu = 1$. $Q_1(t)$ – Red. $Q_2(t)$ – Blue.

The policy is defined as follows: If $|Q_2(t) - Q_1(t)| \geq 2$, the system (both servers 1 and 2) operate on the type with the shortest queue. Otherwise, both servers push. More specifically, if $Q_2(t) > Q_1(t) + 1$ then server 1 pushes and server 2 pulls. Similarly for type 2. The stationary

policy was only found for the symetric case in which $\lambda = \lambda_1 = \lambda_2$ and $\mu = \mu_1 = \mu_2$. The stationary distribution over $\mathbb{Z}_+^2$ is given below:

$$P(n_1, n_2) = \begin{cases} P(0,0) \prod_{i=0}^{n_1-1} \frac{\frac{\lambda}{\mu} + \frac{\lambda}{\lambda-\mu}\left(\left(\frac{\lambda}{\mu}\right)^i - 1\right)}{\left(\frac{\lambda}{\mu}\right)^i + \frac{\lambda}{\lambda-\mu}\left(\left(\frac{\lambda}{\mu}\right)^i - 1\right)}, & n_1 > 0,\ n_2 = 0 \\ P(n_1, 0) \frac{\frac{\lambda}{\lambda-\mu}\left(\frac{\lambda}{\mu}\right)^{n_1-1} + 2\left(\frac{\lambda}{\mu}\right)^{n_2+1} - \frac{\lambda}{\mu}\frac{\lambda}{\lambda-\mu}}{\frac{\lambda}{\mu} + \frac{\lambda}{\lambda-\mu}\left(\left(\frac{\lambda}{\mu}\right)^{n_1-1} - 1\right)}, & n_1 > n_2 > 0 \\ P(n_1, 0)\frac{\lambda}{\mu}, & n_1 = n_2 > 0, \end{cases}$$

where $P(0,0)$ normalizes the sum to 1.

Part II

Control

CHAPTER 3

FINITE HORIZON CONTROL

In this chapter we handle the problem of control of a multi-class queueing network over a finite time horizon. The control method employs the concepts of infinite virtual queues to achieve near optimal tracking of an optimal fluid solution. The contents of this chapter was published in Nazarathy and Weiss (2008b).

3.1 Introduction

Queueing networks are commonly used to model service, manufacturing and communication systems. In many situations one is interested in control of the network over a finite time horizon that attempts to minimize costs and maximize utility. In this respect, a sensible objective is to minimize the holding costs of the queues accumulated over the time horizon.

One theoretical approach to such finite horizon problems is to consider them as deterministic, discrete scheduling problems, cf. Lawler *et al.* (1993), Goemans and Williamson (1996) and Fleischer and Sethuraman (2003). In practice however these scheduling problems are too large to be tractable, and furthermore, an optimal schedule may not withstand the trial of application: As it is implemented over time, inaccuracies in the data and unexpected events (many small ones and a few large ones) accumulate and interfere with the solution, and there is no theory to say how close or far from optimum the result may be. Another theoretical approach is to model these problems by discrete stochastic systems and solve them as Markov decision problems, or approximate them on a diffusion scale by a continuous stochastic Brownian control problem, cf. Harrison (1988), Wein (1992), Kelly and Laws (1993), Harrison (1996), Harrison and Van Mieghem (1997), Maglaras (1999) and Kushner (2001). Markov decision problems or Brownian control problems usually focus on the optimization of the steady state of the system. They may therefore not be suitable for finite horizon problems, where typically the initial queue lengths and the total number of items processed are of the same order of magnitude, and one does not expect the system to reach steady state.

The problem which we address here has features of both approaches: In the finite time horizon we only schedule a finite batch of jobs, but we model these jobs in a multi-class queueing

network. As suggested in Weiss (1999), the method we use is to solve a deterministic fluid optimization problem which approximates the system, and then use decentralized local on-line controls to track the fluid solution. To carry this out we integrate three recent ideas which have been developed independently: (1) Solution of separated continuous linear programs (SCLP) by means of an exact simplex type algorithm, see Weiss (2008). (2) Modeling of queueing systems with unlimited supply of work by means of infinite virtual queues, as surveyed in our Chapter 2. (3) Maximum Pressure policies for stochastic processing networks as described in Dai and Lin (2005), see also Dai and Lin (2006), and Ata and Lin (2008).

As a first step, we discard the detailed information on jobs, and aggregate them into classes which are characterized by average processing times, and by their routes through the system. This yields a multi-class queueing network, which we wish to control optimally (Section 3.2). Next, our method approximates the multi-class queueing network by a deterministic continuous multi-class fluid network for which we formulate a finite horizon optimal control problem which is solved as a SCLP. The optimal fluid solution partitions the finite time horizon into time intervals distinguished by sets of empty and non-empty fluid buffers, and by constant fluid flow rates (Section 3.3). We now associate with each time interval a multi-class queueing network where each empty fluid buffer corresponds to a standard queue and each non-empty fluid buffer corresponds to an infinite virtual queue. The state of this associated system measures the deviation of the original system from the fluid solution (Section 3.4).

We then implement an on-line control of the queueing network by the use of a maximum pressure policy, where the pressure is calculated from the state of the associated network. This keeps the deviations from the fluid solution rate stable, and so the queueing network tracks the optimal fluid solution (Section 3.5). We call this procedure the Maximum Pressure Fluid Tracking Policy (MaxFTP).

The solution of the fluid control problem is centralized, and performed at the outset. The maximum pressure tracking is decentralized, and performed on-line using at each queue its own queue length and the queue lengths of queues directly downstream from it. This scheme is asymptotically optimal in the following sense: If we scale up the number of jobs in the system, and speed up the processing by the same amount, then the discrete stochastic system will converge to the optimal fluid solution, and no other policy can achieve asymptotically better results (Section 3.6).

Our purpose in this chapter is to introduce this method, and to sketch the proofs. In fact there is not much to prove, as most of the results we need were derived in Weiss (1999) and Dai and Lin (2005), and we need merely to adapt them to our framework, in Section 3.3, in Theorem 3.1 and Corollary 3.1. The main new result is the asymptotic optimality of MaxFTP which is proven in Theorem 3.2.

To illustrate MaxFTP, we describe its implementation for a simple re-entrant line with 2 servers and 3 classes. We also demonstrate the effectiveness of our results by means of simulations in which the asymptotic attributes of MaxFTP are empirically tested (Section 3.7).

Related fluid approaches for controlling multi-class queueing networks have been used in Avram *et al.* (1995); Maglaras (1999, 2000); Chen and Meyn (1999); Meyn (2001, 2003); Chen

et al. (2003) and others. MaxFTP is distinguished in that it is geared to control the transient finite horizon system, and for that purpose we use an optimal fluid solution.

In the context of wireless communication systems the proposed framework may be applied to mobile ad-hoc wireless networks (MANETs) that are both highly dynamic and heavily congested: When the network is highly dynamic, link conditions vary quickly and as a result link capacities, network topology and routes constantly change. When the network is heavily congested, the sojourn time of messages is high. Combining both highly variable dynamics and heavy congestion has the consequence that messages may experience changes in link capacities, network topology and routes while in transit. If a predictive, location based, routing scheme is employed, such as that presented by Shah and Nahrstedt in Shah and Nahrstedt (2002), predictions regarding the relative stability of link conditions may be made and the duration of the finite time horizons determined. In this case, our finite horizon approach may be used repeatedly for the short durations in which link conditions are relatively stable.

3.2 Finite Horizon Multi-Class Queueing Networks

We now define a MCQN similarly to the definition of Section 1.3. A MCQN consists of $k \in \mathcal{K} = \{1, \ldots, K\}$ job-classes and $i \in \mathcal{I} = \{1, \ldots, I\}$ servers. Jobs of class k queue up in buffer k, and we let the queue length $Q_k(t)$ be the number of jobs of class k in the system at time t. We let $Q_k(0),\ k \in \mathcal{K}$ be the initial queue lengths. Buffer k is served by server $\sigma(k)$, and the constituency of server i is $C_i = \{k \mid \sigma(k) = i\}$. In general a server may serve several classes, i.e. $|C_i| > 1$, hence the term multi-class. The topology of the network is described by the $I \times K$ constituency matrix $\mathbf{A}$ with elements $A_{ik} = 1$ if $k \in C_i$, $A_{ik} = 0$ otherwise.

We are only interested in the MCQN over a finite time horizon $[0, T]$. We may assume that all the jobs which will be processed during that time are already in the system at time 0. This assumption is without loss of generality: The general multi-class model has an arrival stream $E(t)$ which is often thought of as being supplied by buffer 0 that represents the outside world and contains an infinite supply of jobs. Since we are only interested in a finite time horizon, the supply of jobs can be taken as finite, so that the outside world can be replaced by an additional buffer in the network with a finite initial supply of all the jobs that will be served, and with a dedicated server that will release them into the rest of the system as the arrival process.

For $\ell = 1, 2, \ldots$, the ℓ's job out of buffer k requires processing amount $X_k(\ell)$, after which the job may either leave the system or move to another buffer. $S_k(t) = \max\{n \mid \sum_{\ell=1}^{n} X_k(\ell) \leq t\}$ counts the number of jobs completed at buffer k by processing for a total time t. $\phi_{kk'}(\ell)$ is the indicator of the event that the ℓ's job out of buffer k moved into buffer $k' \in \mathcal{K} \setminus k$. Let $\Phi_{kk'}(n) = \sum_{\ell=1}^{n} \phi_{kk'}(\ell)$, this is a count of the number of jobs routed from buffer k to k' out of the first n jobs served at buffer k.

The MCQN is controlled by allocating processing times to the buffers. Let $T_k(t)$ be the total time allocated to buffer k by server $\sigma(k)$, during $[0, t]$. Then the dynamics of the queues are described by:

$$Q_k(t) = Q_k(0) - S_k(T_k(t)) + \sum_{k' \in \mathcal{K} \setminus k} \Phi_{k'k}(S_{k'}(T_{k'}(t))). \tag{3.1}$$

The allocated times have to satisfy:

$$T_k(0) = 0, \quad T_k(t) \text{ non decreasing}, \quad \sum_{k \in C_i} T_k(t) - T_k(s) \leq t - s \text{ for all } s < t \text{ and each } i \in \mathcal{I}.$$

In particular each $T_k(t)$ is Lipschitz continuous with constant 1, and its derivative $\dot{T}_k(t)$ exists for almost all t, and satisfies:

$$\mathbf{A}\dot{T}(t) \leq \mathbf{1}, \quad \dot{T}(t) \geq 0, \quad 0 < t < T, \tag{3.2}$$

where $\mathbf{1}$ denotes a vector of 1's.

Additional constraints have to be satisfied by $T_k(t),\ Q_k(t),\ k \in \mathcal{K}$. First and foremost, $Q_k(t) \geq 0$ and no processing can occur when $Q_k(t) = 0$. We will also assume throughout this chapter that servers cannot be split so each server can work on only one job at a time. In addition we assume, that jobs are not preempted. Hence for almost all t, $\dot{T}_k = 0$ or $\dot{T}_k(t) = 1$, and $\dot{T}_k(t)$ can only change from 1 to 0 when $S_k(T_k(t))$ has a jump of 1, that is when the processing of a job is completed.

The cost associated with the MCQN over the finite time horizon is

$$V = \int_0^T \sum_{k=1}^K w_k Q_k(t) dt \tag{3.3}$$

which is the total inventory cost over the time horizon with holding costs rates w_k. If $w_k = 1$ for all k then V is the total work in process during the time horizon, also equal to the sum of sojourn times over $[0, T)$ of all jobs, where we assume that the sojourn time of a job that does not leave the system by time T is T. Minimization of V is also equivalent to maximization of the sum of the times from the completion of each job until T.

Minimization of V when $X_k(\ell),\ \phi_{kk'}(\ell)$ are known is an NP hard scheduling problem (job shop scheduling). The probabilistic version for infinite time horizon, with long term average cost minimization, is a Markov decision problem, which can sometimes be approximated by a Brownian control problem. Exact solution of the finite horizon problem under probabilistic assumptions is intractable. We will therefore attempt to solve the problem approximately.

To do so, we first of all discard all the detailed information of $X_k(\ell),\ \phi_{kk'}(\ell)$, and retain only averages. Remarkably, our asymptotic results show that for large systems this loss of information does not degrade the performance: The value achieved by our method converges to the value of the optimal solution with full information (Theorem 3.2).

We assume the following about the sequences of processing times and routing of the jobs:

$$\lim_{t \to \infty} \frac{S_k(t)}{t} = \mu_k \tag{3.4}$$

$$\lim_{n \to \infty} \frac{\Phi_{kk'}(n)}{n} = P_{kk'} \tag{3.5}$$

$$\lim_{n \to \infty} \frac{1}{n} \sum_{\ell=1}^n X_k(\ell)^{1+\epsilon} \leq C \tag{3.6}$$

where the last requirement has to hold for some $\epsilon > 0$ and some $C < \infty$. μ_k is the long term average processing rate, and $P_{kk'}$ the long term routing proportion ($\mathbf{P}$ is the $K \times K$ matrix of these values).

It is customary in papers on MCQN to cast (3.4–3.6) in a probabilistic framework. One assumes a stochastic process on probability space Ω and requires (3.4–3.6) to hold for almost every $\omega \in \Omega$. Examination of the proofs in Dai and Lin (2005) shows that for our purposes this is not necessary: our results hold for every sequence of $X_k(\ell)$, $\phi_{kk'}(\ell)$ which satisfies (3.4–3.6).

We define the input output matrix of the network by:

$$\mathbf{R} = (\mathbf{I} - \mathbf{P}')\text{diag}(\mu) \tag{3.7}$$

where $\mathbf{I}$ is the identity matrix and $\text{diag}(\mu)$ is a matrix with the rates μ_k in the diagonal. Here $R_{k'k}$ is the long term average rate at which buffer k' is depleted as a result of processing buffer k:

$$R_{k'k} = \begin{cases} \mu_k & k' = k \\ -\mu_k P_{kk'} & k' \neq k \end{cases}$$

Our approximate solution is in two stages: We first use $Q(0), w, \mathbf{R}, \mathbf{A}$ to formulate and solve a fluid optimization problem. This is done off line, centrally. We then use the fluid solution and the current queue lengths for decentralized tracking of the fluid solution.

The fluid approximation is suitable only when we process a large number of jobs over the time horizon. Our results are therefore asymptotic when the initial workload and processing speed are scaled up. We let the queueing network with $Q(0)$ and μ be our basic unit system and define a sequence of queueing networks. All the networks share the same sequence of processing times and routing indicators. For $N = 1, 2, \ldots$, the N scaled network is represented by $Q_k^N(t), T_k^N(t)$, in which we have $Q_k^N(0) = NQ_k(0)$, and the processing times are $X_k(\ell)/N$. Thus the initial work load is N times larger than the basic network, and the processing is speeded up N fold. In particular, $\mu_k^N = N\mu_k$, and $\mathbf{R}^N = N\mathbf{R}$.

Example Model

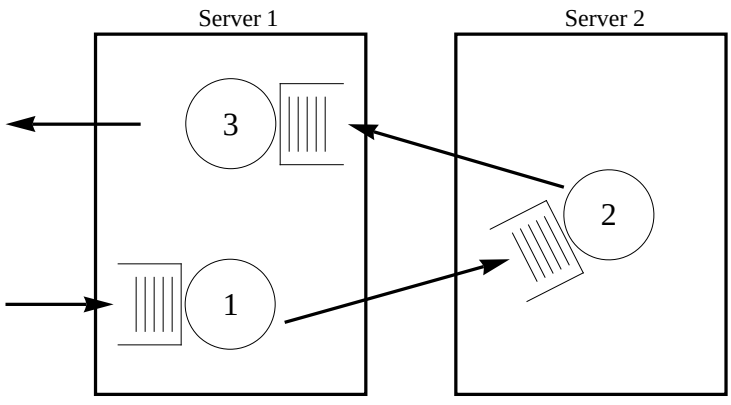

Figure 3.1: Example network.

The example model that we use throughout this chapter is the $K = 3$, $I = 2$ re-entrant line with $C_1 = \{1, 3\}$, $C_2 = \{2\}$. An illustration of the network is in Figure 3.1. Routing is

deterministic: jobs move from class 1 to class 2 and then to class 3, so that $\phi_{12}(\ell) = \phi_{23}(\ell) = 1, \ell = 1, 2, \ldots$, and all other routing indicators are 0. The processing time sequences are drawn from independent exponential random variables. The processing rates are $\mu_1 = 1$, $\mu_2 = \frac{1}{4}$ and $\mu_3 = 1$. The resulting input-output matrix is:

$$\mathbf{R} = \begin{pmatrix} 1 & 0 & 0 \\ -1 & \frac{1}{4} & 0 \\ 0 & -\frac{1}{4} & 1 \end{pmatrix}$$

We assume initial queue amounts: $Q_1(0) = 8$, $Q_2(0) = 1$, $Q_3(0) = 15$, a time horizon of $T = 40$ and holding costs $w_1 = w_2 = w_3 = 1$.

3.3 Optimization of the Multi-Class Fluid Network

Fluid approximations have been a major tool in the research on multi-class queueing networks, they have been used to verify the stability of networks, to evaluate performance in steady state, and to control multi-class queueing networks so as to improve their steady state performance. For some relevant recent work see Chen and Yao (1993), Connors *et al.* (1994), Avram *et al.* (1995), Dai (1995), Chen and Meyn (1999) ,Meyn (2001), Meyn (2003) and Chen *et al.* (2003).

Corresponding to the MCQN, we formulate a multi-class fluid network (MCFN) optimization problem: find bounded measurable functions $u_k(t)$ and absolutely continuous functions $q_k(t)$ such that

$$\begin{aligned} \min V_f &= \int_0^T w' q(t) dt && (3.8) \\ \text{s.t.} & \\ q(t) &= q(0) - \int_0^t \mathbf{R} u(s) ds && (3.9) \\ \mathbf{A}u(t) &\leq \mathbf{1} && (3.10) \\ q(t),\ u(t) &\geq 0 && (3.11) \\ & && t \in [0, T] \end{aligned}$$

Here the dynamics of $q(t)$ are given by

$$q_k(t) = q_k(0) - \mu_k T_k(t) + \sum_{k' \in \mathcal{K} \setminus k} P_{k'k} \mu_{k'} T_{k'}(t) \geq 0$$

which is the fluid analog of the MCQN dynamics (3.1). The processing of fluid out of buffer k is at a deterministic continuous rate μ_k, and the fluid out of k is routed in exact proportions $P_{kk'}$ to the other buffers $k' \neq k$. Thus instead of the discrete stochastic nature of the MCQN the MCFN is a continuous deterministic system. A fraction $u_k(t)$ of the server $\sigma(k)$ is allocated to the fluid buffer k at time t, and $T_k(t) = \int_0^t u_k(s) ds$. $u_k(t)$ satisfy the same constraints as $\dot{T}$ in (3.2), but servers are infinitely divisible, so that $u_k(t)$ can take any value in $[0, 1]$.

The problem (3.8–3.11) is a special case of a separated continuous linear program (SCLP), cf. Anderson (1981), Anderson and Nash (1987), Bellman (1953) and Pullan (1993). A simplex based algorithm that solves a wide class of SCLP problems optimally in a finite number of

steps has been recently found, see Weiss (2008). This has been an open problem for half a century. Current naive implementations[1] of the simplex based algorithm are able to quickly solve MCFN problems in which the dimensions of K and I are of the order of 100.

The analysis in Weiss (2008) reveals important features of the solution, essential to our control method. The MCFN problem is always feasible and bounded, and the SCLP simplex algorithm will always produce a fluid solution that consists of a partition of the time horizon $0 = t_0 < t_1 < \cdots < t_M = T$, where M is bounded, and in each of the time intervals the server allocations $u_k(t)$ are constant, and as a result the fluid buffer levels $q_k(t)$ are continuous piecewise linear.

We let $\tau_m = (t_{m-1}, t_m)$ denote the m'th time interval of the solution, and we denote by $u_k^m = u_k(t)$, $t \in \tau_m$ the values of the server allocations. During τ_m, server i will be busy for a fraction $\rho_i^m = \sum_{k \in C_i} u_k^m$. This is the utilization of server i during τ_m, and is always ≤ 1.

In each τ_m some of the buffers will be empty throughout the whole time interval, and the remaining buffers will be non-empty throughout the whole time interval. In general empty buffers are not inactive: they may have fluid flowing into them as well as out of them, with the inflow rate and outflow rates equal. We partition $\mathcal{K}$ during the m'th time interval as follows:

$$\mathcal{K}_0^m = \{k \mid q_k(t) = 0, t \in \tau_m\}$$
$$\mathcal{K}_\infty^m = \{k \mid q_k(t) > 0, t \in \tau_m\}$$

In our control approach, the fluid solution, summarized by $\tau_m, u^m, \mathcal{K}_0^m, \mathcal{K}_\infty^m$, is calculated off-line at the outset (time 0), from the data $T, w, \mathbf{R}, \mathbf{A}$, and the initial fluid levels $q_k(0) = Q_k(0)$. This solution is made available to all the servers.

Solution of the Example Fluid Network

The multi-class fluid network problem for our example is:

$$\begin{aligned}
\min V_f &= \int_0^T q_1(t) + q_2(t) + q_3(t) dt \\
\text{s.t.} & \\
q_1(t) &= q_1(0) - \int_0^t \mu_1 u_1(s) ds \\
q_2(t) &= q_2(0) - \int_0^t \mu_2 u_2(s) ds + \int_0^t \mu_1 u_1(s) ds \\
q_3(t) &= q_3(0) - \int_0^t \mu_3 u_3(s) ds + \int_0^t \mu_2 u_2(s) ds \\
& u_1(t) \qquad + u_3(t) \leq 1 \\
& \qquad u_2(t) \qquad \leq 1 \\
& u(t), q(t) \geq 0 \qquad\qquad t \in [0, T]
\end{aligned} \tag{3.12}$$

To understand the dynamic of the fluid solution, we study three different feasible solutions of (3.12). These are shown in Figures 3.2, 3.3, and 3.4 where we plot the values $\{q_1(t),\ \ q_1(t) + q_2(t),\ \ q_1(t) + q_2(t) + q_3(t)\}$ for $0 < t < T$. The three solutions are the last buffer first served

[1] Updated demonstrations and versions of this implementation are posted at `http://www.stat.haifa.ac.il/~gweiss`.

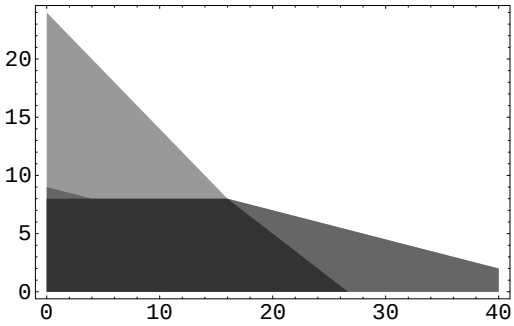

Figure 3.2: LBFS. $V_f = 376$.

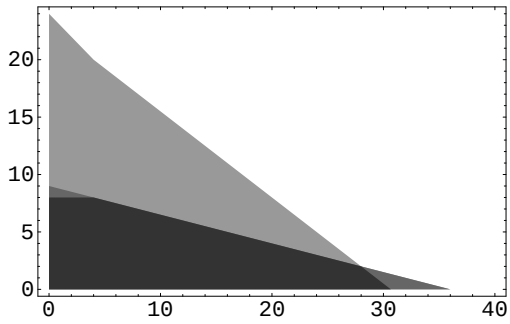

Figure 3.3: Minimal makespan. $V_f = 360$.

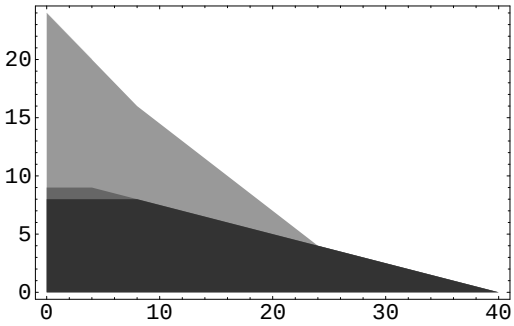

Figure 3.4: Optimal SCLP. $V_f = 352$.

(LBFS) solution, and minimum makespan solution and the optimal solution of (3.12) respectively.

The LBFS policy for a general re-entrant line with flow $1 \to 2 \to \cdots \to K$ gives priority to higher indexed buffers. In our example, server 1 gives priority to buffer 3 over buffer 1, and hence it allocates full capacity to buffer 3 until it is empty, and thereafter enough capacity is allocated to buffer 3 to keep it empty, and the remaining capacity is allocated to buffer 1. Server 2 is allocating full capacity to buffer 2. This is illustrated in Figure 3.2. The cost over time $[0, 40]$ is 376. Note that under LBFS server 2 which is the bottleneck server is kept idle from time 4 to time 16. If we continue using LBFS after time 40 the system will empty at 48, for a total cost of 384, (over the time horizon $[0, 48]$).

We can avoid idleness of the bottleneck server, by allocating $u_1(t) = 1/4$ once buffer 2 is empty. Doing so will keep server 2 busy and the system will empty in minimal time: $(q_1(0) +$

$q_2(0))\mu_2^{-1} = 36$. This is called a minimum makespan solution. It is illustrated in Figure 3.3: Server 1 is allocated to buffer 3 until $t = 4$, when buffer 2 is empty, at which point server 1 allocates $u_1 = 1/4$ to buffer 1, and the remaining capacity $u_3 = 3/4$ to buffer 3, which is emptying at rate $3/4$. The total cost is 360.

The optimal solution, which minimizes the cost in (3.12) over the time horizon of $T = 40$ is presented in Figure 3.4. It is a compromise between LBFS and minimal makespan. LBFS is employed until time $t = 8$, and after that, the bottleneck server is kept fully utilized. The optimal cost is 352. All details of this fluid solution are in Table 3.1.

Let t^* be the time at which we start to divert processing capacity from buffer 3 to buffer 1 in order that server 2 will be fully utilized. For minimum makespan $t^* = 4$, for the optimal solution it is $t^* = 8$ and for LBFS it is $t^* = 16$. The cost to empty the system is actually given by the quadratic function $W(t^*) = \frac{1}{2}(t^*)^2 - 8t^* + 384$ and it is minimized at $t^* = 8$. It thus happens that for our example this simple "local optimization" actually solves the optimization problem over all possible controls, as we find by solving the SCLP[2].

3.4 Modeling as MCQN+IVQ

The fluid solution indicates that in time interval τ_m, buffers $k \in \mathcal{K}_\infty^m$ are not empty throughout the entire time interval. Assume that we are able to track the fluid solution with the actual MCQN, so that $Q_k(t) > 0,\ k \in \mathcal{K}_\infty^m$ for all $t \in \tau_m$. In that case, the class k buffer always has work available, and so for the dynamics of the network, its level during τ_m is not relevant. This can be modeled by MCQN with infinite virtual queues (MCQN+IVQ), which we describe now. For background on infinite virtual queues see Chapter 2.

A multi-class queueing network with infinite virtual queues (MCQN+IVQ) is defined as follows: It consists of classes $\mathcal{K} = \mathcal{K}_0 \cup \mathcal{K}_\infty$, servers $\mathcal{I}$ and constituency matrix $\mathbf{A}$. Queues of classes $k \in \mathcal{K}_0$ are standard queues. Queues of classes $k \in \mathcal{K}_\infty$ are infinite virtual queues: They always have an unlimited supply of jobs available for processing. Let $S_k(t)$ be the counting process of job completions after processing duration t. Let $\Phi_{kk'}(n)$ count the number of jobs routed from queue k to queue k' out of the first n job completions of class k, where $k \in \mathcal{K}$, and $k' \in \mathcal{K}_0 \backslash k$. Since buffers with infinite virtual queues are never empty, we do not keep record of jobs which are routed into them, hence we do not consider $\Phi_{kk'}(n)$ when $k' \in \mathcal{K}_\infty$.

For $k \in \mathcal{K}_0$ we let $Q_k(t) \geq 0$ denote the number of jobs of class k in the system at time t. For $k \in \mathcal{K}_\infty$ we indicate the relative state of the queue by counting the departures, and by relating them to some nominal input rate α_k. We denote this relative queue length by $R_k(t)$. For time allocations $T_k(t),\ k \in \mathcal{K}$ we then have the dynamics of a MCQN with IVQs:

$$Z_k(t) = \begin{cases} Q_k(t) &= Q_k(0) - S_k(T_k(t)) + \sum_{k' \in \mathcal{K} \backslash k} \Phi_{k'k}(S_{k'}(T_{k'}(t))) & k \in \mathcal{K}_0 \\ R_k(t) &= R_k(0) - S_k(T_k(t)) + \alpha_k t & k \in \mathcal{K}_\infty \end{cases} \tag{3.13}$$

Here $Q_k(0)$ are initial queue lengths, and $R_k(0)$ are some arbitrarily chosen initial values.

To summarize: The MCQN with IVQs is controlled by time allocations $T_k(t)$. The standard queues $Q_k(t),\ k \in \mathcal{K}_0$ have input of jobs routed from other queues, and output $S_k(T_k(t))$. They

[2] A graphical demonstration of this fluid solution is at `http://www.stat.haifa.ac.il/~yonin/qsm/main.html`.

are non-negative integer valued. The infinite virtual queues supply a stream of jobs into the system with their output $S_k(T_k(t))$ and sustain continuous input themselves at nominal rates α_k. Thus their relative level $R_k(t)$ is not restricted in sign and is not restricted to be integer.

The infinite virtual queues replace the input stream of standard MCQN. If $R_k(t)$ is stable in the sense that it is kept close to zero, then the departure process of the infinite virtual queue provides input to the rest of the system at the rate α_k. There are two new elements here which provide wider modeling capacity: (i) the input streams are controlled from within the network, by the allocation of processing time $T_k(t)$, and (ii) the infinite virtual queues share servers with other classes. In particular, if a server i serves some standard as well as some infinite virtual queues, then it will always have work to do, and so it can work at full utilization of $\rho_i = 1$, and yet, it is possible that the standard queues $Q_k(t)$ will be stable, and behave like queues in light traffic.

Let $\alpha = (\alpha_1, \ldots, \alpha_K)'$ be the vector of nominal input rates for $k \in \mathcal{K}_\infty$ and $\alpha_k = 0$ for $k \in \mathcal{K}_0$. The overall traffic intensity ρ for this exogenous input vector α is determined by the following linear program, which is a modified version of the *static planning problem LP* introduced in Harrison (2000):

$$\begin{aligned} \min \quad & \rho \\ \text{s.t.} \quad & \mathbf{R}u = \alpha, \\ & \mathbf{A}u \le \mathbf{1}\rho, \\ & u \ge 0. \end{aligned} \tag{3.14}$$

Here $\mathbf{R}$ is the input-output matrix for the MCQN with IVQs and thus the first constraint in (3.14) reads:

$$\begin{array}{ll} \mu_k u_k - \sum_{k' \in \mathcal{K} \setminus k} P_{k'k} \mu_{k'} u_{k'} = 0, & k \in \mathcal{K}_0 \\ \mu_k u_k = \alpha_k, & k \in \mathcal{K}_\infty \end{array} \tag{3.15}$$

A MCQN with IVQs is called *rate stable* if $\lim_{t\to\infty} \frac{1}{t} Z(t) = 0$. It can be shown that a necessary condition for rate stability of MCQN with IVQs, under any policy, is that the overall traffic intensity is $\rho \le 1$ (see Dai and Lin (2005)).

MCQN with IVQs for each Time Interval

We return to our original MCQN over the finite time horizon, with its fluid solution. We associate a MCQN with IVQs to each of the M intervals of the fluid solution. During τ_m the associated MCQN with IVQs is defined by the partition $\mathcal{K}_0^m$ and $\mathcal{K}_\infty^m$. The nominal input rates of $k \in \mathcal{K}_\infty^m$ are set to $\alpha_k^m = \mu_k u_k^m$, which is the optimal outflow rate from the non-empty fluid buffer in the solution of (3.12). The corresponding matrix $\mathbf{R}^m$ is that of (3.7) changed to have $R_{k'k}^m = 0$ for $k' \in \mathcal{K}_\infty^m$, $k' \ne k$.

Table 3.1 describes the 4 MCQN with IVQs associated with the 4 intervals of the fluid solution of the example network. The flow rates of the fluid solution, u_k^m, $k \in \mathcal{K}$, provide a feasible solution of (3.14, 3.15), with $\rho^m = \max_{i\in\mathcal{I}} \rho_i^m \le 1$. Hence the necessary condition for rate stability is satisfied by each of the MCQN with IVQs. Clearly, in the fluid solution, $q_k(t) = 0$ for $k \in \mathcal{K}_0^m$, and so $Q_k(t)$ is the deviation from the fluid solution for buffers $k \in \mathcal{K}_0^m$.

Time interval m	$=$	1	2	3	4
τ_m	$=$	$(0,\ 4)$	$(4,\ 8)$	$(8,\ 24)$	$(24,\ 40)$
(u_1^m, u_2^m, u_3^m)	$=$	$(0,\ 0,\ 1)$	$(0,\ 1,\ 1)$	$(\frac{1}{4},\ 1,\ \frac{3}{4})$	$(\frac{1}{4},\ 1,\ \frac{1}{4})$
(ρ_1^m, ρ_2^m)	$=$	$(1,\ 0)$	$(1,\ 1)$	$(1,\ 1)$	$(\frac{1}{2},\ 1)$
$\mathcal{K}_0^m$	$=$	$\emptyset$	$\emptyset$	$\{2\}$	$\{2,\ 3\}$
$\mathcal{K}_\infty^m$	$=$	$\{1,\ 2,\ 3\}$	$\{1,\ 2,\ 3\}$	$\{1,\ 3\}$	$\{1\}$
$(\alpha_1^m,\ \alpha_2^m,\ \alpha_3^m)$	$=$	$(0,\ 0,\ 1)$	$(0,\ \frac{1}{4},\ 1)$	$(\frac{1}{4},\ 0,\ \frac{3}{4})$	$(\frac{1}{4},\ 0,\ 0)$
$\mathbf{R}^m$	$=$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ -1 & \frac{1}{4} & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ -1 & \frac{1}{4} & 0 \\ 0 & -\frac{1}{4} & 1 \end{pmatrix}$

Table 3.1: Fluid Solution of the example network and parameters of the associated MCQNs with IVQs

For the classes $k \in \mathcal{K}_\infty^m$ the fluid solution has outflow at rate $\mu_k u_k^m$, and so $R_k(t)$ measures the deviation from the fluid outflow rate for $k \in \mathcal{K}_\infty^m$. If we keep $Z^m(t)$ rate stable we will keep these deviations small. Furthermore, for $k \in \mathcal{K}_\infty^m$, rate stability of $Z^m(t)$ will yield that $\Phi_{k'k}(S_{k'}(T_{k'}(t))) - P_{k'k}\mu_{k'}u_{k'}^m$ will also be stable, but this is the deviation between the actual fluid inflow into buffer k, and the fluid inflow in the fluid solution. It follows that keeping $Z^m(t)$ rate stable implies good tracking of the fluid solution during τ_m (this is formally stated in Theorem 3.2).

3.5 Application of Maximum Pressure Policies

Brief background regarding maximum pressure policies is in Chapter 1. These scheduling policies were studied in the general framework of stochastic processing networks (introduced by Harrison (2000, 2002, 2003)) in Dai and Lin (2005) (see also Dai and Lin (2006), and Ata and Lin (2008)). Maximum pressure policies are distinguished by two properties: Under sufficient conditions they are rate stable for systems with overall traffic intensity $\rho \leq 1$, and in heavy traffic ($\rho \approx 1$), networks with complete resource pooling have optimal diffusion scale approximations. We now briefly describe maximum pressure policies and their adaptation to MCQN with IVQs. The results presented in this section are an adaptation from Dai and Lin (2005).

Denote by $a_k = \dot{T}_k(t)$ the level at which class k jobs are served. When $a_k = 1$ server $\sigma(k)$ serves class k fully. When $a_k = 0$, there is no processing of jobs from class k. Momentarily assume that we allow processor sharing, thus $0 \leq a_k \leq 1$. The column vector $a = (a_1, \ldots, a_K)'$ is an allocation. Let the set of feasible allocations $\mathcal{A}$ be defined as the non-negative vectors a such that $\sum_{k \in C_i} a_k \leq 1$ for all $i \in \mathcal{I}$ (these are the conditions 3.2). $\mathcal{A}$ is a non-empty (contains 0), bounded convex polytope, and has a finite number of extreme points. Denote the set of extreme points of $\mathcal{A}$ by $\mathcal{E}$. While $\mathcal{A}$ summarizes the set of allocations that satisfy the resource consumption constraints, it may be that due to emptiness of buffers $k \in \mathcal{K}_0$ some allocations in A are not available at certain times. Denote by $\mathcal{A}(t) \subseteq \mathcal{A}$ the set of available allocations at time t. Let $\mathcal{E}(t) = \mathcal{E} \cap \mathcal{A}(t)$ denote the set of the extreme allocations which are available at time t.

Denote the column vector that is the state of a MCQN at time t by $z = (Z_1(t), \ldots, Z_K(t))'$.

Define the total network pressure for an allocation a to be $z'\mathbf{R}a$ (where z' is a row vector, $\mathbf{R}$ is the input-output matrix and a is a column allocation vector). A service policy is said to be a maximum pressure policy if at each time t, the network chooses an allocation $a^* \in \text{argmax}_{a\in\mathcal{E}(t)} z'\mathbf{R}a$.

By following all the steps in the proofs of the results of Dai and Lin (2005) one can see that they hold without any change also for MCQN with IVQs, as defined in Section 3.4. We therefore adopt Dai and Lin's main throughput optimality result to our context:

Theorem 3.1. *Let $Z(t)$ be a MCQN with IVQs. Assume that the processing and routing counts satisfy conditions (3.4–3.6). Let ρ be the overall traffic intensity of the network for nominal input α, and assume that $\rho \leq 1$. Then under maximum pressure policy with no splitting of servers and with no preemptions $\lim_{t\to\infty} \frac{1}{t} Z(t) = 0$.*

We make just one comment about the proof: One needs to show that MCQN with IVQs satisfy EAA assumption, see Dai and Lin (2005). The steps of the proof are as in Dai and Lin (2005), but we need to make use of the fact that $\phi_{kk'}(\ell) = 0$ for $k' \in \mathcal{K}_\infty$.

In fact we need the asymptotic result for slightly different scaling. Let $Z(t)$ be a MCQN with IVQs. Let $S_k(t)$, $\Phi_{kk'}(n)$ be the processing and routing counts of $Z(t)$, and let α be its nominal inputs vector. $Z^N(t)$ is called an N scaling of $Z(t)$ with initial conditions $Z^N(0)$, if it has processing counts $S_k^N(t) = S_k(Nt)$, routing counts $\Phi_{kk'}(n)$, and nominal input $N\alpha$. We then have:

Corollary 3.1. *Let $Z(t)$ be a MCQN with IVQs. Assume that the processing and routing counts satisfy conditions (3.4–3.6). Let ρ be the overall traffic intensity of the network for nominal input α, and assume that $\rho \leq 1$. Let $Z^N(t)$ be a sequence of N scalings of $Z(t)$, with initial states $Z^N(0)$ that satisfy $Z^N(0)/N \to 0$ as $N \to \infty$. Then under a maximum pressure policy with no splitting of servers and with no preemptions, $Z^N(t)/N \to 0$ uniformly for $0 < t < T$ for any $T > 0$.*

As observed in Dai and Lin (2005) and Tassiulas (1995), the maximization of the pressure $z'\mathbf{R}a$ over $Aa \leq \mathbf{1}$, $a \geq 0$ separates into maximization of the pressure for each of the servers. This in turn is achieved by

$$k^* \in \arg\max_{k\in C_i\cup 0} \{\mu_k(Z_k(t) - \sum_{k'\in\mathcal{K}_0\setminus k} P_{kk'} Z_{k'}(t)), 0\}$$

where $k^* = 0$ corresponds to idling because all available queues have non-positive pressure.

3.6 Maximum Pressure Tracking of the Optimal Fluid Solution

We come now to integration of the SCLP fluid solution, the associated MCQN with IVQs, and the maximum pressure policy, as described in Sections 3.4–3.6 into a policy for the solution of the finite horizon MCQN control problem of Section 3.2.

Maximum Pressure Fluid Tracking Policy (MaxFTP):

Phase 1 (at time 0, centralized): Use $Q(0),\ w, \mathbf{R}, \mathbf{A}$ to solve the fluid network problem (3.8–3.11), and obtain the time intervals τ_m, the sets of empty and non-empty fluid buffers $\mathcal{K}_0^m$, $\mathcal{K}_\infty^m$ and the flow rates $\alpha_k^m = u_k^m \mu_k$ for $k \in \mathcal{K}_\infty^m$.

Phase 2 (on-line, decentralized): Track the fluid solution, for $t \in [0, T)$ by applying a maximum pressure policy in each of the intervals $\tau_m,\ m = 1, \ldots, M$ as follows:

- Let $Q(t)$ be the queue lengths process for $t \in \tau_m$.
- Let $Z^m(t)$ for $t \in \tau_m$ be the state process of an associated MCQN with IVQs $\mathcal{K}_\infty^m$, and nominal inputs α^m, such that the processing times and routings of $Z^m(t)$ are identical to those of Q for $t \in \tau_m$. (Note that Z^m is as defined in (3.13) but with the time shifted by t_{m-1}).
- Set initial values for Z^m:

$$Z_k^m(t_{m-1}) = \begin{cases} Q_k(t_{m-1}) & k \in \mathcal{K}_0^m \\ -h_k^m \alpha_k^m \sqrt{|Q(0)|} & k \in \mathcal{K}_\infty^m \end{cases} \tag{3.16}$$

  where $|Q(0)| = \sum_{k \in \mathcal{K}} Q_k(0)$, and $h_k^m = 0$ if $k \in \mathcal{K}_0^m$ or if $k \in \mathcal{K}_\infty^m$ and $q_k(t_{m-1}) > 0$, and $h_k^m = \min_{\{k' : P_{k'k} > 0\}} h_{k'}^m + 1$ for the remaining k.
- At every time t let $\mathcal{E}(t)$ be the set of extreme allocations available for $Q(t)$.
  Let $Z^m(t)' \mathbf{R^m} a$ be the pressure of allocation a, calculated for $Z^m(t)$.
  Use the maximum pressure allocation, $\max_{a \in \mathcal{E}(t)} Z^m(t)' \mathbf{R^m} a$, without processor splitting or preemptions.

For the example network (and in fact for any re-entrant line), implementation of the maximum pressure fluid tracking policy is described in Table 3.2. When server i is available at time $t \in \tau_m$, it calculates the pressure of all buffers $k \in C_i$ which have $Q_k(t) > 0$ according to Table 3.2 and starts to process a job from the queue with the highest pressure if the resulting pressure is positive. Otherwise, it idles.

	$k \in \mathcal{K}_0^m$	$k \in \mathcal{K}_\infty^m$
$k+1 \in \mathcal{K}_0^m$	$\mu_k(Q_k(t) - Q_{k+1}(t))$	$Z_k^m(t_{m-1}) + \mu_k(\alpha_k(t - t_{m-1}) - (S_k(T_k(t)) - S_k(T_k(t_{m-1}))) - Q_{k+1}(t))$
$k+1 \in \mathcal{K}_\infty^m$	$\mu_k Q_k(t)$	$Z_k^m(t_{m-1}) + \mu_k(\alpha_k(t - t_{m-1}) - (S_k(T_k(t)) - S_k(T_k(t_{m-1}))))$

Table 3.2: Calculation of pressure in a re-entrant line, for interval τ_m. Pressure at buffer k depends on type of queue and queue length of classes $k,\ k+1$. By convention $K+1 \in \mathcal{K}_\infty^m$.

Our main result in this Chapter is:

Theorem 3.2. *Let $Q(t)$ be the queue length process of a finite horizon MCQN. Assume that the processing and routing counts satisfy conditions (3.4–3.6). Let $Q^N(t)$ be N scalings of $Q(t)$, with $Q^N(0) = NQ(0)$. Let $q(t)$ be the optimal fluid solution, and let V_f be its objective value.*

(i) Let V^N denote the objective value of $Q^N(t)$ for any general policy. Then

$$\liminf_{N\to\infty} \frac{1}{N} V^N \geq V_f$$

(ii) Under MaxFTP policy:

$$\lim_{N\to\infty} \frac{1}{N} Q^N(t) = q(t) \text{ uniformly on } 0 \leq t \leq T$$

and $\lim_{N\to\infty} \frac{1}{N} V^N = V_f$.

Proof. (i) Consider some general policy and let $\bar{V} = \liminf_{N\to\infty} \frac{1}{N} V^N$. Let r be a subsequence for which $\bar{V} = \lim_{r\to\infty} \frac{1}{r} V^r$. By the argument of Dai and Lin (2005), Section A.2 we can find a subsequence r' of the r such that $\lim_{r'\to\infty} (\frac{1}{r'} Q^{r'}(t), T^{r'}(t), \frac{1}{r'} V^{r'}) = (\bar{Q}(t), \bar{T}(t), \bar{V})$ uniformly on $[0, T]$, where $\bar{Q}(t), \bar{T}(t)$ are Lipschitz continuous fluid limits, and in particular $\bar{T}$ has derivative $\dot{\bar{T}}$ almost everywhere. One can see that $\bar{Q}(t), \dot{\bar{T}}(t), \bar{V}$ must be a feasible solution to (3.8–3.11). Hence, $\bar{V} \geq V_f$.

(ii) We now consider the sequence $(Q^N(t), Z^N(t), T^N(t), V^N)$ under the MaxFTP policy, where Z^N are the processes of deviations from the fluid solution. As above we have that for a subsequence r $\lim_{r\to\infty} (\frac{1}{r} Q^r(t), \frac{1}{r} Z^r(t), T^r(t), \frac{1}{r} V^r) = (\bar{Q}(t), \bar{Z}(t), \bar{T}(t), \bar{V})$, uniformly on $[0, T]$. Our goal is to show that $\bar{Q}(t), \dot{\bar{T}}(t)$ equal the optimal solution of (3.8–3.11). We do this by induction on the intervals τ_m, $m = 1, \ldots, M$. There is nothing to show for $t = 0$. Assume then that $\bar{Q}(t_{m-1}) = q(t_{m-1})$. Assume first that $q_k(t_{m-1}) > 0$ for all $k \in \mathcal{K}_\infty^m$. In that case we have that the initial values $Z_k^{m,N}(t_{m-1})$ as defined by (3.16) are equal to 0. We define:

$$\bar{\bar{t}} = \min[t_m, \inf\{t : t_{m-1} < t < t_m,\ \bar{Q}_k(t) = 0 \text{ for some } k \in \mathcal{K}_\infty^m\}]$$

By continuity of $\bar{Q}$, $\bar{\bar{t}} > t_{m-1}$. Let $t_{m-1} < \bar{t} < \bar{\bar{t}}$. Then for $r > r_0$ we will have $Q_k^r(t) > 0$ for $t_{m-1} < t < \bar{t}$, $k \in \mathcal{K}_\infty^m$. Hence for $r > r_0$ the MaxFTP policy will act on $Z^{m,r}(t)$ identical to max pressure policy. Consider then the sequence of scalings $Z^{m,r}$ under maximum pressure policy. The optimal solution of SCLP in the mth interval satisfies:

$$R^m u^m = \begin{bmatrix} R_{\mathcal{K}_0^m, \mathcal{K}_0^m} & R_{\mathcal{K}_0^m, \mathcal{K}_\infty^m} \\ 0 & \operatorname{diag}(\mu_{\mathcal{K}_\infty^m}) \end{bmatrix} u^m = \alpha^m$$
$$A u^m \leq 1, \qquad u^m \geq 0,$$

hence, comparing with (3.14) we see that $\rho \leq 1$ for the network $Z^{m,1}$. Hence, by Corollary 3.1, $\lim_{r\to\infty} Z^{m,r}(t) = 0$ on $0 < t < \bar{t}$, and because $\bar{t} < \bar{\bar{t}}$ was arbitrary the same holds for $0 < t < \bar{\bar{t}}$. We then have for all $0 < t < \bar{\bar{t}}$:

$$\bar{Q}_k(t) = \bar{Z}_k(t) = 0, \quad k \in \mathcal{K}_0^m,$$
$$\bar{Z}_k(t) = \alpha_k(t - t_{m-1}) - \mu_k(\bar{T}_k(t) - \bar{T}_k(t_{m-1})) = 0 \quad \text{implies } \dot{\bar{T}}_k(t) = \alpha_k/\mu_k, \quad k \in \mathcal{K}_\infty^m$$

so $\bar{Q}_k(t) = q_k(t)$, $k \in \mathcal{K}_0^m$, and $\dot{\bar{T}}_k(t) = u_k^m$, $k \in \mathcal{K}_\infty^m$. We next obtain for $\mathcal{K}_0^m$:

$$\bar{Q}_{\mathcal{K}_0^m}(t) = 0 - R_{\mathcal{K}_0^m, \mathcal{K}_0^m}[\bar{T}_{\mathcal{K}_0}(t) - \bar{T}_{\mathcal{K}_0}(t_{m-1})] - R_{\mathcal{K}_0^m, \mathcal{K}_\infty^m}[\bar{T}_{\mathcal{K}_\infty}(t) - \bar{T}_{\mathcal{K}_\infty}(t_{m-1})] = 0$$
$$\text{implies } \dot{\bar{T}}_{\mathcal{K}_0}(t) = -(R_{\mathcal{K}_0^m, \mathcal{K}_0^m})^{-1} R_{\mathcal{K}_0^m, \mathcal{K}_\infty^m} \operatorname{diag}(\alpha_k/\mu_k)_{k \in \mathcal{K}_\infty^m}$$

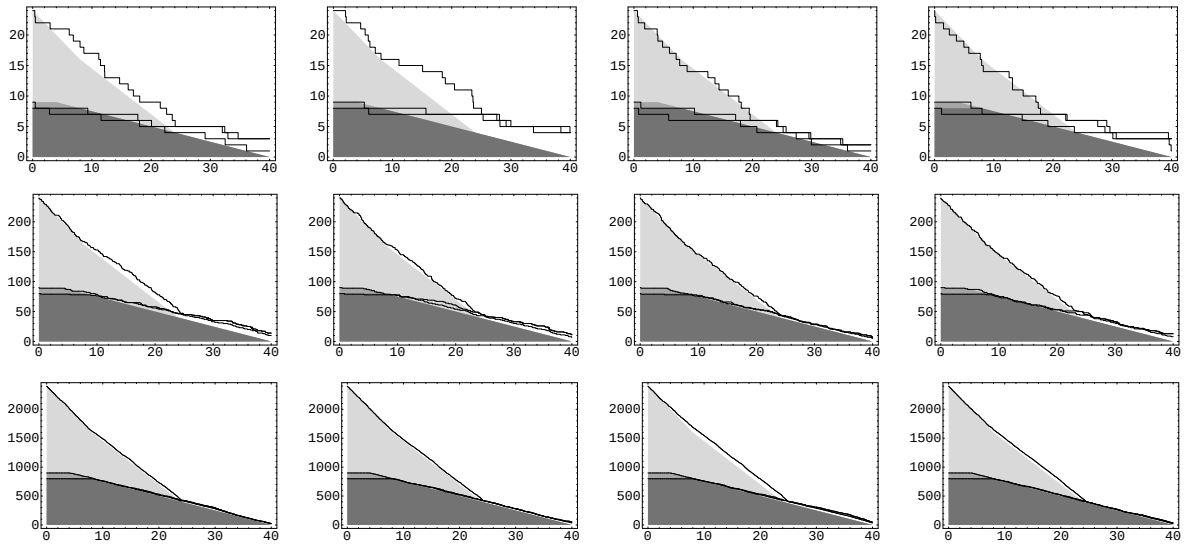

Figure 3.5: Example Realizations: 4 replicates (columns) with scalings $N = \{1, 10, 100\}$ for each replicate.

and so $\dot{\bar{T}}_k(t) = u_k^m$, $k \in \mathcal{K}_0^m$. Finally we substitute these values into

$$\bar{Q}_{\mathcal{K}_\infty^m}(t) = \bar{Q}_{\mathcal{K}_\infty^m}(t_{m-1}) - R_{\mathcal{K}_\infty^m,\mathcal{K}_0^m}[\bar{T}_{\mathcal{K}_0}(t) - \bar{T}_{\mathcal{K}_0}(t_{m-1})] - R_{\mathcal{K}_\infty^m,\mathcal{K}_\infty^m}[\bar{T}_{\mathcal{K}_\infty}(t) - \bar{T}_{\mathcal{K}_\infty}(t_{m-1})]$$

to get $\bar{Q}_k(t) = q_k(t)$, $k \in \mathcal{K}_\infty^m$. Since $\bar{Q}_k(\bar{\bar{t}}) = q_k(\bar{\bar{t}})$ we must have $\bar{\bar{t}} = t_m$.

Consider now the case that for some of the $k \in K_\infty^m$, $q_k(t_{m-1}) = 0$. We sketch the proof in this case. The MaxFTP policy will start off at time t_{m-1} with negative pressure in these buffers. As a result there will be no processing out of these buffers for a duration of $h_k^m\sqrt{N|Q(0)|}$. All the buffers with $h_k^m = 0$ will be processed according to maximum pressure. It can then be seen that the buffers with $h_k^m > 0$ will fill up with a quantity of jobs of the order of magnitude $\sqrt{N}$ by the time they reach positive pressure. As a result we get that $\frac{d}{dt}\bar{Q}(t_{m-1}) > 0$, and the proof proceeds as before.

We have shown that each fluid limit must equal the optimal fluid solution, and we also know that every sequence of scalings has a subsequence which has a fluid limit. This proves that $\lim_{N\to\infty} \frac{1}{N}Q^N(t) = q(t)$, uniformly on $[0, T]$. In particular this implies that $\lim_{N\to\infty} \frac{1}{N}V^N = V_f$. □

3.7 Simulation Results

We simulated the example network for N scalings of up to $N = 10^6$. We generated the processing times for each of the classes as pseudo random i.i.d. exponential random variables. For each single replicate we used $3 \times 4 = 12$ generator seeds which gave us long sequences of processing times for each of the three buffers in each of the four time intervals of the fluid solution. We then created N scalings of each replicate as defined in Section 3.5.

Figure 3.5 illustrates the simulation results for 4 such replicates (the four columns) and N scalings of $N = \{1,\ 10,\ 100\}$. In the Figure we show for each of the 12 simulated queueing

processes the values of $Q^N(t)$ plotted against the optimal fluid solution for that N (this is the fluid solution $q(t)$ multiplied by N for each t). The illustrations show that even for $N = 10$, the approximation is quite good.

Figure 3.6 examines the asymptotics of our policy in more detail and on a mass scale. Here we have plotted unscaled deviations, namely $Q_k^N(t) - Nq_k(t)$, at the time points $t = 4,\ 8,\ 24,\ 40$, which are the breakpoint times of the optimal fluid solution. This is performed for each queue $k = 1, 2, 3$. For 100 replicates we have performed the following N scalings: $N = \{1,\ 5 \cdot 10^3,\ 10^4,\ 5 \cdot 10^4,\ 10^5,\ 2 \cdot 10^5,\ 4 \cdot 10^5,\ 6 \cdot 10^5,\ 8 \cdot 10^5,\ 10^6\}$. The N scalings of each replicate are plotted as a continuous line. This allows us to appreciate how the deviations evolve along a single sample path, as the scaling increases.

As may be expected, the deviations all appear to be of the order of magnitude of $\sqrt{N}$. An exception is for the N scalings for $k = 3$ and $t = 40$, where the deviations look like queue lengths of a queue in light traffic (indeed the utilization of server 1 in the last interval is $1/2$).

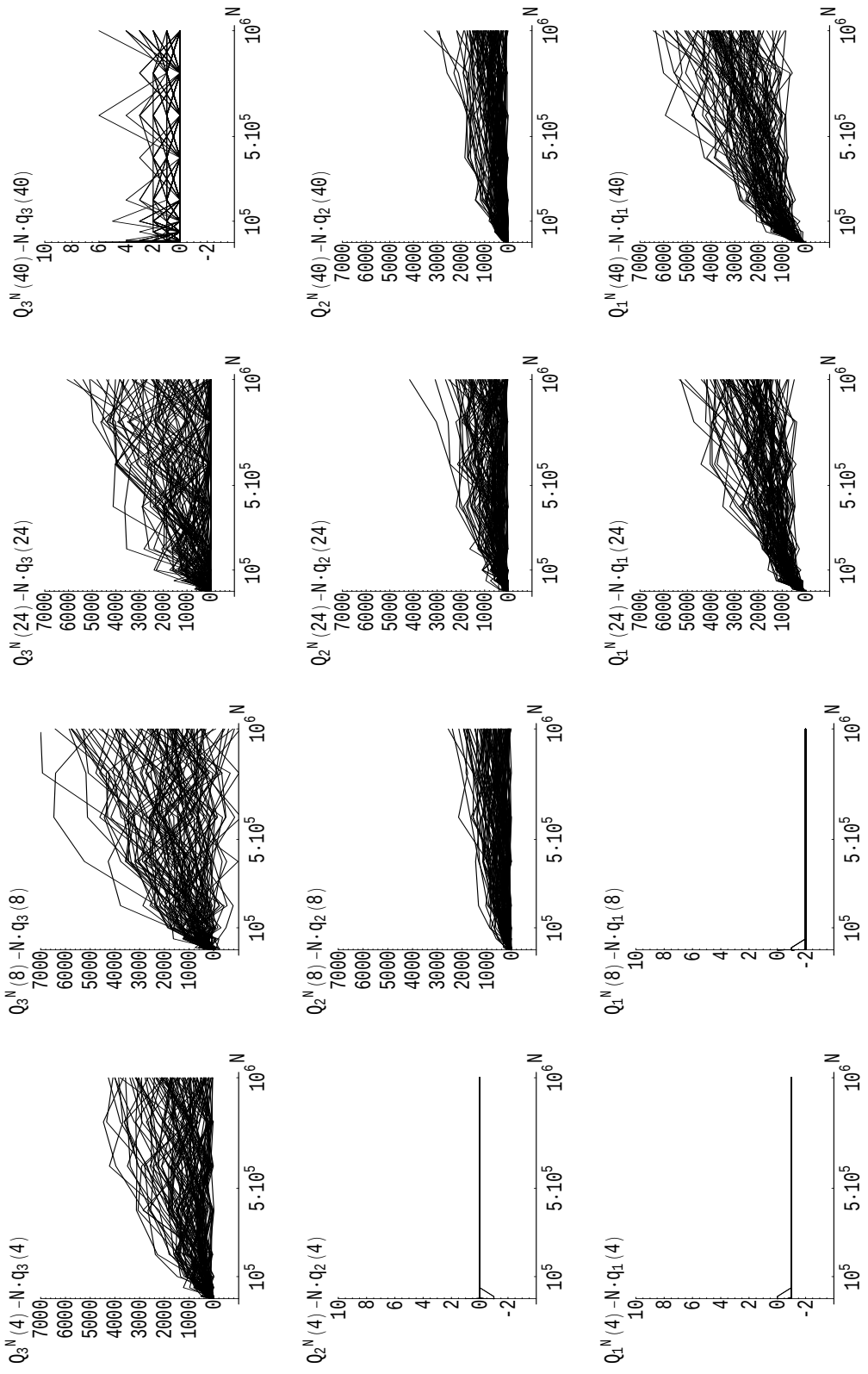

Figure 3.6: Empirical asymptotics: 100 replicates with N scalings up to $N = 10^6$.

CHAPTER 4

FULL UTILIZATION CONTROL

In this chapter we present an example of a queueing network with general processing times that may operate under full utilization while maintaining stability. The network we consider is a the push-pull network that has a similar structure to the KSRS network described in Section 1.7. This push-pull network was first analyzed in Kopzon and Weiss (2002) and further analyzed in Kopzon *et al.* (2008). We summarized the main results from those papers in Chapter 2. In both cases, the analysis was for the exponential processing time case. We now remove this memory less assumptions and thus resort to asymptotic methods of analysis. The results presented in this chapter were published in Nazarathy and Weiss (2008c).

The structure of this Chapter is as follows: In Section 4.1 we define the push-pull network and the policies that we analyze. In Section 4.2 we formulate the network as a multi-class queueing network with infinite virtual queues (MCQN+IVQ). Here we make the needed assumptions regarding the processing times. In Section 4.3 we analyze the fluid limit model of this network under fluid scaling, and show that the fluid model is stable under the corresponding policies. In Section 4.4 we assume i.i.d. processing times and formulate the network as a Markov process. We then follow the proof method of Dai (1995) to show that this Markov process is positive Harris recurrent, and so the two queues of the network posses a stationary limiting distribution. In Section 4.5 we present a minorization proof which is needed to show positive Harris recurrence.

4.1 The Push-Pull Network and Policies

We defined the push-pull network in Chapter 2. We now repeat the definition, this time with slightly different notation (the job classes are now labeled $1, 2, 3, 4$).

The *push-pull network* is pictured in Figure 4.1, it consists of two servers, numbered 1, 2 and two types of jobs numbered 1, 2 each of which is processed by both servers. Type 1 is processed by server 1 and then by server 2 (activities 1 and 2), while type 2 is first processed by server 2 and then by server 1 (activities 3 and 4). We call the first step of each type a *push activity* and the second step a *pull activity*. We denote by $Q_i(t), i = 2, 4$ the number of jobs in the two queues

at time t (including the job in process), and by $D_i(t), i = 1, 2, 3, 4$ the number of jobs that have completed activity i in the time interval $[0, t]$. When $Q_4(t) > 0$, server 1 can either pull, by serving a type 2 job from $Q_4(t)$ or push, by serving a type 1 job from the infinite supply. When $Q_4(t) = 0$ server 1 can still always push jobs of type 1. Hence, server 1 never needs to idle. Similarly for server 2.

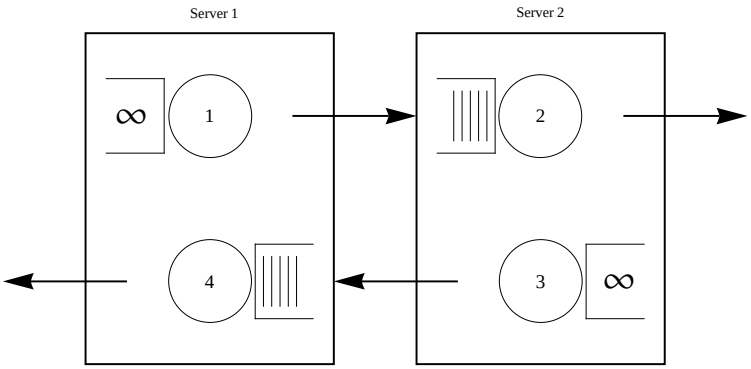

Figure 4.1: The push-pull queueing network with jobs classes labeled $1, 2, 3, 4$.

Assume that the long term average processing time for activity i is $1/\mu_i$, $i = 1, 2, 3, 4$. Let θ_i, $i = 1, 2, 3, 4$ be the long term fraction of time spent in activity i. If the system never idles then,

$$\theta_1 = 1 - \theta_4, \qquad \theta_3 = 1 - \theta_2.$$

Furthermore, if $Q_i(t)$ are stable then their input and output rates are equal, so:

$$\nu_1 = \nu_2 = \theta_1\mu_1 = \theta_2\mu_2, \qquad \nu_3 = \nu_4 = \theta_3\mu_3 = \theta_4\mu_4$$

where ν_i is the long term average rate of the departure process D_i, $i = 1, 2, 3, 4$, and in particular ν_2 (ν_4) is the rate at which jobs of type 1 (type 2) leave the network. Solving these equations we get:

$$\nu_1 = \nu_2 = \frac{\mu_1\mu_2(\mu_3 - \mu_4)}{\mu_1\mu_3 - \mu_2\mu_4}, \quad \nu_3 = \nu_4 = \frac{\mu_3\mu_4(\mu_1 - \mu_2)}{\mu_1\mu_3 - \mu_2\mu_4}.$$

We now specify the policies which we use. We consider preemptive resume head of the line policies for the inherently stable case and inherently unstable case, we do not consider the other two cases which were described in Chapter 2: the unbalanced case and the completely balanced case.

Inherently stable network: When $\mu_1 < \mu_2$ and $\mu_3 < \mu_4$, operation of the network on just one type causes the network to behave like a *stable* single server queue. In this case the policy which we use is preemptive resume head of the line priority for pull activities 4 and 2 over push activities 1 and 3. We refer to this as *Case 1*, and to the policy as *pull priority policy*.

Inherently unstable network: When $\mu_1 > \mu_2$ and $\mu_3 > \mu_4$, operation of the network on just one type causes the network to behave like an *unstable* single server queue. In this case priority to pull over push is unstable. A policy that works here is that while $Q_2(t)$

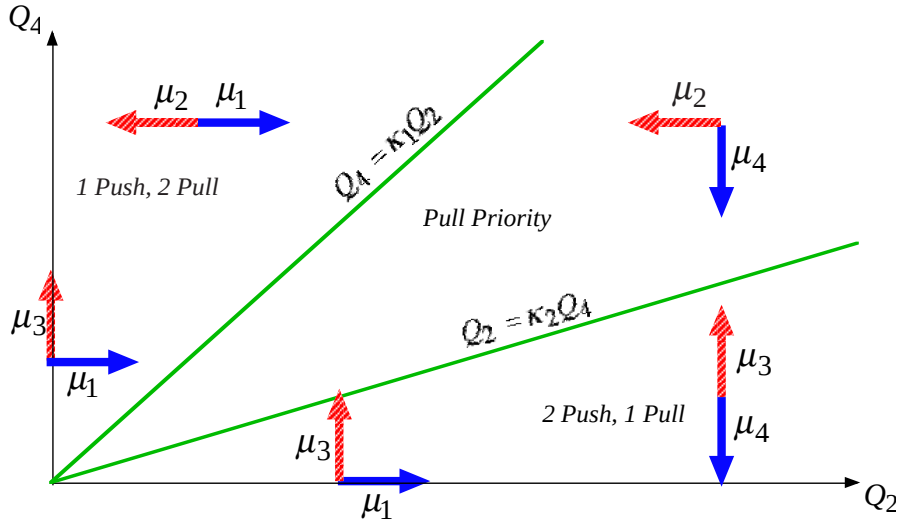

Figure 4.2: The linear threshold policy for the inherently unstable network (Case 2).

is below some threshold level server 1 will push work to server 2, and server 1 will only pull from $Q_4(t)$ when $Q_2(t)$ is above the threshold, with a similar rule for server 2. We use a linear threshold to determine pull or push preemptive head of the line priority. We define a family of such policies, each determined by a pair of constants κ_1, κ_2 which satisfy $\kappa_1 > \frac{\mu_3}{\mu_1}$, $\kappa_2 > \frac{\mu_1}{\mu_3}$:

Server 1: Priority to pull activity 4 over push activity 1 if $0 < Q_4(t) < \kappa_1 Q_2(t)$.

Server 2: Priority to pull activity 2 over push activity 3 if $0 < Q_2(t) < \kappa_2 Q_4(t)$.

We refer to this as *Case 2*, and to the policy as *linear threshold policy*, see Figure 4.2.

Preliminary Comparison to KSRS

We now wish to survey known results about the well studied Kumar-Seidman Rybko-Stolyar (KSRS) network, and contrast them with the very different behavior of our push-pull network.

The Kumar-Seidman Rybko-Stolyar multi-class queueing network (see Chapter 1) differs from our push-pull network in that instead of infinite supply of jobs there are two stochastic arrival streams of jobs of type 1 and of type 2, with long term average arrival rates α_1, α_3.

In that case there are 4 queues $Q_i(t)$ of jobs waiting for activities $i = 1, 2, 3, 4$ in the network, and the offered loads for servers 1 and 2 are $\rho_1 = \alpha_1/\mu_1 + \alpha_3/\mu_4$ and $\rho_2 = \alpha_3/\mu_3 + \alpha_1/\mu_2$ respectively. A necessary condition for stability is $\rho_i < 1$, $i = 1, 2$.

The same two cases of parameters reappear: If $\mu_1 < \mu_2$ and $\mu_3 < \mu_4$ then $\rho_i < 1$, $i = 1, 2$ is sufficient for stability of the network under any work conserving (i.e. any non idling) policy. On the other hand, if $\mu_1 > \mu_2$ and $\mu_3 > \mu_4$ then $\rho_i < 1$, $i = 1, 2$ may not be sufficient for stability. In particular, there exist $\gamma_i < 1$ such that the last buffer first served policy (LBFS), which gives priority to the pull activities 2 and 4, will not be stable for $\gamma_i < \rho_i < 1$, $i = 1, 2$.

The discovery of this phenomenon by Kumar and Seidman Kumar and Seidman (1990) (deterministic processing times) and by Rybko and Stolyar Rybko and Stolyar (1992) (exponential

processing times) revolutionized research on multi-class queueing networks, and it is now realized that stability is not a property of the network, but of the policy in conjunction with the network. In our network, this is exemplified by the need to use the pull priority (last buffer first served) for the inherently stable Case 1, and a different policy for the inherently unstable Case 2.

Nevertheless, if $\rho_i < 1$, $i = 1, 2$ then there are some work conserving (non idling) policies which keep all four queues of the KSRS network stable. However, as ρ_i increase towards 1, either for one of the servers or for both together, the network becomes increasingly congested under any policy.

Of particular interest is the behavior of multi-class queueing networks under balanced heavy traffic conditions (cf. Harrison (1988)). Balanced heavy traffic in the KSRS network occurs when $\alpha_1 \to \nu_1$, $\alpha_3 \to \nu_3$. When this happens queues at both servers become congested under any policy.

A diffusion scale analysis of KSRS under balanced heavy traffic considers a sequence $n = 1, 2, \ldots$ of networks, parameterized by α_i^n, $i = 1, 3$ such that $\sqrt{n}(\alpha_i^n - \nu_i)$ converges to some constant as $n \to \infty$. In that case one can hope to show that the diffusion scaled queues,

$$\hat{Q}_i^n(t) = \frac{Q_i^n(nt)}{\sqrt{n}},\ i = 1, 2, 3, 4,$$

will converge to a 4 dimensional Reflected Brownian Motion.

Recent results of Dai and Lin Dai and Lin (2005, 2006) and Ata and Lin Ata and Lin (2008) show that with the use of a maximum pressure policy,$(\hat{Q}_1^n(t), \ldots, \hat{Q}_4^n(t))$, converges to a 4 dimensional reflected Brownian motion which is actually lifted from a 2 dimensional workload process. Henderson, Meyn and Tadic Henderson *et al.* (2003) also considered the KSRS network and obtained stability. Their policy uses affine switching curves, and is similar to our linear threshold policy for the push-pull network.

As the scaling indicates, for the KSRS network under balanced heavy traffic, the diffusion approximation relates to a sequence of networks in which the total number of jobs in the nth network at any time is expected to be of order $\Theta(\sqrt{n})$.

The behavior of the push-pull network, as we will show, is of an entirely different nature: Both servers are active all the time, which can be thought of as operating at $\rho_i = 1$, $i = 1, 2$ and jobs leave the network at the rates ν_i, $i = 1, 3$. At the same time, with i.i.d. processing times the network is positive Harris recurrent. Thus in the push-pull network with $\rho_i = 1$ the number of jobs in the queues $Q_2(t), Q_4(t)$ is expected to be $O(1)$, and it is 0 under diffusion scaling.

4.2 Formulation as MCQN+IVQ

We assume that the processing durations of the jobs in activity $i = 1, 2, 3, 4$ are drawn from a sequence of positive random variables: $\xi_i = \{\xi_i^j,\ j = 1, 2, \ldots\}$. The assumptions that we make regarding the processing durations are as follows:

$$(A1) \quad \lim_{n\to\infty} \frac{\sum_{j=1}^n \xi_i^j}{n} = \frac{1}{\mu_i},\ \text{a.s.}$$
$$\text{for some } \mu_i \in (0, \infty),\ i = 1, 2, 3, 4.$$

$$(A2)\quad \begin{cases} (a) & \xi_i,\ i=1,2,3,4 \\ & \quad \text{are mutually independent i.i.d.} \\ (b) & P(\xi_i^1 \geq x) > 0 \text{ for all } x > 0,\ i=1,3. \\ & \exists k_0^i > 0,\ q_i(\cdot) \geq 0 \text{ with } \int_0^\infty q_i(x)dx > 0: \\ & \quad P(\xi_i^1 + \ldots + \xi_i^{k_0^i} \in dx) \geq q_i(x)dx,\ \ i=1,3. \\ (b') & \text{Compact sets are petite.} \end{cases}$$

Assumptions (A1) require that there exist strong laws of large numbers for the sequences of processing times and that the rate of processing of activity i be μ_i. Assumptions (A2) are to be used in a Markov process setting. (a) implies renewal processing. (b) States that the processing times of the push operations are unbounded and spread-out. (b') is a technical assumption to be made precise in Section 4.4. It is used to prove positive Harris recurrence. We show that under the pull priority policy, (b) implies (b').

We associate counting processes with each activity i:

$$S_i(t) = \sup\{n : \sum_{j=1}^{n} \xi_i^j \leq t\}, \quad t \geq 0.$$

We denote by $T_i(t)$, $i = 1,2,3,4$, the total time that the server allocates to the processing of activity i during the interval $[0,t]$. We require that $T_i(0) = 0$ and that $T_i(\cdot)$ be nondecreasing. Under our policies of full utilization, the servers never idle, thus:

$$T_1(t) + T_4(t) = t, \qquad T_2(t) + T_3(t) = t. \tag{4.1}$$

Note that $T_i(\cdot)$ are Lipschitz, and are therefore absolutely continuous. Thus their derivative exists almost everywhere with respect to Lebesgue measure on $[0,\infty)$.

The number of jobs that have completed processing of activity i by time t is $D_i(t) = S_i(T_i(t))$. Let $Q_i(0)$, $i = 2,4$ be the initial queue lengths. The number of jobs at time t is:

$$Q_i(t) = Q_i(0) + D_{i-1}(t) - D_i(t), \quad i = 2,4. \tag{4.2}$$

We further require that $Q_i(\cdot) \geq 0$ for $i = 2,4$.

The policies which we use in the two cases impose additional conditions on the dynamics of the queues. In the inherently stable Case 1, we use pull priority policy. Hence we will not serve activities 1 or 3 (push activities) unless the corresponding Q_4 or Q_2 are empty. This implies that the allocation processes $T(\cdot)$ need to satisfy:

$$\begin{array}{rcl} \int_0^t Q_4(s)dT_1(s) & = & 0, \\ \int_0^t Q_2(s)dT_3(s) & = & 0. \end{array} \tag{4.3}$$

In the inherently unstable Case 2, we use a linear threshold policy. The linear threshold for server 1 is the line $Q_4(t) = \kappa_1 Q_2(t)$. Server 1 will give preemptive priority to activity 4 only if $0 < Q_4(t) < \kappa_1 Q_2(t)$, and in that case it will not allocate time to activity 1. On the other hand, if $Q_4(t) \geq \kappa_1 Q_2(t)$ then server 1 will give priority to activity 1, to prevent starvation at the queue of server 2, and will not allocate time to activity 4. A symmetric rule is used by server 2, with the linear threshold given by the line $Q_2(t) = \kappa_2 Q_4(t)$. Hence, for Case 2:

$$\begin{array}{rcl} \int_0^t \mathbf{1}\{0 < Q_4(s) < \kappa_1 Q_2(s)\}dT_1(s) & = & 0, \\ \int_0^t \mathbf{1}\{Q_2(s) \leq \frac{1}{\kappa_1} Q_4(s)\}dT_4(s) & = & 0, \\ \int_0^t \mathbf{1}\{0 < Q_2(s) < \kappa_2 Q_4(s)\}dT_3(s) & = & 0, \\ \int_0^t \mathbf{1}\{Q_4(s) \leq \frac{1}{\kappa_2} Q_2(s)\}dT_2(s) & = & 0. \end{array} \tag{4.4}$$

Recall that we require $\kappa_1 > \frac{\mu_3}{\mu_1}$, $\kappa_2 > \frac{\mu_1}{\mu_3}$.

4.3 Fluid Limits and Fluid Models

In this section we assume (A1), and consider the behavior of the push-pull network under fluid scaling. We use the pull priority policy in Case 1, and the linear threshold policy in Case 2.

To study the network under fluid scaling we consider the six dimensional network process $Y(t) = (Q(t), T(t))$, and parameterize it by $n = 1, 2, ...$ as follows: For each n set the initial queue lengths as $Q^n(0)$, and let $Y^n(t)$ be the network process starting from this initial condition, where all the Y^n share the same sequences of random processing times $\xi_i, i = 1, 2, 3, 4$. Denote by $Y^n(t, \omega)$ the realization of the n'th network process for some ω in the sample space. We define *fluid scalings* as:

$$\bar{Y}^n(t, \omega) = \frac{Y^n(nt, \omega)}{n}.$$

A function $\bar{Y}(t) = (\bar{Q}(t), \bar{T}(t))$ is said to be a *fluid limit* of our network if there exists a sequence of integers $r \to \infty$ and a sample path ω such that:

$$\bar{Y}^r(\cdot, \omega) \to \bar{Y}(\cdot), \text{ u.o.c.}$$

It may now be shown (cf. Theorem 4.1 of Dai (1995) or Appendix A.2 of Dai and Lin (2005)) that under Assumption (A1), except for a set of ω of measure zero, fluid limits exist for every ω, and every one of them satisfies the following fluid equations:

$$\begin{array}{l} \bar{Q}_i(t) = \bar{Q}_i(0) + \mu_{i-1}\bar{T}_{i-1}(t) - \mu_i\bar{T}_i(t)\,, i = 2, 4 \\ \bar{Q}_i(t) \geq 0, \quad i = 2, 4 \\ \bar{T}_i(0) = 0,\ \bar{T}_i \text{ is non-decreasing}, \quad i = 1, 2, 3, 4 \end{array} \tag{4.5}$$

as well as

$$\bar{T}_1(t) + \bar{T}_4(t) = t, \qquad \bar{T}_2(t) + \bar{T}_3(t) = t, \tag{4.6}$$

and in addition, under pull priority they satisfy:

$$\begin{array}{rcl} \int_0^t \bar{Q}_4(s)d\bar{T}_1(s) & = & 0, \\ \int_0^t \bar{Q}_2(s)d\bar{T}_3(s) & = & 0, \end{array} \tag{4.7}$$

and under linear threshold policy they satisfy:

$$\begin{array}{rcl} \int_0^t \mathbf{1}\{0 < \bar{Q}_4(s) < \kappa_1\bar{Q}_2(s)\}d\bar{T}_1(s) & = & 0, \\ \int_0^t \mathbf{1}\{\bar{Q}_2(s) \leq \frac{1}{\kappa_1}\bar{Q}_4(s)\}d\bar{T}_4(s) & = & 0, \\ \int_0^t \mathbf{1}\{0 < \bar{Q}_2(s) < \kappa_2\bar{Q}_4(s)\}d\bar{T}_3(s) & = & 0, \\ \int_0^t \mathbf{1}\{\bar{Q}_4(s) \leq \frac{1}{\kappa_2}\bar{Q}_2(s)\}d\bar{T}_2(s) & = & 0. \end{array} \tag{4.8}$$

Equations (4.5)-(4.8) represent a deterministic continuous fluid analog of the stochastic model introduced in the previous section. We shall refer to equations (4.5)–(4.7) as the *fluid model of Case 1*. Similarly we shall refer to (4.5),(4.6) and (4.8) as the *fluid model of Case 2*.

A *fluid solution of Case 1* (*Case 2*) is any pair $(\bar{Q}, \bar{T})$ that satisfies the fluid model equations of Case 1 (Case 2). We say that the fluid model of Case 1 (Case 2) is *stable* if there exists a $\delta > 0$ such that for every fluid solution of Case 1 (Case 2), whenever $|\bar{Q}(0)| = 1$ then $\bar{Q}(t) = 0$ for any $t \geq \delta$.

Our main result in this section is:

Theorem 4.1. *Consider the push-pull network, assume that Assumption (A1) holds, and use in Case 1 the pull priority policy, and in Case 2 the linear threshold policy. Then the fluid model is stable.*

This theorem will be used to show positive Harris Recurrence in the next section. It also immediately leads to the following corollary, which describes the fluid scale behavior of the push-pull network:

Corollary 4.1. *Consider the push-pull network with some fixed initial queue lengths, $Q(0)$, under the assumptions of Theorem 4.1. Then almost surely $Y(nt)/n$ will converge as $n \to \infty$ u.o.c. to a fluid limit $\bar{Y}(t) = (\bar{Q}(t), \bar{T}(t))$ which satisfies:*

$$\bar{T}_i(t) = \theta_i t, \quad \bar{D}_i(t) = \nu_i t, \quad \bar{Q}_i(t) = 0, \quad i = 1, 2, 3, 4.$$

The proof of Theorem 4.1 is by means of a Lyapounov function, f. As in Dai and Weiss (1996), we shall make use of the following elementary Lemma 4.1. Recall that $T_i(t)$ are Lipschitz with constant 1. It then follows that $\bar{T}_i$, and also $\bar{Q}_i(t)$, are Lipschitz, for every fluid solution. Hence they are absolutely continuous with derivative defined almost everywhere. We say that t is a regular point of a fluid solution if the derivatives of $\bar{Y}$ exist at t.

Lemma 4.1. *Let f be an absolutely continuous nonnegative function, and let $\dot{f}$ denote its derivative whenever it exists.*

(i) If $f(t) = 0$ and $\dot{f}(t)$ exists, then $\dot{f}(t) = 0$.

(ii) Assume that for some $\epsilon > 0$ at regular points $t > 0$, whenever $f(t) > 0$ then $\dot{f}(t) \leq -\epsilon$. Then $f(t) = 0$ for all $t \geq f(0)/\epsilon$. Furthermore, $f(\cdot)$ is non increasing and hence once it reaches 0 it stays there forever.

Proof of Theorem 4.1: Case 1: Define $f(t) = \bar{Q}_2(t) + \bar{Q}_4(t)$. Clearly $f(t) \geq 0$ and $f(t) = 0$ if and only if $\bar{Q}(t) = 0$. Also, if $|\bar{Q}(0)| = 1$ then $f(0)$ is bounded (by $B = 1$). We show that f satisfies the conditions of Lemma 4.1, for some ϵ, and hence $f(t) = 0$ for $t > f(0)/\epsilon$, and so if $|\bar{Q}(0)| = 1$, $\bar{Q}(t) = 0$ for $t \geq B/\epsilon$ which proves stability of the fluid model.

Define $\epsilon = \min\{\mu_2 - \mu_1,\ \mu_4 - \mu_3\}$. The values of μ_i in Case 1 ensure that $\epsilon > 0$. We now bound $\dot{f}(t)$ by $-\epsilon$ for all regular time points t at which $f(t) > 0$ by analyzing all possible values of $\bar{Q}_i(t)$, $i = 2, 4$:

- Assume $\bar{Q}_2(t),\ \bar{Q}_4(t) > 0$:

  By (4.7), $\dot{\bar{T}}_1 = \dot{\bar{T}}_3 = 0$ and thus by (4.6), $\dot{\bar{T}}_2 = \dot{\bar{T}}_4 = 1$. As a consequence, $\dot{\bar{Q}}_i(t) = -\mu_i$ for $i = 2, 4$ and

  $$\dot{f} = -(\mu_2 + \mu_4) \leq -\epsilon.$$

- Assume $\bar{Q}_2(t) > 0$, $\bar{Q}_4(t) = 0$:

  By (4.7) $\dot{\bar{T}}_3 = 0$ and thus by (4.6), $\dot{\bar{T}}_2 = 1$. As a consequence,

  $$\dot{f} = \mu_1 \dot{\bar{T}}_1 - \mu_2 - \mu_4 \dot{\bar{T}}_4 = \mu_1 - \mu_2 - (\mu_1 + \mu_4)\dot{\bar{T}}_4 \leq -(\mu_2 - \mu_1) \leq -\epsilon.$$

- Assume $\bar{Q}_2(t) = 0$, $\bar{Q}_4(t) > 0$:

  Similarly to the previous argument,

$$\dot{f} \leq -(\mu_4 - \mu_3) \leq -\epsilon.$$

This completes the proof for Case 1.

Case 2: We use the same technique as in Case 1. Define:

$$f(t) = \begin{cases} (1+\beta)\bar{Q}_2(t) - (\kappa_2 - \beta)\bar{Q}_4(t) & \text{if } \bar{Q}_2(t) \geq \kappa_2 \bar{Q}_4(t), \\ (1+\beta)\bar{Q}_4(t) - (\kappa_1 - \beta)\bar{Q}_2(t) & \text{if } \bar{Q}_4(t) \geq \kappa_1 \bar{Q}_2(t), \\ \beta(\bar{Q}_2(t) + \bar{Q}_4(t)) & \text{otherwise.} \end{cases}$$

where

$$\beta = \frac{1}{2}\min\{\frac{\kappa_1 - \frac{\mu_3}{\mu_1}}{1 + \frac{\mu_3}{\mu_1}}, \frac{\kappa_2 - \frac{\mu_1}{\mu_3}}{1 + \frac{\mu_1}{\mu_3}}\}.$$

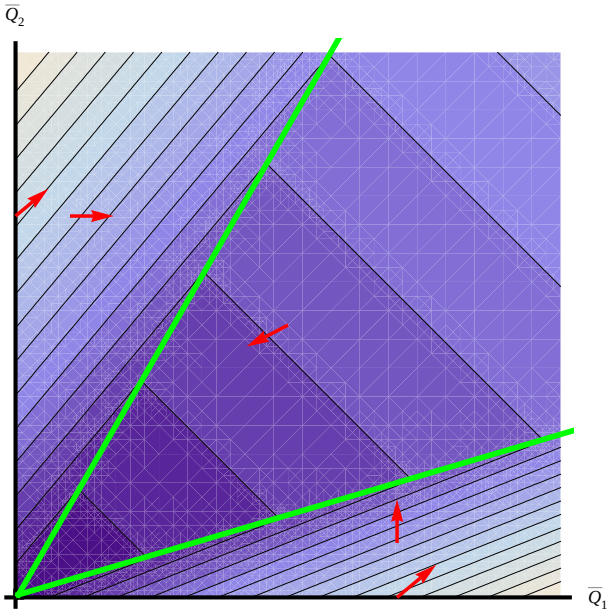

Figure 4.3: Lyapounov function for case 2.

An example contour plot of this Lyapounov function and the mean drift arrows (in red) is in Figure 4.3. Again, it is easily seen that $f(t) \geq 0$ and $f(t) = 0$ if and only if $\bar{Q}(t) = 0$, and if $|\bar{Q}(0)| = 1$ then $f(0)$ is bounded by some finite value B.

All we need to do is find an ϵ to satisfy the conditions of Lemma 4.1. We now bound $\dot{f}(t)$ for all regular time points t at which $f(t) > 0$, by analyzing all possible values of $\bar{Q}_i(t)$, $i = 2, 4$:

- Assume $\frac{1}{\kappa_2}\bar{Q}_2(t) < \bar{Q}_4(t) < \kappa_1 \bar{Q}_2(t)$:

  Then $f(t) = \beta(\bar{Q}_2(t) + \bar{Q}_4(t))$, and in this region both servers use pull priority. Hence

$$\dot{f} = \beta(\mu_1 \dot{T}_1 - \mu_2 \dot{T}_2 + \mu_3 \dot{T}_3 - \mu_4 \dot{T}_4)$$

  and by (4.8) we have that $\dot{T}_1 = \dot{T}_3 = 0$ and thus $\dot{T}_2 = \dot{T}_4 = 1$. Hence

$$\dot{f} = -\beta(\mu_2 + \mu_4).$$

- Assume $0 < \bar{Q}_4(t) \leq \frac{1}{\kappa_2}\bar{Q}_2(t)$:

  Then $f(t) = (1+\beta)\bar{Q}_2(t) - (\kappa_2 - \beta)\bar{Q}_4(t)$ and in this region both queues are not empty, and server 1 gives priority to pull while server 2 gives priority to push. Hence

$$\dot{f} = (1+\beta)(\mu_1\dot{T}_1 - \mu_2\dot{T}_2) - (\kappa_2 - \beta)(\mu_3\dot{T}_3 - \mu_4\dot{T}_4),$$

  and by (4.8) we have that $\dot{T}_1 = \dot{T}_2 = 0$ and thus $\dot{T}_3 = \dot{T}_4 = 1$. Hence

$$\dot{f} = -(\kappa_2 - \beta)(\mu_3 - \mu_4).$$

- Assume $0 < \bar{Q}_2(t) \leq \frac{1}{\kappa_1}\bar{Q}_4(t)$:

  The analysis is symmetric to the previous case, and yields:

$$\dot{f} = -(\kappa_1 - \beta)(\mu_1 - \mu_2).$$

- Assume $\bar{Q}_2(t) > 0,\ \bar{Q}_4(t) = 0$:

  Again $f(t) = (1+\beta)\bar{Q}_2(t) - (\kappa_2 - \beta)\bar{Q}_4(t)$, and in this region server 2 gives priority to push. With $\bar{Q}_4(t) = 0$ we cannot say where server 1 will work. Hence

$$\dot{f} = (1+\beta)(\mu_1\dot{T}_1 - \mu_2\dot{T}_2) - (\kappa_2 - \beta)(\mu_3\dot{T}_3 - \mu_4\dot{T}_4)$$

  and by (4.8) $\dot{T}_2 = 0$ and as a result $\dot{T}_3 = 1$. Hence:

$$\begin{aligned}
\dot{f} &= (1+\beta)\mu_1\dot{T}_1 - (\kappa_2 - \beta)(\mu_3 - \mu_4\dot{T}_4) \\
&= (1+\beta)\mu_1\dot{T}_1 - (\kappa_2 - \beta)[\mu_3(\dot{T}_1 + \dot{T}_4) - \mu_4\dot{T}_4] \\
&= -(\kappa_2 - \beta)[(\mu_3 - \frac{1+\beta}{\kappa_2 - \beta}\mu_1)\dot{T}_1 + (\mu_3 - \mu_4)\dot{T}_4] \\
&\leq -(\kappa_2 - \beta)\min\{\mu_3 - \frac{1+\beta}{\kappa_2 - \beta}\mu_1\ ,\ \mu_3 - \mu_4\}.
\end{aligned}$$

- Assume $\bar{Q}_4(t) > 0,\ \bar{Q}_2(t) = 0$: The analysis is symmetric to the previous case, and yields:

$$\dot{f} \leq -(\kappa_1 - \beta)\min\{\mu_1 - \frac{1+\beta}{\kappa_1 - \beta}\mu_3\ ,\ \mu_1 - \mu_2\}.$$

All five bounds above are negative, and we choose $-\epsilon$ as their maximum. This completes the proof. □

Remark: So far in this section we assumed that the nth system starts with queue lengths $Q^n(0)$, and that all the jobs in the system had no previous processing, so that the $S_i(t)$ are counting processes, with intervals ξ_i which have long term rate μ_i. A more general model assumes that at time 0 the head of the line job in each queue or infinite supply has received some processing, and let $\xi_{i,0}$ be the residual processing time of this first job. Then the first interval is a residual processing time with a different mean from the other ξ_i^j, $j > 1$. In that case $S_i(t)$ are delayed counting processes. We now associate with the nth system an initial state consisting of $Q_i^n(0), \xi_{i,0}^n, i = 1,2,3,4$. All the results of this section remain valid and unchanged as long as we assume that $\xi_{i,0}^n/n \to 0$ a.s. (see Bramson (1998a)).

4.4 Positive Harris Recurrence

In this section we add the set of Assumptions (A2) to Assumption (A1), and use the fluid stability results from the previous section to show that the push-pull network under our policies can be described by a positive Harris recurrent Markov chain. To do so we adapt the framework developed by Dai Dai (1995), see also Bramson (1998a).

We begin by defining the network state process. Denote by $U_i(t), V_i(t)$ the residual processing times of the head of the line activities which are in process or preempted at the current time t. $U_i(t), i = 2, 4$ is for the pull activities and $V_i(t), i = 1, 3$ is for the push activities. Now denote the *network state process* by,

$$X(t) = (Q(t), U(t), V(t)).$$

The state space is $\mathbb{S} = \mathbb{Z}_+^2 \times \mathbb{R}_+^2 \times \mathbb{R}_+^2$, and $|X(t)|$ is the sum of the components of $X(t)$. Since the evolution of $X(t)$ between arrivals and departures is deterministic, $X(t)$ is piecewise deterministic, and it is not difficult to show that $X(t)$ is a piecewise deterministic strong Markov process (cf. Davis (1984)):

Proposition 4.1. *Under Assumptions (A1), (A2a), $X = \{X(t),\ t \geq 0\}$ is a strong Markov process with state space $\mathbb{S}$.*

Let $P^t(x, \cdot)$ be the transition probability of X. That is for $x \in \mathbb{S}$, $B \in \mathcal{B}(\mathbb{S})$,

$$P^t(x, B) \equiv P_x\{X(t) \in B\} \equiv P\{X(t) \in B \,|\, X(0) = x\}.$$

A nonzero measure π on $(\mathbb{S}, B(\mathbb{S}))$ is *invariant* for X if π is σ-finite, and for each $t \geq 0$,

$$\pi(B) = \int_{\mathbb{S}} P^t(x, B)\, \pi(dx), \quad B \in \mathcal{B}(\mathbb{S}).$$

Let $\tau_A = \inf\{t \geq 0 : X(t) \in A\}$. We say that X is *Harris recurrent* if there exists some σ-finite measure ν on $(\mathbb{S}, B(\mathbb{S}))$, such that for all $A \in \mathcal{B}(\mathbb{S})$ with $\nu(A) > 0$ we have $P_x(\tau_A < \infty) = 1$ for all $x \in \mathbb{S}$. If X is Harris recurrent then an essentially (up to a positive scalar multiplier) unique invariant measure π exists. When π is finite (in which case we normalize it to a probability measure) we say that X is *positive Harris recurrent*. Positive Harris recurrence is a common notion of stability since it implies certain ergodicity properties. For example, given $f : \mathbb{S} \mapsto \mathbb{R}_+$, denote

$$\pi(f) = \int_{\mathbb{S}} f(x)\, \pi(dx)$$

whenever the integral makes sense. Then if $\pi(|f|) < \infty$:

$$\lim_{t\to\infty} \frac{1}{t} \int_0^t f(X(s))ds = \pi(f) \quad P_x \text{ a.s. for each } x \in \mathbb{S}.$$

To establish positive Harris recurrence of $X(t)$, we need a further concept: A non-empty set A is said to be *petite* if there exists a probability distribution $\mathbf{a}$ on $(0, \infty)$ and a nontrivial measure ν on $(\mathbb{S}, \mathcal{B}(\mathbb{S}))$, such that for all $x \in A$

$$\int_0^\infty P^t(x, B)\mathbf{a}(dt) \geq \nu(B), \qquad \text{for all } B \in \mathcal{B}(\mathbb{S}).$$

Petiteness of A may be interpreted as the property that all sets B are "equally accessible" from any $x \in A$. For more on Markov processes, positive Harris recurrence and petite sets, see Meyn and Tweedie (1993a) for an introduction and discrete time results, and Meyn and Tweedie (1993b,c) for continuous time results.

We are now in a position to rigorously define Assumption (A2b'):

$$(A2b') \qquad A = \{x : |x| \leq \sigma\} \text{ is petite for any } \sigma > 0.$$

Our main result in this chapter is:

Theorem 4.2. *Under Assumptions (A1), (A2a) and (A2b'), the network state process X is Positive Harris Recurrent for Case 1 under the pull priority policy and for Case 2 under the linear threshold policy. Furthermore, for Case 1 we may substitute Assumptions (A2b') with (A2b).*

Proof. The proof uses the framework of Dai Dai (1995). The main theorem in that paper (Theorem 4.2) states that if the fluid model of a multi-class queueing network (with exogenous arrival streams) is stable then the associated Markov process is positive Harris recurrent. However, our model does not fall in that scope and hence we must adapt the proof.

The following discussion outlines the adaptation. Dai shows that positive Harris recurrence of the network state process follows directly from two statements:

(i) Convergence of a fluid scaled process scaled by its initial state: There exists $\delta > 0$ such that

$$\lim_{|x|\to\infty} \frac{1}{|x|} E_x |X(\delta|x|)| = 0.$$

(ii) Petiteness of closed bounded sets as in our Assumption (A2b').

The arguments of Dai that statements (i) and (ii) imply positive Harris recurrence are valid also for our push-pull network, and so to prove the theorem we need to show that (i) and (ii) hold.

The main result of Dai is to show that stability of the fluid model, as defined in the previous Section 4.3, implies (i). The proof that fluid stability implies (i) needs no changes in our case. Hence, under Assumptions (A1) and (A2a), our Theorem 4.1, in which we have proved stability of the fluid model, implies (i) for the push-pull network.

Hence, if we make Assumption (A2b'), the positive Harris recurrence of the push-pull network follows.

The technical Assumption (ii), that all compact sets are petite is awkward, as it is difficult to check. Thus it is useful instead of Assumption (A2b') to find a sufficient condition which is easier to check. Dai's paper asserts that for multi-class queueing networks with an exogenous input stream the assumption that inter-arrival times have a spread out distribution with unbounded support implies (ii). His proof follows directly from the earlier work of Meyn and Down Meyn and Down (1994), who proved the same result for generalized Jackson networks. This needs to be extended to the case of infinite supply of work. The difference is that with infinite supply of work the output process from an infinite virtual queue is in general not independent of the state of the other queues. Guo and Zhang Guo and Zhang (2007) have adapted Meyn and Down's

ideas to a reentrant line with infinite supply of work where the policy is to give lowest priority to the activity with the infinite supply.

The following Lemma 4.2 extends these results, and shows that in Case 1, under pull priority, the Assumption (A2b) implies (A2b'), and hence positive Harris recurrence □

Lemma 4.2. *For the network state process X, operating with the pull priority policy, under Assumptions (A1) and (A2a), the Assumption (A2b) implies (A2b').*

The proof of the above Lemma is called a "Minorization". It is in the next section. Up to now, we were unable to provide a similar result for the more complex linear threshold policy.

4.5 A Minorization Proof

The proof requires some more concepts (cf. Meyn and Tweedie (1993b)): We say that X is *ψ-irreducible*, if there exists a measure ψ on $(\mathbb{S}, \mathcal{B}(\mathbb{S}))$ such that, whenever $\psi(A) > 0$, we have $P_x\{\tau_A < \infty\} > 0$ for all $x \in \mathbb{S}$.

Let $\mathbf{a}$ be a probability distribution on $\mathbb{R}_+$. Define the *Markov transition function* $K_{\mathbf{a}}$ as

$$K_{\mathbf{a}}(x, \cdot) = \int_0^\infty P^t(x, \cdot)\, \mathbf{a}(dt).$$

A *continuous component* of $K_{\mathbf{a}}$ is a non-negative function $T(x, A)$ which is lower semi-continuous in x, and satisfies

$$K_{\mathbf{a}}(x, A) \geq T(x, A), \quad x \in \mathbb{S}, \quad A \in \mathcal{B}(\mathbb{S}),$$

We say that X is a *T-process* if there exists a distribution $\mathbf{a}$ such that $K_{\mathbf{a}}$ possesses a continuous component T, with $T(x, \mathbb{S}) > 0$ for all $x \in \mathbb{S}$. The following proposition (cf. Theorem 4.1(i) of Meyn and Tweedie (1993b)), connects ψ-irreducible T-processes and petiteness of compacts.

Proposition 4.2. *If X is a ψ-irreducible T-process then every compact set in $\mathcal{B}(\mathbb{S})$ is petite.*

We say that a state x^* is reachable if $\int_0^\infty P^t(x, O)dt > 0$ for every open neighborhood O of x^* and every $x \in \mathbb{S}$. It can be shown (cf Guo and Zhang (2007)) that if X is a T-process with a reachable point x^* then it is also ψ-irreducible with $\psi(\cdot) = T(x^*, \cdot)$.

Returning to our push-pull queueing network with pull priority, it is easy to see, by Assumption (A2b), that the state $Q(t) = 0, U(t) = 0, V(t) = 0$ is reachable.

Thus the main part of the proof is to show that X is a T-process: We need to construct a lower semi-continuous function $T(\cdot, A)$ and a transition kernel $K_{\mathbf{a}}(\cdot, A)$, so that $K_{\mathbf{a}}(x, A) \geq T(x, A)$, for all $(x, A) \in (\mathbb{S}, \mathcal{B}(\mathbb{S}))$.

Following Meyn and Down Meyn and Down (1994) the construction is in several steps. The crucial step in the construction of T is to consider the initial state in a bounded rectangle, the set of states to be reached is an empty system with both servers engaged in push activity, and to then bound the probability of reaching this set after a deterministic integer time by a continuous function.

For an integer ℓ define $R_\ell = \{0,\dots,\ell\}^2 \times [0,\ell)^2 \times [0,\ell)^2$. Now take the initial state at time 0 as $x_0 \in R_\ell$.

Define $Z(t) = (Q_2(t), Q_4(t), U_2(t), U_4(t))$. Then the network state process is $X(t) = (Z(t), V(t))$. Let $A_1, A_3 \in \mathcal{B}(\mathbb{R}_+)$. The set to be reached is the set $\{Z = 0, V \in A_1 \times A_3\}$. For an integer time n_l we will bound $P_{x_0}(Z(n_\ell) = 0, V(n_\ell) \in A_1 \times A_3)$ from below by a function $T_l'(x_0, A_1, A_3)$, which is continuous in x_0.

Define two events:

$$D_\ell = \{\sum_{j=1}^{k_0^i} \xi_i^j \le \frac{n_\ell}{4}, \xi_i^{k_0^i+1} \ge 2n_\ell \text{ for } i = 1,3\},$$

for large n_l it has a positive probability, since we assume that the distribution of ξ_1^1, ξ_3^1 has infinite support.

$$E_{L,\ell} = \{\xi_i^j \le L, j = 1,\dots,\ell + k_0^i \text{ for } i = 2,4\},$$

where L is taken large enough such that, $\epsilon_{L,\ell} = P(E_{L,\ell}) > 0$. If we require that

$$n_\ell > 4\ell + 2(\ell-1)L + 2\frac{n_\ell}{4} + (k_0^1 + k_0^3)L, \tag{4.9}$$

that is set n_ℓ to

$$n_\ell > 8\ell + 4(\ell-1)L + 2(k_0^1 + k_0^3)L,$$

then we have that the event $D_\ell \cap E_{L,\ell}$ implies that at time n_ℓ, $Z(n_\ell) = 0$ and server 1 (server 3) is engaged in push activity 1 (push activity 3) with the long $k_0^1 + 1$st ($k_0^3 + 1$st) job from the infinite supply. To see this, recall that our policy is head of the line with low priority to push activities. Therefore prior to the first time that the servers are both working on the long push activities, at least one of them is working on pull activities or on the first k_0^i push activities. The expression (4.9) is an upper bound on the total amount of work that has to be done, and it will therefore be completed by time n_l. The long push activities will of course not be complete by time n_l.

With the above definitions in hand,

$$\begin{aligned}
&P_{x_0}(Z(n_\ell) = 0, V(n_\ell) \in A_1 \times A_3) \\
&\quad \ge P_{x_0}(Z(n_\ell) = 0, V(n_\ell) \in A_1 \times A_3, D_\ell, E_{L,\ell}) \\
&\quad = P_{x_0}(V(n_\ell) \in A_1 \times A_3, D_\ell, E_{L,\ell}) \\
&\quad = \epsilon_{L,\ell} P_{x_0}(V(n_\ell) \in A_1 \times A_3, D_\ell | E_{L,\ell}).
\end{aligned}$$

The number of jobs to be processed by activity $i = 2,4$ by time n_ℓ, apart from the residuals, is

$$\ell_i = Q_i(0) - I\{Q_i(0) > 0\} + k_0^{i-1}.$$

Now define the truncation $\zeta_i^j = I\{\xi_i^j \le L\}\, \xi_i^j$ for $i = 2,4$, and observe that when D_ℓ occurs and conditional on $E_{L,\ell}$,

$$\begin{aligned}
V_1(n_\ell) &= V_1(0) + U_4(0) + \sum_{j=1}^{k_0^1} \xi_1^j + \sum_{j=1}^{\ell_4} \xi_4^j + \xi_1^{k_0^1+1} - n_\ell \\
&= V_1(0) + U_4(0) + \sum_{j=1}^{k_0^1} \xi_1^j + \sum_{j=1}^{\ell_4} \zeta_4^j + \xi_1^{k_0^1+1} - n_\ell,
\end{aligned}$$

with a similar expression for $V_3(n_\ell)$.

Denote the distribution of ξ_i^1 by η_i and the k_0^i fold convolutions of these distributions by $\eta_i^{*k_0^i}$ for $i = 1, 3$. Also, for $i = 2, 4$, use η_i' to denote the distribution of $\sum_{j=1}^{\ell_i} \zeta_i^j$.

We now have

$$\begin{array}{l} P_{x_0}(V(n_\ell) \in A_1 \times A_3, D_\ell | E_{L,\ell}) = \\ \int I_{s_1,s_3,t_1,t_3,r_2,r_4} \eta_1^{*k_0^1}(ds_1)\eta_3^{*k_0^3}(ds_3)\eta_1(dt_1)\eta_3(dt_3)\eta_2'(dr_2)\eta_4'(dr_4) \end{array} \tag{4.10}$$

where the integral is on the range $(s_1, s_3, t_1, t_3, r_2, r_4) \in [0, \infty)^6$, and the integrand is the indicator function

$$\begin{array}{l} I_{s_1,s_3,t_1,t_3,r_2,r_4} = \\ I\{V_1(0) + U_4(0) + s_1 + r_4 + t_1 - n_\ell \in A_1\} \cdot \\ I\{V_3(0) + U_2(0) + s_3 + r_2 + t_3 - n_\ell \in A_3\} \cdot \\ I\{s_1 \leq \frac{n_\ell}{4}\} I\{s_3 \leq \frac{n_\ell}{4}\} I\{t_1 \geq 2n_\ell\} I\{t_3 \geq 2n_\ell\}. \end{array} \tag{4.11}$$

We now use Assumption (A2b) to get,

$$P_{x_0}(V(n_\ell) \in A, G_{n_\ell} | E_{L,\ell}) \geq \int I_{s_1,s_3,t_1,t_3,r_2,r_4} q_1(s_1)ds_1 q_3(s_3)ds_3 \eta_1(dt_1)\eta_3(dt_3)\eta_2'(dr_2)\eta_4'(dr_4) \tag{4.12}$$

We define the function $T_\ell'(x_0, A)$ as $\epsilon_{L,\ell}$ multiplied by the integral in (4.12). It is evident that T_ℓ' is continuous in each of the coordinates $V_1(0), V_3(0), U_2(0), U_4(0)$ and hence it is continuous in x_0. It is also strictly positive, as required.

For every x_0 this $T_\ell'(x_0, A_1 \times A_3)$ is now defined for $A_1 \times A_3 \in \mathcal{B}(\mathbb{R}^2)$. We can extend it to a measure on the whole $\mathcal{B}(\mathbb{R}^2)$, so that $T_\ell'(x_0, A) > 0$ for every $A \in \mathcal{B}(\mathbb{R}^2)$ with positive Lebesgue measure, and so that $T_\ell'(x_0, A)$ is continuous in x_0 and satisfies

$$P_{x_0}(Z(n_\ell) = 0, V(n_\ell) \in A) \geq T_\ell'(x_0, A).$$

The remainder of the construction of the continuous component T follows exactly the steps of Meyn and Down Meyn and Down (1994).

Remark: The above proof can be extended to a proof for petiteness of compacts of the network state process of a multi-class queueing network with infinite virtual queues (cf Section 3 of Nazarathy and Weiss (2008b)) operating under a policy that gives lowest priority to the infinite virtual queues. Writing this statement and proof does not require any further ideas than those presented here.

Part III

Output Variance

CHAPTER 5

ASYMPTOTIC VARIANCE RATE OF OUTPUTS

The purpose of this chapter is to serve as an introduction to Chapters 6 and 7 which contain our main contribution with regards to the asymptotic variance rate of outputs. In Chapter 6 we consider outputs from finite capacity queues with overflows and in Chapter 7 we consider outputs of examples of multi-class queueing networks with infinite virtual queues.

In Chapters 3 and 4 we presented our results about control of queueing networks that attempt to minimize holding costs or insure maximum utilization and throughput. These types of performance measures are often reasonable in many applications: In classical queueing theory where the typical phrase for a *job* is a *customer* (i.e. a person), it only makes sense to concentrate most attention on performance analysis of sojourn times or delays whose mean is directly related to the holding costs by Little's result. On the contrary in supply chains, manufacturing and certain type of communication networks, the output processes of the system are typically of great importance. A first measure of interest regarding the output process is the throughput and typically the next thing that is of interest is the variability of these processes. We now shift attention to analysis of the variability of outputs.

There is not one agreed upon performance measure for the variability of output point processes and it appears that different applications are best analyzed by different types of performance measures. We choose to concentrate on the *asymptotic variance rate of the outputs* which measures the linear growth rate of the variance of the number of outputs in the interval $[0, t]$. We define it more rigorously in the discussion that follows. This is a natural measure to consider if one assumes that the output process follows some central limit theorem law, as it can be immediately used to estimate the variance and the distribution of the number of outputs in a long time interval. On the contrary it appears that this performance measure does not capture short term variability when the point process at hand is a non-renewal process.

This chapter is organized as follows: We start with Section 5.1 where we make definitions and overview some methods that may be used to evaluate the asymptotic variance rate. We then move on to present some introductory examples: In Section 5.2 we discuss the asymptotic

variance rate of a GI/G/1 queue with input rate λ and service mean μ^{-1}. We know that when $\lambda < \mu$, the asymptotic variance rate of outputs equals the asymptotic variance rate of the input process and when the inequality is reversed it equals that of a renewal processes generated by services. The asymptotic variance rate in the critically loaded case ($\lambda = \mu$) is still an open problem. In Section 5.3 we look at the special case of the M/G/1 queue with $\lambda < \mu$. We show how the asymptotic variance rate can be used to obtain the cross moment between the busy cycle and the number served in it. This result is used in the next Section, 5.4 where we analyze the asymptotic variance rate of the outputs of the inherently stable push-pull network operating under pull priority. The results of that Section are presented here as a prelude to the more general diffusion limit results in Chapter 7.

5.1 Methods for Calculating Asymptotic Variance Rate

We denote by $D(t)$ a counting point process, i.e. $D(t)$ counts the number of events during the interval $[0, t]$. We shall typically assume it counts the number of outputs from a queueing system. We now define the asymptotic variance rate as:

$$\overline{V} = \lim_{t \to \infty} \frac{\text{Var}(D(t))}{t},$$

whenever the limit exists. We shall typically ignore situations in which the limit does not exists, such cases have been termed "long range dependent" processes in the literature. Although extremely interesting and important, they are not within the scope of this thesis. The first order rate of increase of the point process (throughput) is labeled by:

$$\lambda^* = \lim_{t \to \infty} \frac{\mathbb{E}\left[D(t)\right]}{t}.$$

When the point process at hand is a renewal process then $\overline{V} = c^2 \lambda^*$ where $c^2 = \frac{\sigma^2}{m^2}$ denotes the squared coefficient of variation (SCV) of the stationary inter-output time having expectation m and variance σ^2, cf. Asmussen (2003, pp. 161). In the special case of a Poisson process we have $\overline{V} = \lambda^*$.

Evaluation of $\overline{V}$ is important in manufacturing type settings. When the system operates for a long duration, T, the variance of the number of items produced is approximately $\overline{V}T$. Several studies have investigated computational procedures that evaluate this quantity for the output of a series of queues, cf. Miltenburg (1987); Hendricks (1992); Gershwin (1993); Hendricks and McClain (1993); Tan (1999, 2000); Ciprut *et al.* (1999). The results in our work are not of a computational nature. Instead we exploit some analytic methods for determining $\overline{V}$. Here are the methods we use:

Markovian Arrival Processes Method: A Markovian Arrival Process (MAP) is a point process that is associated with a "background" finite state space CTMC. The process essentially counts transitions of the CTMC (including fake transitions from a state to itself). There are well known matrix formulas for the asymptotic variance rate of these processes. MAPs can easily be used to model outputs from queueing systems whose queue level

process can be modeled as a CTMC with a finite number of states. We do so in the next chapter and present the details regarding MAPs there.

Diffusion Limit Method: One can look at diffusion scaled versions of a counting process:

$$\hat{D}^n(t) = \frac{D(nt) - \bar{D}(nt)}{\sqrt{nt}}, n = 1, 2, ...$$

Here, $\bar{D}(t)$ is the mean of the process at time t. When the above sequence of processes converges to some diffusive process, then typically the asymptotic variance rate of the diffusive process equals that of the original process. We employ such an analysis in Chapter 7.

Renewal Reward Method: If the queueing process that is analyzed has a regenerative structure then we can associate with it a renewal reward process as follows. Let $\{(X_i, Y_i), i = 0, 1, \ldots\}$ be a sequence of independent vectors (the coordinates are not necessarily independent) where $(X_i, Y_i), i \geq 1$ are identically distributed. Assume that $\{X_i, i = 0, 1, \ldots\}$ is the sequence of inter-regeneration times (e.g. busy cycles of a queue) and define the renewal process $A(t)$:

$$A(t) = \inf\{n : \sum_{j=0}^{n} X_j > t\}.$$

Now let $\{Y_i, i = 0, 1, \ldots\}$ denote the number of outputs from the queueing system during the i'th regeneration cycle. Define the renewal reward process

$$D(t) = \sum_{i=0}^{A(t)-1} Y_i.$$

Then $D(t)$ counts the number of outputs during the interval $[0, \tau]$ where $\tau \leq t$ is the last regeneration time that is not later than t. It is clear that as long as $P(X_1 < \infty) = 1$, then the asymptotic variance rate of $D(t)$ is the asymptotic variance rate of the outputs, $\overline{V}$.

Let us denote, $x_k = \mathbb{E}[X_1^k]$, $y_k = \mathbb{E}[Y_1^k]$ and $n_{k\ell} = \mathbb{E}[X_1^k Y_1^\ell]$. Smith (1955), presents a formula for $\overline{V}$ when $x_2, y_2, n_{11} < \infty$ and the distribution of X_1 is spread-out. Brown and Solomon (1974) extend Smith's result and approximate the y-intercept of the linear asymptote. Their result also requires that $y_3 < \infty$ and $n_{12} < \infty$. These conditions ensure that:

$$\text{Var}(D(t)) = \overline{V}t + \overline{B} + o(1)$$

Where the asymptotic variance rate is[1]:

$$\overline{V} = \frac{1}{x_1}(\frac{x_2 y_1^2}{x_1^2} - 2\frac{n_{11} y_1}{x_1} + y_2). \tag{5.1}$$

Observe that in case of a renewal process, $y_1 = 1$, $y_2 = 0$, $n_{11} = x_1$ and as expected, $\overline{V} = \frac{x_2 - x_1^2}{x_1^3}$. The formula for $\overline{B}$ depends on the distribution of the first renewal-reward

[1] Whitt (2002) obtains a special case of this formula by means of a diffusion approximation when X_i and Y_i are independent.

pair[2], X_0, Y_0:

$$\overline{B} = d - \overline{V}\mathbb{E}\left[X_0\right] + \mathrm{Var}(Y_0 - \frac{y_1}{x_1}X_0).$$

Where,

$$d = \frac{5}{4}\frac{x_2^2 y_1^2}{x_1^4} - \frac{2}{3}\frac{x_3 y_1^2}{x_1^3} + 2\frac{n_{21}y_1}{x_1^2} - 3\frac{x_2 y_1 n_{11}}{x_1^3} + \frac{n_{11}^2}{x_1^2} + \frac{1}{2}\frac{x_2 y_2}{x_1^2} - \frac{n_{12}}{x_1}.$$

In case (X_0, Y_0) is identically 0 then $\overline{B} = d$.

Other approaches: There are other approaches which we haven't explored: For example, one may analyze queueing systems using *large deviations theory* and *strong approximations*. A more classic method is to analyze so-called *traffic sets* of a Markovian process (cf. Disney and Kiessler (1987) and Barnes and Disney (1990)). Other important results are related to *point processes in random environments*, as in Whitt (2002, pp. 312). We make use of these results which also require solving Poisson's equation, in the next chapter.

5.2 The Infinite Buffer Single Server Queue

We have the following theorem:

Theorem 5.1. *Consider a single server GI/G/1 queue with inter-arrival and service distributions having a second moment. Denote the squared coefficients of variation of the inter-arrival and service times by c_a^2 and c_s^2 respectively. Denote by $Q(t)$, the queue level process at time t and $D(t)$ the cumulative number of departures during the interval $[0,t]$. Define the asymptotic variance rate of the outputs:*

$$\overline{V} = \lim_{t\to\infty} \frac{Var(D(t))}{t},$$

whenever the limit exists. Then,

(i) If $\lambda < \mu$ then $\overline{V} = \lambda c_a^2$ for any distribution of $Q(0)$.

(ii) If $\lambda > \mu$ then $\overline{V} = \mu c_s^2$ for any distribution of $Q(0)$.

This intuitive theorem states that any stable single server queue preserves the asymptotic variance rate of the arrival process and any unstable single server queue overrides the asymptotic variance of the arrival process by that of the service process.

Proof. Case (i) is proved in the same way as Lemma 6.1 in the next chapter. We omit the details. Case (ii) is immediate because if $\lambda > \mu$, then after some random time which is finite w.p. 1, the queue will never empty and produce with a renewal process having asymptotic variance rate μc_s^2 □

Handling the borderline case of $\lambda = \mu$ appears more complicated. Personal communication with W. Whitt 2008 along with some simulation experiments indicate that it is possible that the $\lambda = \mu$ case contains a singularity in terms of the asymptotic variance rate. This is still not clear and will hopefully be resolved in the near future[3].

[2]It would be somewhat interesting to obtain this formula for single server queueing systems, even for the M/M/1 queue. It is an open question. It is known that $\overline{B} = 0$ when the M/M/1 is stationary by Burke's theorem.

[3]B. Fralix (Personal Communication 2008) has independently found a formula for the variance function of the outputs. His formula is in terms of probabilities of two independent busy period random variables and holds also for the $\lambda = \mu$ case. It may in principle be used to evaluate the asymptotic variance rate.

5.3 Example: The Stable M/G/1 Queue

There are some obvious special cases to Theorem 5.1. The simplest is if one considers a stationary M/M/1 queue, then by Burke's theorem the output is Poisson and thus $\overline{V} = \lambda$.

Another case that may be handled is the stable (not necessarily stationary) M/G/1 queue operating under some work conserving policy. The arrival rate is λ and the service mean is μ^{-1}. We denote $\rho = \lambda/\mu$ and assume $\rho < 1$. We further assume that the service time distribution has a squared coefficient of variation, c^2 and a Laplace transform $H^*(\cdot)$. In this case we can use the renewal reward method discussed in Section 5.1 to obtain the asymptotic variance rate of the outputs. To do so we need to calculate the first, second and cross moments of the busy period, idle period and number served during a busy period. Calculation of these first and second moments is a simple exercise using classic queueing results: functional equations for transforms of these random variables. For the cross moment, one can use a similar functional equation for the joint Laplace transform and generating function of both the busy period and number served (cf. Prabhu (1998))[4].

For illustrative purposes, we shall do the reverse: Calculate the cross moment by means of the asymptotic variance rate. While this calculation does not introduce any new results we believe that possibly this type of "mean value analysis method" can be useful in other types of settings in which one knows the asymptotic variance rate and wants to calculate certain moments or cross moments.

The following functional equations are well known (cf. Kleinrock (1974)):

$$G^*(s) = H^*(s + \lambda - \lambda G^*(s)), \qquad F^*(z) = zH^*(\lambda - \lambda F^*(z)).$$

Here $G^*(\cdot)$ is the Laplace transform of the busy period duration and $F^*(z)$ is the z-transform of the number of jobs served during the busy period. Differentiating each of the above equations seperatly and setting $s = 0$ and $z = 1$ respectively, we obtain equations for the moments which are easily solved (separately) to yield the well known results:

$$\begin{array}{rclrcl} \mathbb{E}[B] & = & \frac{1/\mu}{1-\rho}, & \mathbb{E}[N] & = & \frac{1}{1-\rho}, \\ \mathbb{E}[B^2] & = & \frac{c^2+1}{\mu^2(1-\rho)^3}, & \mathbb{E}[N^2] & = & \frac{1+\rho^2 c^2}{(1-\rho)^3}. \end{array} \tag{5.2}$$

In addition, the idle period is exponentially distributed with rate λ, so,

$$\mathbb{E}[I] = \frac{1}{\lambda}, \qquad \mathbb{E}[I^2] = \frac{2}{\lambda^2}. \tag{5.3}$$

We now use Theorem 5.1 for the left hand side and formula (5.1) for the right hand side of the equation:

$$\lambda = \frac{1}{\mathbb{E}[B+I]}\left(\frac{\mathbb{E}[(B+I)^2]\mathbb{E}[N^2]}{(\mathbb{E}[B+I])^2} - 2\frac{\mathbb{E}[N]\mathbb{E}[(B+I)N]}{\mathbb{E}[B+I]} + \mathbb{E}[N^2]\right).$$

Here X_i are taken to be busy cycles composed of a busy period and idle period. And Y_i are the number of jobs served during a busy cycle (also during a busy period). Now we may plug in,

[4] An alternative derivation is due to personal communication with Y. Kerner and D. Perry, August 2008.

(5.2) and (5.3) and use the fact that I and N are independent. Solving, we obtain:

$$\mathbb{E}\left[NB\right] = \frac{1+\rho c^2}{\mu(1-\rho)^3} \tag{5.4}$$

We shall make use of this result in the next section.

5.4 Example: Inherently Stable Push-Pull Network

We are interested in the asymptotic variance rate of outputs of the push-pull network (described in Chapters 2 and 4). In Chapter 7 we use a diffusion limit to obtain it for both the inherently stable case and the inherently unstable case under general processing times. For illustration, we now derive the asymptotic variance of the outputs for a special case of the push-pull network using the renewal reward method described above.

Consider the push-pull network as described in Chapter 4 operating in the inherently stable case ($\lambda_1 < \mu_1$ and $\lambda_2 < \mu_2$) under a pull-priority preemptive policy. Assume that the processing times of the push operations are exponentially distributed and the processing times of the pull operations have some general distribution functions $H_1(\cdot)$ and $H_2(\cdot)$ with finite second moments and coefficients of variation c_1^2 and c_2^2.

As we described in previous chapters, the behavior of the network in this case is like two alternating single server queues. Also, since the push operations are exponential, every time the system empties is a regeneration epoch. We can thus employ the renewal reward method to evaluate the asymptotic variance rate of outputs. We shall do all calculations for the outputs of type 1, the calculation for output of type 2 is symmetric.

We now look at the sequence of times between successive returns to an empty system. Since the process is regenerative, we may analyze one such cycle. Denote the cycle time, X. Also, denote by ξ an indicator random variables that takes 1 when the first push operation to complete during a start of a cycle is of type 1 and takes 0 otherwise. Denote by $\tilde{I}$, the duration of the period until the first push operation (this is similar to the idle-period). Denote by B_1 the duration of a busy period of type 1 and B_2 the duration of a busy period of type 2. Then:

$$X =^d \tilde{I} + \xi B_1 + (1-\xi)B_2, \qquad X^2 =^d \tilde{I}^2 + \xi(B_1^2 + 2\tilde{I}B_1) + (1-\xi)(B_2^2 + 2\tilde{I}B_2),$$

where the above equalities are in distribution. The reward, Y and its square, equals:

$$Y = \xi N_1, \qquad Y^2 = \xi^2 N_1^2$$

where N_1 is distributed as the number of jobs of type 1 served during a busy period. Also,

$$XY =^d \xi(N_1\tilde{I} + N_1 B_1).$$

Now ξ is independent of N_1 and $\tilde{I}$. And N_1 is independent of $\tilde{I}$. We also have,

$$\mathbb{E}\left[\xi\right] = \frac{\lambda_1}{\lambda_1+\lambda_2}, \qquad \mathbb{E}\left[\tilde{I}\right] = \frac{1}{\lambda_1+\lambda_2}, \qquad \mathbb{E}\left[\tilde{I}^2\right] = \frac{2}{(\lambda_1+\lambda_2)^2}. \tag{5.5}$$

Now the ingrediants of (5.2), (5.4) and (5.5) are ready to plug into:

$$\begin{aligned} n_{11} &= \mathbb{E}[\xi](\mathbb{E}[N_1]\mathbb{E}[\tilde{I}] + \mathbb{E}[N_1 B_1]), \\ x_1 &= \mathbb{E}[\tilde{I}] + \mathbb{E}[\xi]\mathbb{E}[B_1] + (1 - \mathbb{E}[\xi])\mathbb{E}[B_2], \\ x_2 &= \mathbb{E}[\tilde{I}^2] + \mathbb{E}[\xi](\mathbb{E}[B_1^2] + 2\mathbb{E}[\tilde{I}]\mathbb{E}[B_1]) + (1 - \mathbb{E}[\xi])(\mathbb{E}[B_2^2] + 2\mathbb{E}[\tilde{I}]\mathbb{E}[B_2]), \\ y_1 &= \mathbb{E}[\xi]\mathbb{E}[N_1], \\ y_2 &= \mathbb{E}[\xi]\mathbb{E}[N_1^2]. \end{aligned}$$

The above is substituted into (5.1), and after considerable simplification we obtain the asymptotic variance rate of type 1 outputs:

$$\overline{V}_1 = \frac{\lambda_1\mu_1}{(\mu_1\mu_2 - \lambda_1\lambda_2)^3}\Big(\lambda_1\lambda_2\mu_1\mu_2(1 + c_2^2)(\mu_1 - \lambda_1) + (\lambda_1^2\lambda_2^2c_1^2 + \mu_1^2\mu_2^2)(\mu_2 - \lambda_2)\Big). \tag{5.6}$$

The expression for $\overline{V}_2$, the asymptotic variance rate of outputs of type 2 is symmetric. As stated previously, this result will be generalized in Chapter 7.

Discussion

For illustration let us evaluate (5.6) for the symmetric exponential case with unit service time of the pull activities: $\lambda_1 = \lambda_2 = \lambda$, $\mu_1 = \mu_2 = 1$ and $c_1^2 = c_2^2 = 1$:

$$\overline{V}_1 = \frac{\lambda}{\lambda + 1}\left(\frac{1 + \frac{1}{\lambda^2}}{1 - \frac{1}{\lambda^2}}\right)^2$$

In this case, the output rate as specified in Chapter 2 or Chapter 4 is:

$$\nu_1 = \frac{\lambda}{\lambda + 1}.$$

So the limiting index of dispersion of counts ($\overline{V}_1/\nu_1$) grows to infinity as λ increases to 1 (μ). It is simple to observe that the push-pull network is congested when $\lambda \approx \mu$ (the reader should keep in mind that this is not the typical congestion level that is associated with high utilization as in standard queueing systems). Thus the outputs of the push-pull network become more variable (in the sense of limiting index of dispersion of counts) as the system becomes more congested. This behavior is different from a GI/G/1 queue in which the limiting index of dispersion of counts is constant for any congestion level. And is also different from the BRAVO phenomena that we detail in the next Chapter.

The above observations tempt us to consider the following question: *Find control policies for the push-pull network that minimize the limiting index of dispersion of counts while maintaining full utilization and stable queues.* Indeed it seems possible that one can "improve" the pull-priority policy that we have just analyzed in that respect. To our surprise, the diffusion limit result that we present in Chapter 7 has shown us that the asymptotic variance rate of outputs (or limiting index of dispersion of counts) is insensitive to the scheduling policy.

CHAPTER 6

ASYMPTOTIC VARIANCE RATE OF FINITE QUEUE OUTPUTS

In this chapter we analyze the asymptotic variance rate of outputs of some of the most fundamental queueing systems: Finite capacity birth-death queues. In general, output processes of one-pass single class systems and their second moments have been studied extensively, cf. the surveys Reynolds (1975); Daley (1976); Disney and Konig (1985). For finite state space loss systems, the overflow process has received a considerable amount of attention, cf. Cinlar and Disney (1967); van Doorn (1984); Branford (1986); Pourbabai (1987); Berger and Whitt (1992); Whitt (2004); Parthasarathy and Sudhesh (2005). Fewer papers have considered the output process of loss systems, cf. Disney and de Morais (1976); Barnes and Disney (1990); Neuts and Li (2000)) and to the best of our knowledge none have analyzed the asymptotic variance rate of the outputs.

This chapter is organized as follows: Section 6.1 is an introduction. In Section 6.2 we present some chapter specific notation and fundamental results that are used throughout. In section 6.3 we state and prove the main theorem of this Chapter. In section 6.4 we analyze the M/M/1/K queue. In Section 6.5 we further discuss the BRAVO effect. The contents of this chapter was published in Nazarathy and Weiss (2008a).

6.1 Introduction

Let $\mathcal{Q} = \{Q(t),\ t \geq 0\}$ be the number of jobs in a queueing system and assume that it is an irreducible, stationary continuous time Markov chain (CTMC) with a birth-death structure on the finite state space $\{0, \ldots, K\}$. Let $\mathcal{D} = \{D(t),\ t \geq 0\}$ be the *output process* associated with the queue: $D(0) = 0$ and $\mathcal{D}$ increases by 1 when $\mathcal{Q}$ decreases. It can be shown that the expectation and the variance functions of $\mathcal{D}$ are $O(t)$ (cf. formulas (6.9, 6.10) and accompanying discussion and references) and may thus be described by the *flow rate*, λ^* and *asymptotic variance rate*, $\bar{V}_{\mathcal{D}}$:

$$\mathbb{E}\left[D(t)\right] = \lambda^* t \tag{6.1}$$

$$\text{Var}(D(t)) = \bar{V}_{\mathcal{D}} t + o(t) \tag{6.2}$$

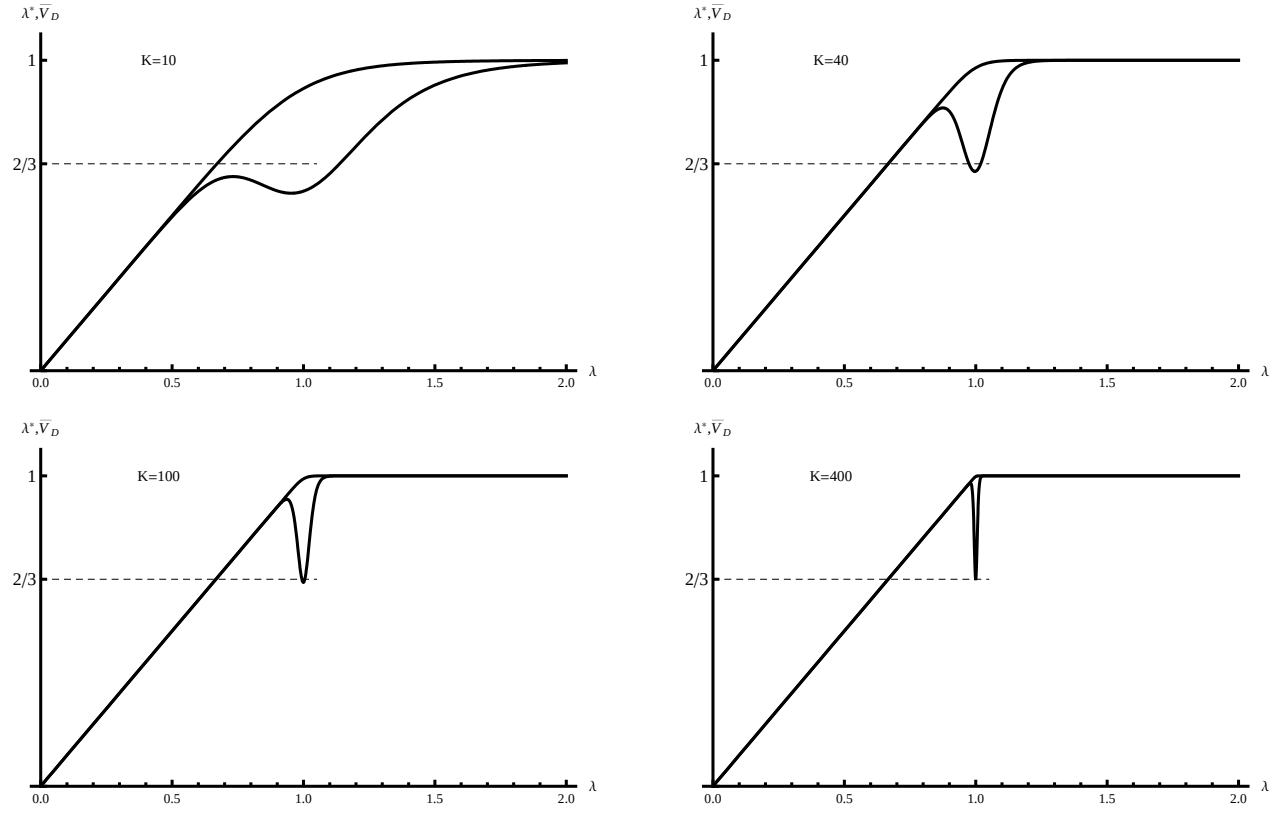

Figure 6.1: M/M/1/K: λ^* (top curve) and $\bar{V}_{\mathcal{D}}$ (bottom curve) as a function of λ when $\mu = 1$ for various buffer sizes.

Figures 6.1 and 6.2 display $\bar{V}_{\mathcal{D}}$ for different parameter values of the M/M/1/K queue with arrival rate λ and service rate μ. The plots may be partially understood as follows: For $\lambda \ll \mu$ the finite queue is hardly ever full and it behaves almost like an M/M/1 queue. In the M/M/1 queue, reversibility arguments imply that $\mathcal{D}$ is a Poisson process (cf. Kelly (1979)), and thus for M/M/1/K we expect $\bar{V}_{\mathcal{D}} \approx \lambda^* \approx \lambda$ when $\lambda \ll \mu$. For $\lambda \gg \mu$ the queue is almost always full and thus the outputs are similar to a Poisson process with rate μ so we expect $\bar{V}_{\mathcal{D}} \approx \lambda^* \approx \mu$ when $\lambda \gg \mu$. The behavior of the plots of $\bar{V}_{\mathcal{D}}$ when $\lambda \approx \mu$ is not easily explained: There is a pronounced decrease to a value of approximately $\frac{2}{3}\lambda$. To the best of our knowledge, this phenomenon has not been documented previously. We loosely refer to this as the **BRAVO** effect which stands for: **B**alancing **R**educes **A**symptotic **V**ariance of **O**utputs. Our results show that BRAVO occurs in a variety of finite capacity queueing models with losses.

Still focusing on the M/M/1/K queue as an example, notice that while it can be shown that the process which is the sum of the outputs and the overflows is Poisson, $\mathcal{D}$ by itself is not Poisson. One may attempt to evaluate the asymptotic variance rate by treating $\mathcal{D}$ as a renewal process. In this case $\bar{V}_{\mathcal{D}} = c^2\lambda^*$ where $c^2 = \frac{\sigma^2}{m^2}$ denotes the squared coefficient of variation (SCV) of the stationary inter-output time having expectation m and variance σ^2 (cf. Asmussen (2003), pp. 161). Variations of this method have been used to approximate inter-node flows in queueing networks (cf. Whitt (1982), Whitt (1983b) and references therein, or our review in Chapter 1). But it is known that the output process of most finite buffer queueing systems is not a renewal process (cf. Disney and Konig (1985), Section VII) and thus there is no theoretical

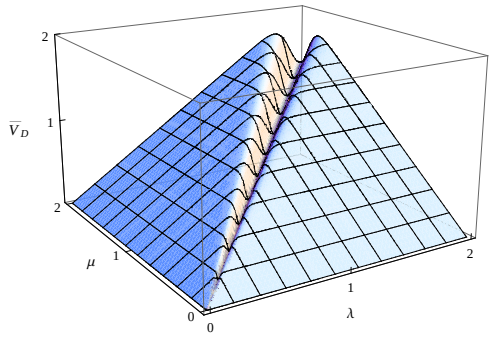

Figure 6.2: M/M/1/40: $\bar{V}_{\mathcal{D}}$ as a function of λ and μ.

justification for approximating $\bar{V}_{\mathcal{D}}$ using a renewal process. In fact, this type of approximation may yield completely incorrect results when the service rate and arrival rate are similar. For example, in the M/M/1/K queue case, the renewal approximation yields $\frac{\bar{V}_{\mathcal{D}}}{\lambda^*} = 1$ for $\lambda = \mu$, while the actual value is nearly $\frac{2}{3}$.

The probability law of $\mathcal{D}$ has been thoroughly researched. It is a Markov Renewal Process and also a Markovian Arrival Process (MAP) (cf. Asmussen (2003) and Disney and Kiessler (1987)). It is possible numerically to compute $\bar{V}_{\mathcal{D}}$, and even Var$(D(t))$ for any t, using well established matrix analytic results (see formulas (6.10) and (6.11) and references Naryana and Neuts (1992) and Neuts and Li (2000)). An alternative method for calculation is by the renewal reward approach that we described in Chapter 5. Thus, discovery of BRAVO did not require any new machinery.

Our results that we present in this Chapter are as follows: Part (i) of our main theorem (Theorem 6.1) is the formula $\bar{V}_{\mathcal{D}} = \lambda^* + \sum_{i=0}^{K-1} v_i$, where $v_i,\ i = 0, \ldots, K-1$ are expressions based on the birth and death rates. When applied to the M/M/1/K queue this formula yields a simple closed form expression. Part (ii) shows that when the birth rates are non-increasing and the death rates are non-decreasing (as is the case in many queueing systems), $v_i < 0$ for $i = 0, \ldots, K-1$ and hence, $\frac{\bar{V}_{\mathcal{D}}}{\lambda^*}$, the *limiting index of dispersion of counts* (cf. Cox and Isham (1980)) is less than unity. For the M/M/1/K queue we also derive additional results: an expression for the asymptotic correlation between the output and overflow processes and an expression for the y-intercept of the linear asymptote of Var$(D(t))$ for the balanced case.

The proof of Part (i) of our main theorem relies on a complementary result (Proposition 6.2) which relates to a class of MAPs that count every transition of a CTMC. We show that such MAPs have an associated Markov Modulated Poisson Process (MMPP) which has the same expectation and variance functions as the original MAP. This result may be of independent interest.

6.2 Preliminaries

We now introduce further notation and preliminary results that will be used.

Birth-Death CTMCs: We assume throughout that $\mathcal{Q}$ is a finite state space stationary birth-death process with generator matrix:

$$\mathbf{\Lambda} = \begin{pmatrix} -\lambda_0 & \lambda_0 & & & 0 \\ \mu_1 & -(\mu_1+\lambda_1) & \lambda_1 & & \\ & \ddots & \ddots & \ddots & \\ & & \mu_{K-1} & -(\mu_{K-1}+\lambda_{K-1}) & \lambda_{K-1} \\ 0 & & & \mu_K & -\mu_K \end{pmatrix} \tag{6.3}$$

The birth rates are $\lambda_0, \ldots, \lambda_{K-1} > 0$ and the death rates are $\mu_1, \ldots, \mu_K > 0$. The stationary probability distribution $\boldsymbol{\pi} = \{\pi_i,\ i = 0, \ldots, K\}$ is the solution of the equations: $\boldsymbol{\pi}\mathbf{\Lambda} = \mathbf{0}$, $\boldsymbol{\pi}\mathbf{1} = 1$, where we take $\boldsymbol{\pi}$ to be a row vector, $\mathbf{0}$ to be a row vector of 0s and $\mathbf{1}$ to be column vectors of 1s. It is well known that the stationary distribution is:

$$\pi_i = \frac{\lambda_0 \cdot \ldots \cdot \lambda_{i-1}}{\mu_1 \cdot \ldots \cdot \mu_i}\pi_0, \quad \text{where } \pi_0 \text{ is such that } \boldsymbol{\pi} \text{ sums to 1.} \tag{6.4}$$

We also have that the flow rate is:

$$\lambda^* = \sum_{i=0}^{K-1} \pi_i \lambda_i = \sum_{i=1}^{K} \pi_i \mu_i \tag{6.5}$$

We shall also be interested in systems for which $\lambda_0 \geq \ldots \geq \lambda_{K-1}$ and $\mu_1 \leq \ldots \leq \mu_K$. Examples include M/M/c/K queue where service effort is increased when more customers are present, as well as systems where queue build up discourages arrivals.

Traffic Processes: In addition to the output process $\mathcal{D}$, we shall also be interested in the following counting processes: Let $\mathcal{A} = \{A(t),\ t \geq 0\}$ count arrivals, $\mathcal{E} = \{E(t),\ t \geq 0\}$ count entrances (admissions) and $\mathcal{L} = \{L(t),\ t \geq 0\}$ count overflows (jobs that arrive to a full system and are thus immediately lost). Immediate relations are:

$$A(t) = E(t) + L(t) \tag{6.6}$$

$$E(t) = Q(t) + D(t) \tag{6.7}$$

We shall also make use of the process $\mathcal{M} = \{M(t),\ t \geq 0\}$ defined as follows:

$$M(t) := E(t) + D(t) \tag{6.8}$$

$\mathcal{M}$ counts the number of transitions in the birth-death state space. The asymptotic variance rates of the processes $\mathcal{Q},\ \mathcal{A},\ \mathcal{E},\ \mathcal{L}$ and $\mathcal{M}$ are defined similarly to $\bar{V}_{\mathcal{D}}$ (see (6.2)) and are labeled $\bar{V}_{\mathcal{Q}},\ \bar{V}_{\mathcal{A}},\ \bar{V}_{\mathcal{E}},\ \bar{V}_{\mathcal{L}}$ and $\bar{V}_{\mathcal{M}}$ respectively. Note that when $\mathcal{A}$ is Poisson with rate λ, $\bar{V}_{\mathcal{A}} = \lambda$, and that $\bar{V}_{\mathcal{Q}} = 0$ because $0 \leq Q(t) \leq K$.

The following lemma is a version of Theorem 5.1 from the previous chapter for finite capacity queues. It shows that analysis of the entrances, outputs or transitions in terms of the asymptotic variance rate is equivalent:

Lemma 6.1. $\bar{V}_{\mathcal{E}} = \bar{V}_{\mathcal{D}} = \frac{1}{4}\bar{V}_{\mathcal{M}}$

Proof. Using (6.7) we have, $\bar{V}_{\mathcal{E}} = \bar{V}_{\mathcal{Q}} + \bar{V}_{\mathcal{D}} + 2\overline{\mathrm{Cov}}_{\mathcal{Q},\mathcal{D}}$ where,

$$\overline{\mathrm{Cov}}_{\mathcal{Q},\mathcal{D}} := \lim_{t\to\infty} \frac{\mathrm{Cov}(Q(t), D(t))}{t},$$

is the asymptotic covariance rate of the pair ($\mathcal{Q}$,$\mathcal{D}$). Using (6.8) and (6.7) we have $M(t) = Q(t) + 2D(t)$ and thus

$$\bar{V}_{\mathcal{M}} = \bar{V}_{\mathcal{Q}} + 4\bar{V}_{\mathcal{D}} + 4\overline{\mathrm{Cov}}_{\mathcal{Q},\mathcal{D}}$$

The result follows since $\overline{\mathrm{Cov}}_{\mathcal{Q},\mathcal{D}}$ and $\bar{V}_{\mathcal{Q}}$ are 0. To show that $\overline{\mathrm{Cov}}_{\mathcal{Q},\mathcal{D}} = 0$ we note:

$$\left|\frac{\mathrm{Cov}(Q(t), D(t))}{\sqrt{\mathrm{Var}(Q(t))(\bar{V}_{\mathcal{D}}t + o(t))}}\right| \leq 1$$

which implies that $\mathrm{Cov}(Q(t), D(t)) = O(\sqrt{t})$, and hence $\overline{\mathrm{Cov}}_{\mathcal{Q},\mathcal{D}} = 0$. □

MAPs: We now briefly review Markov Arrival Processes (MAPs) and define the specific MAPs that are used throughout this chapter. A brief description of MAPs is in Asmussen (2003), Chapter XI, Section 1a, more examples, results and applications are in Breuer and Baum (2005) and Latouche and Ramaswami (1999). A MAP, $\mathcal{N} = \{N(t),\ t \geq 0\}$, is a counting process specified by a generator matrix, $\mathbf{Q}$, of a finite irreducible CTMC on the states $\{0, \ldots, K\}$ with stationary distribution $\boldsymbol{\eta}$ (row vector), and two matrices, $\mathbf{C}$, $\mathbf{D}$ such that $\mathbf{Q} = \mathbf{C} + \mathbf{D}$. $\mathbf{C}$ has negative diagonal elements and non-negative off-diagonal elements. $\mathbf{D}$ is a non-negative matrix. We choose to refer to $\mathbf{D}$ by the name: *event intensity matrix* [1].

$\mathcal{N}$ evolves as follows (loosely stated): When a CTMC (with generator $\mathbf{Q}$) makes a transition from state i to state j at time t, $N(t)$ is incremented w.p d_{ij}/q_{ij}. Further, during time intervals at which the CTMC is in state i, $N(t)$ is incremented by a Poisson process with state dependent rate: d_{ii}. Thus the non-diagonal elements of $\mathbf{D}$ specify the proportion of transitions that are to be counted and the diagonal elements, allow to increase $\mathcal{N}$ by a Poisson process that is modulated by the state of the CTMC.

We assume that $\mathcal{N}$ has stationary increments, which occurs when the initial distribution of the underlying CTMC, with the generator $\mathbf{Q}$, is $\boldsymbol{\eta}$. The following results are summarized in Asmussen (2003):

$$\mathbb{E}\left[N(t)\right] = \boldsymbol{\eta}\mathbf{D}\mathbf{1}\, t \tag{6.9}$$

$$\mathrm{Var}(N(t)) = \{\boldsymbol{\eta}\mathbf{D}\mathbf{1} - 2(\boldsymbol{\eta}\mathbf{D}\mathbf{1})^2 - 2\boldsymbol{\eta}\mathbf{D}\mathbf{Q}^{-}\mathbf{D}\mathbf{1}\}t + 2\boldsymbol{\eta}\mathbf{D}\mathbf{Q}^{-}(e^{\mathbf{Q}t} - \mathbf{I})\mathbf{Q}^{-}\mathbf{D}\mathbf{1} \tag{6.10}$$

Where $\mathbf{Q}^{-} = (\mathbf{Q} - \mathbf{1}\boldsymbol{\eta})^{-1}$ and $\mathbf{I}$ is the identity matrix. We may express $\mathrm{Var}(N(t))$ without the matrix exponential as:

$$\mathrm{Var}(N(t)) = \bar{V}_{\mathcal{N}}\, t + \bar{B}_{\mathcal{N}} + O(t^{3r+2}e^{-bt})$$

for some integer r and $b > 0$ (cf. Asmussen (2003)). Here the asymptotic variance rate, $\bar{V}_{\mathcal{N}}$, and the y-intercept of the linear asymptote, $\bar{B}_{\mathcal{N}}$, are given by:

$$\bar{V}_{\mathcal{N}} = \boldsymbol{\eta}\mathbf{D}\mathbf{1} - 2(\boldsymbol{\eta}\mathbf{D}\mathbf{1})^2 - 2\boldsymbol{\eta}\mathbf{D}\mathbf{Q}^{-}\mathbf{D}\mathbf{1} \tag{6.11}$$

$$\bar{B}_{\mathcal{N}} = 2(\boldsymbol{\eta}\mathbf{D}\mathbf{1})^2 - 2\boldsymbol{\eta}\mathbf{D}\mathbf{Q}^{-}\mathbf{Q}^{-}\mathbf{D}\mathbf{1} \tag{6.12}$$

[1] Note that in other texts, the term "arrival" is generally used to refer to events because MAPs are often used to model arrival processes. Here we use "event" to avoid confusion.

Clearly $\mathcal{M}$ and $\mathcal{D}$ are MAPs. With the exception of the numerical results of section 6.5, all of the MAPs that we use have the birth-death generator matrix $\mathbf{\Lambda}$ as in (6.3). This implies that the event intensity matrix is all that is required to specify a MAP. The event intensity matrices for $\mathcal{D}$ and $\mathcal{M}$ are:

$$\mathbf{D}_{\mathcal{D}} = \begin{pmatrix} 0 & 0 & & & 0 \\ \mu_1 & \ddots & 0 & & \\ & \mu_2 & \ddots & \ddots & \\ & & \ddots & \ddots & 0 \\ 0 & & & \mu_K & 0 \end{pmatrix} \tag{6.13}$$

$$\mathbf{D}_{\mathcal{M}} = \begin{pmatrix} 0 & \lambda_0 & & & 0 \\ \mu_1 & \ddots & \lambda_1 & & \\ & \mu_2 & \ddots & \ddots & \\ & & \ddots & \ddots & \lambda_{K-1} \\ 0 & & & \mu_K & 0 \end{pmatrix} \tag{6.14}$$

It is easily verified that $\mathbb{E}\left[D(t)\right] = \lambda^* t$ and $\mathbb{E}\left[M(t)\right] = 2\lambda^* t$.

Fully Counting MAPs and Markov Modulated Poisson Processes: We define *Fully Counting MAPs* as MAPs for which the event intensity matrix consists of all the off diagonal elements of the generator Q, i.e. $\mathbf{D} = \mathbf{Q} - \text{diag}(\mathbf{Q})$, where $\text{diag}(\mathbf{Q})$ is a diagonal matrix with the same diagonal as $\mathbf{Q}$. In a fully counting MAP, all the events are state transitions of the underlying CTMC and every state transition of the underlying CTMC is an event, so that $N(t)$ is the number of all the transitions of the underlying stationary CTMC with generator $\mathbf{Q}$, over the period $[0, t]$. Note that $\mathcal{M}$ is a fully counting MAP but $\mathcal{D}$ is not.

When the event intensity matrix $\mathbf{D}$ of a MAP is a diagonal matrix then the MAP is a Markov Modulated Poisson Process (MMPP). All the events of a MMPP are generated by a doubly stochastic Poisson process whose rate is a function of the state of the underlying CTMC. A comprehensive reference about MMPPs is Fischer and Meier-Hellstern (1992).

Fully counting MAPs and MMPPs are in a sense the extreme cases of MAPs. In a MMPP, the events do not coincide with state transitions (with probability 1). In contrast, in a fully counting MAP the events are precisely all the transitions of the CTMC. An early reference that analyzes both fully counting MAPs and MMPPs is Rudemo (1973).

Birth-Death MMPPs: Let $\tilde{\mathcal{N}}$ be a MMPP, with generator $\mathbf{Q}$ having stationary distribution $\boldsymbol{\eta}$. Denote the i'th diagonal element of $\mathbf{D}$ by $r(i)$ or r_i. This is the rate of events given that the CTMC is in state i. In example 9.6.2 of Whitt (2002), Whitt shows:

$$\bar{V}_{\tilde{\mathcal{N}}} = \sum_{i=0}^{K} r_i \eta_i + \bar{V}_{\mathcal{R}} \tag{6.15}$$

Here the asymptotic variance rate of the MMPP $\tilde{\mathcal{N}}$, $\bar{V}_{\tilde{\mathcal{N}}}$, is decomposed into two parts, where the first part is the average of the Poisson rate and the second part, $\bar{V}_{\mathcal{R}}$, is the asymptotic

variance rate of the integrated rate process, $R(t) = \int_0^t r(Q(s))ds$. The Internet supplement of Whitt (2002) shows how $\bar{V}_{\mathcal{R}}$ may be found from Poisson's equation for the CTMC (Theorem 2.3.4 of the Internet supplement). In general this requires solving a system of linear equations (numerically), but when $\mathbf{Q}$ is birth-death, the following result holds (cf. Whitt (1992), formula (6)):

$$\bar{V}_{\mathcal{R}} = 2\sum_{i=0}^{K-1} \frac{1}{\eta_i \lambda_i} [\sum_{j=0}^{i} (r_j - \sum_{l=0}^{K} r_l \eta_l)\eta_j]^2 \tag{6.16}$$

We summarize (6.15) and (6.16) of Whitt as a proposition. It is one of the ingredients that yield the main result of this chapter:

Proposition 6.1. *Let $\tilde{\mathcal{N}}$ be a MMPP with a birth-death generator matrix $\mathbf{Q}$ having birth rates λ_i, $i = 0, \ldots, K-1$ and stationary distribution η_i, $i = 0, \ldots, K$. Denote the diagonal elements of the event intensity matrix of $\tilde{\mathcal{N}}$ by r_i, $i = 0, \ldots, K$. Then the asymptotic variance of $\tilde{\mathcal{N}}$ is:*

$$\bar{V}_{\tilde{\mathcal{N}}} = \sum_{l=0}^{K} r_l \eta_l + 2\sum_{i=0}^{K-1} \frac{1}{\eta_i \lambda_i} [\sum_{j=0}^{i} (r_j - \sum_{l=0}^{K} r_l \eta_l)\eta_j]^2.$$

6.3 Asymptotic Variance Rate of Birth-Death Queues

We now consider a birth and death queue with generator $\mathbf{\Lambda}$, stationary distribution $\boldsymbol{\pi}$ and flow rate λ^*, as in (6.3–6.5). We introduce the following notations, for $i = 0, \ldots, K-1$:

$d_i := \lambda_i \pi_i$.

$D_i := \sum_{j=0}^{i} d_j$ (Note that $D_{K-1} = \lambda^*$).

$P_i := \sum_{j=0}^{i} \pi_j$.

$M_i := D_{i-1} - \lambda^* P_i$ (where we let $D_{-1} := 0$).

$v_i := 2(M_i + \frac{M_i^2}{d_i})$.

Note that by the detailed balance equations, $d_i = \mu_{i+1}\pi_{i+1}$. Thus $D_{i-1} = \sum_{j=1}^{i} \mu_j \pi_j$, and hence M_i measures the difference between the actual rate of outputs observed on the states $\{0, 1, \ldots, i\}$ and the rate of outputs that would have been observed if the output rate on these states was uniformly equal to the flow rate, λ^*, independent of the state. Our main result is:

Theorem 6.1. *Let $\mathcal{Q}$ be a stationary CTMC with a birth-death structure as defined in (6.3–6.5).*

(i)

$$\bar{V}_{\mathcal{D}} = \lambda^* + \sum_{i=0}^{K-1} v_i$$

(ii) If the birth and death rates of $\mathcal{Q}$ satisfy $\lambda_0 \geq \ldots \geq \lambda_{K-1}$ and $\mu_1 \leq \ldots \leq \mu_K$, then $v_i < 0$ for $i = 0, \ldots, K-1$ and as a result $\frac{\bar{V}_{\mathcal{D}}}{\lambda^*} < 1$.

Example 6.1. *We may verify Theorem 6.1 for the M/M/1/1 queue (This example is also analyzed in Chandramohan* et al. *(1985)). This is a 2 state CTMC and it is the only M/M/1/K queue that has a renewal output process (cf. Disney and Kiessler (1987)). The distribution of the inter-output times is the convolution of an exponential rate λ and an exponential rate μ distribution. Thus the expectation is $m = \frac{1}{\lambda} + \frac{1}{\mu}$, the variance is $\sigma^2 = \frac{1}{\lambda^2} + \frac{1}{\mu^2}$ and since $\mathcal{D}$ is a renewal processes,*

$$\bar{V}_{\mathcal{D}} = \frac{\sigma^2}{m^3} = \frac{\lambda\mu(\lambda^2+\mu^2)}{(\lambda+\mu)^3}$$

Now, $P_0 = \pi_0 = \frac{\mu}{\lambda+\mu}$, $\lambda^* = d_0 = \frac{\lambda\mu}{\lambda+\mu}$, $M_0 = -\frac{\lambda\mu^2}{(\lambda+\mu)^2}$ and $v_0 = -2\frac{\lambda^2\mu^2}{(\lambda+\mu)^3}$ (notice it is negative). And we obtain $\lambda^* + v_0 = \frac{\sigma^2}{m^3}$.

To prove Theorem 6.1 we use the following result, which is also of independent interest.

Proposition 6.2. *Let $\mathbf{Q}$ be a generator of a finite state irreducible CTMC. For any $0 \le \alpha \le 1$ let $\mathcal{N}_\alpha = \{N_\alpha(t),\ t \ge 0\}$ be a stationary MAP with generator $\mathbf{Q}$ and event intensity matrix $\mathbf{D}_\alpha = \alpha\mathbf{Q} - diag(\mathbf{Q})$. Then $\mathbb{E}\left[N_\alpha(t)\right]$ and $Var(N_\alpha(t))$ are independent of α.*

Note that when $\alpha = 1$ we have a fully counting MAP and when $\alpha = 0$ we have a MMPP.

Proof. From equations (6.9) and (6.10) we see that $\mathbb{E}\left[N_\alpha(t)\right]$ and $\mathrm{Var}(N_\alpha(t))$ only depend on $\mathbf{D}_\alpha\mathbf{1}$ and $\boldsymbol{\eta}\mathbf{D}_\alpha$.

First observe that $\mathbf{D}_\alpha\mathbf{1}$ is independent of α: Denote the elements of $\mathbf{Q}$ by q_{ij}. $\mathbf{Q}$ is a generator matrix so $q_{ii} = -\sum_{j\neq i} q_{ij}$, thus i'th element of $\mathbf{D}_\alpha\mathbf{1}$ is:

$$\alpha\sum_{j\neq i} q_{ij} - (1-\alpha)q_{ii} = \sum_{j\neq i} q_{ij}$$

Next observe that $\boldsymbol{\eta}\mathbf{D}_\alpha$ is independent of α: Since $\boldsymbol{\eta}$ is the stationary distribution we have $\boldsymbol{\eta}\mathbf{Q} = \mathbf{0}$. Thus $\boldsymbol{\eta}\mathbf{D}_\alpha = \alpha\boldsymbol{\eta}\mathbf{Q} - \boldsymbol{\eta}\mathrm{diag}(\mathbf{Q}) = -\boldsymbol{\eta}\mathrm{diag}(\mathbf{Q})$. □

We are now ready to prove Theorem 6.1:

Proof of (i): Let $\bar{V}_{\tilde{\mathcal{M}}}$ be the asymptotic variance rate of the MAP (also MMPP) $\tilde{\mathcal{M}} = \{\tilde{M}(t),\ t \ge 0\}$ having the following event intensity matrix:

$$\mathbf{D}_{\tilde{\mathcal{M}}} = \begin{pmatrix} \lambda_0 & & & & & 0 \\ & \mu_1+\lambda_1 & & & & \\ & & \mu_2+\lambda_2 & & & \\ & & & \ddots & & \\ & & & & \mu_{K-1}+\lambda_{K-1} & \\ 0 & & & & & \mu_K \end{pmatrix} \tag{6.17}$$

Denote the diagonal elements of $\mathbf{D}_{\tilde{\mathcal{M}}}$ by $r_i,\ i = 0,\ldots,K$. We now have:

$$4\bar{V}_{\mathcal{D}} = \bar{V}_{\mathcal{M}} = \bar{V}_{\tilde{\mathcal{M}}} = \sum_{l=0}^{K} r_l\pi_l + 2\sum_{i=0}^{K-1}\frac{1}{\pi_i\lambda_i}[\sum_{j=0}^{i}(r_j - \sum_{l=0}^{K} r_l\pi_l)\pi_j]^2 \tag{6.18}$$

The first equality is from Lemma 6.1. The second equality is from Proposition 6.2 by taking $\alpha = 1$ for the fully counting MAP, $\mathcal{M}$ and $\alpha = 0$ for the MMPP, $\tilde{\mathcal{M}}$ (see (6.14) and (6.17)).

The third equality is from Proposition 6.1 since $\tilde{\mathcal{M}}$ is a Birth and Death MMPP. Now note that $\sum_{l=0}^{K} \pi_l r_l = 2\lambda^*$ and

$$\sum_{j=0}^{i}(r_j - \sum_{l=0}^{K} r_l \pi_l)\pi_j = \sum_{j=0}^{i} \lambda_j \pi_j + \sum_{j=1}^{i} \mu_j \pi_j - 2\lambda^* \sum_{j=0}^{i} \pi_i = d_i + 2(D_{i-1} - \lambda^* P_i) \tag{6.19}$$

The first equality follows from direct substitution of r_j and the second follows from the detailed balance equations $\mu_i \pi_i = \lambda_{i-1}\pi_{i-1}$ and simplification. Now using the definition of M_i and substituting (6.19) in (6.18) we get:

$$4\bar{V}_{\mathcal{D}} = 2\lambda^* + 2\sum_{i=0}^{K-1} \frac{d_i^2 + 4d_i M_i + 4M_i^2}{d_i}$$

Noticing that $\sum_{i=0}^{K-1} d_i = \lambda^*$, result (i) follows.

Proof of (ii): We use the following two simple inequalities:

(a) For $a, b, c, d > 0$

$$\frac{a}{b} < \frac{c}{d} \;\Leftrightarrow\; \frac{a}{b} < \frac{a+c}{b+d} \;\Leftrightarrow\; \frac{a+c}{b+d} < \frac{c}{d}$$

(b) For $a, b, c, d, \Delta > 0$

$$\frac{a}{b} \leq \frac{c}{d} \text{ and } a < b \;\Rightarrow\; \frac{a}{\Delta + b} < \frac{c}{\Delta + d}$$

From $\lambda_0 \geq \lambda_1 \geq \cdots \geq \lambda_{K-1}$ we get:

$$\frac{d_0}{\pi_0} \geq \frac{d_1}{\pi_1} \geq \cdots \geq \frac{d_{K-1}}{\pi_{K-1}}$$

and therefore using (a):

$$\frac{D_0}{P_0} \geq \frac{D_1}{P_1} \geq \cdots \geq \frac{D_{K-1}}{P_{K-1}} > D_{K-1} = \lambda^*$$

From $\mu_1 \leq \mu_2 \leq \cdots \leq \mu_K$ we get:

$$\frac{d_0}{\pi_1} \leq \frac{d_1}{\pi_2} \leq \cdots \leq \frac{d_{K-1}}{\pi_K}$$

and therefore using (b):

$$\frac{D_0}{P_1 - \pi_0} \leq \frac{D_1}{P_2 - \pi_0} \leq \cdots \leq \frac{D_{K-1}}{P_K - \pi_0}$$

and furthermore, since $D_0 < D_1 < \cdots < D_{K-1}$ we also have:

$$0 < \frac{D_0}{P_1} < \frac{D_1}{P_2} < \cdots < \frac{D_{K-1}}{P_K} = \lambda^*.$$

Hence, for all $i = 0, \ldots, K-1$,

$$d_i + D_{i-1} - \lambda^* P_i > 0 > D_{i-1} - \lambda^* P_i,$$

which implies:

$$M_i < 0 \text{ and } d_i > |M_i|$$

from which it follows that:

$$v_i = 2(M_i + \frac{M_i^2}{d_i}) < 0$$

(to clarify: $d_i + M_i > 0 > M_i \Rightarrow d_i > -M_i > 0 \Rightarrow d_i|M_i| > M_i^2 \Rightarrow -M_i = |M_i| > \frac{M_i^2}{d_i} \Rightarrow 0 > M_i + \frac{M_i^2}{d_i}$). □

Note that the condition on the birth and death rates in Part (ii) implies that the sequence $\frac{\lambda_i}{\mu_{i+1}},\ i = 0, \ldots, K-1$ is non-increasing and as a result $\boldsymbol{\pi}$ is unimodal (cf. Keilson (1979)). This observation makes it tempting to attempt to generalize the theorem in this direction. The following example shows that this is not possible:

Example 6.2. *Let* $K = 2,\ \lambda_0 = \frac{1}{3},\ \mu_1 = \frac{1}{3},\ \lambda_1 = 1$ *and* $\mu_2 = \frac{3}{2}$. *The stationary distribution is* $(\pi_0 \quad \pi_1 \quad \pi_2) = (\frac{3}{8} \quad \frac{3}{8} \quad \frac{1}{4})$. *It is unimodal as expected because* $\frac{\lambda_i}{\mu_{i+1}}$ *is non-increasing but* $v_0 = \frac{3}{16},\ v_1 = -\frac{1}{6}$ *and* $\frac{\bar{V}_{\mathcal{D}}}{\lambda^*} = 1 + \frac{1}{24}$.

6.4 Traffic Processes of M/M/1/K

We now apply Theorem 6.1 to the case where the birth and death rates are constant, $\lambda,\ \mu > 0$. Denote $\rho = \frac{\lambda}{\mu}$. The stationary distribution and the flow rate are:

$$\begin{aligned} \pi_i &= \begin{cases} \frac{1}{K+1} & \rho = 1 \\ \rho^i \frac{1-\rho}{1-\rho^{K+1}} & \rho \neq 1 \end{cases} \quad i = 0, \ldots, K \\ \lambda^* &= \begin{cases} \lambda \frac{K}{K+1} & \rho = 1 \\ \lambda \frac{1-\rho^K}{1-\rho^{K+1}} & \rho \neq 1 \end{cases} \end{aligned} \tag{6.20}$$

Corollary 6.1. *For the M/M/1/K queue:*

$$\begin{aligned} \bar{V}_{\mathcal{D}} &= \begin{cases} \lambda \frac{2K^2+K}{3K^2+6K+3} & \rho = 1 \\ \lambda \frac{(1+\rho^{K+1})(1-(1+2K)\rho^K(1-\rho)-\rho^{2K+1})}{(1-\rho^{K+1})^3} & \rho \neq 1 \end{cases} \\ \frac{\bar{V}_{\mathcal{D}}}{\lambda^*} &= \begin{cases} \frac{2K+1}{3K+3} & \rho = 1 \\ \frac{(1+\rho^{K+1})(1-(1+2K)\rho^K(1-\rho)-\rho^{2K+1})}{(1-\rho^K)(1-\rho^{K+1})^2} & \rho \neq 1 \end{cases} \end{aligned} \tag{6.21}$$

Proof. Using straight forward (but lengthy) calculations we obtain:

$$\begin{aligned} M_i &= \begin{cases} -\lambda \frac{K-i}{(K+1)^2} & \rho = 1 \\ -\lambda \rho^i \frac{(1-\rho)(1-\rho^{K-i})}{(1-\rho^{K+1})^2} & \rho \neq 1 \end{cases} \quad i = 0, \ldots, K-1 \\ v_i &= \begin{cases} -\lambda 2 \frac{(i+1)(K-i)}{(K+1)^3} & \rho = 1 \\ -\lambda 2 \rho^K \frac{(1-\rho^{i+1})(1-\rho)(1-\rho^{K-i})}{(1-\rho^{K+1})^3} & \rho \neq 1 \end{cases} \quad i = 0, \ldots, K-1 \end{aligned}$$

The result follows from Theorem 6.1 after summation of finite geometric series and simplification.

□

The following properties of $\bar{V}_{\mathcal{D}}$ and $\frac{\bar{V}_{\mathcal{D}}}{\lambda^*}$ should be noted:

- For fixed K, $\bar{V}_{\mathcal{D}}$ and $\frac{\bar{V}_{\mathcal{D}}}{\lambda^*}$ are continuous in λ and μ for all $\lambda,\ \mu > 0$.

- For fixed $\lambda,\ \mu$ we have:

$$\lim_{K\to\infty} \bar{V}_{\mathcal{D}} = \begin{cases} \lambda & \lambda < \mu \\ \frac{2}{3}\lambda & \lambda = \mu \\ \mu & \lambda > \mu \end{cases}$$

- For fixed K and fixed $C > 0$, $\bar{V}_{\mathcal{D}}$ and $\frac{\bar{V}_{\mathcal{D}}}{\lambda^*}$ are symmetric about the point $\lambda = \mu$ on the interval $\{(\lambda, \mu) \mid \lambda + \mu = C,\ \lambda,\ \mu > 0\}$ (see also Figure 6.2).

The following corollary formalizes the BRAVO effect for M/M/1/K:

Corollary 6.2. *Consider the M/M/1/K queue with $\lambda + \mu = C$ for some $C > 0$. Then when $\lambda = \mu$: $\bar{V}_{\mathcal{D}}$ is locally minimized and $\frac{\bar{V}_{\mathcal{D}}}{\lambda^*}$ is globally minimized.*

Proof. Take derivatives and limits of the expressions of Corollary 6.1. □

Asymptotic Correlation Between Outputs and Overflows

It is well known and easy to observe that the overflow process, $\mathcal{L}$, is a renewal process. The overflow rate is of course $\lambda - \lambda^*$. Berger and Whitt, Berger and Whitt (1992) in their equation (6) derive the SCV for the inter-overflow times. Multiplying these we obtain the asymptotic variance rate of the overflows [2]:

$$\bar{V}_{\mathcal{L}} = \begin{cases} \lambda \frac{2K^2+4K+3}{3K^2+6K+3} & \rho = 1 \\ \lambda \frac{(\rho^K - \rho^{3K+2})(1+\rho) - 4(K+1)(1-\rho)\rho^{2K+1}}{(1-\rho^{K+1})^3} & \rho \neq 1 \end{cases} \tag{6.22}$$

In general, the covariance, asymptotic covariance rate, correlation and limiting correlation of pairs of traffic processes may be numerically calculated by modeling the queueing system as a Marked Markovian Arrival Process (MMAP) (cf. He and Neuts (1998)) and using formulas similar to (6.10) that appear in that reference. For the simple case of the M/M/1/1 queue, an explicit expression was obtained for the limiting correlation coefficient between the outputs and the overflows in Chandramohan *et al.* (1985). We now extend this result:

Corollary 6.3. *For the M/M/1/K queue:*

$$\lim_{t\to\infty} Corr(E(t), L(t)) = \lim_{t\to\infty} Corr(D(t), L(t)) = \begin{cases} \bar{R}_{\rho,K} & \rho < 1 \\ -\frac{1-\frac{1}{K}}{4\sqrt{1+\frac{5}{2K}+\frac{5}{2K^2}+\frac{3}{4K^3}}} & \rho = 1 \\ -\bar{R}_{\rho,K} & \rho > 1 \end{cases}$$

where:

$$\bar{R}_{\rho,K} = \frac{\rho^{\frac{K}{2}} K(1-\rho)(1+3\rho^{1+K}) - \rho(1-\rho^K)(3+\rho^{K+1})}{\sqrt{(1+\rho^{K+1})(1-(2K+1)(1-\rho)\rho^K - \rho^{2K+1}))((1+\rho)(1-\rho^{2K+2}) - 4(K+1)(1-\rho)\rho^{K+1}))}}$$

Proof. In a similar manner to the proof of Lemma 6.1, define the asymptotic covariance rates $\overline{\mathrm{Cov}}_{\mathcal{D},\mathcal{L}}$ and $\overline{\mathrm{Cov}}_{\mathcal{E},\mathcal{L}}$. We are assured that the covariance functions of these traffic process are $O(t)$ since $\mathrm{Var}(D(t))$, $\mathrm{Var}(E(t))$ and $\mathrm{Var}(L(t))$ are $O(t)$.

[2] An alternative derivation of (6.22) is by conditioning $L(t)$ on the occupation time of state K during $[0, t]$, and using the conditional variance formula. This calculation requires evaluation of the asymptotic variance rate of the occupation time using formula (6.16).

Take the variance of equation (6.6), divide by t, take the limit $t \to \infty$ and rearrange to arrive at:

$$\overline{\mathrm{Cov}}_{\mathcal{E},\mathcal{L}} = \frac{\lambda - \bar{V}_{\mathcal{E}} - \bar{V}_{\mathcal{L}}}{2}$$

In a similar manner (and using arguments similar to the proof of Lemma 6.1), obtain:

$$\overline{\mathrm{Cov}}_{\mathcal{D},\mathcal{L}} = \frac{\lambda - \bar{V}_{\mathcal{D}} - \bar{V}_{\mathcal{L}}}{2}$$

Thus from Lemma 6.1 and from substitution of (6.21, 6.22)

$$\overline{\mathrm{Cov}}_{\mathcal{E},\mathcal{L}} = \overline{\mathrm{Cov}}_{\mathcal{D},\mathcal{L}} = \begin{cases} -\lambda \frac{K^2-K}{6K^2+12K+6} & \rho = 1 \\ -\lambda \frac{(1-\rho^K)(3+\rho^{K+1})\rho^{K+1} - K(1-\rho)(1+3\rho^{K+1})\rho^K}{(1-\rho^{K+1})^3} & \rho \neq 1 \end{cases} \tag{6.23}$$

The correlation coefficient is obtained directly from (6.21, 6.22, 6.23) and simplification. □

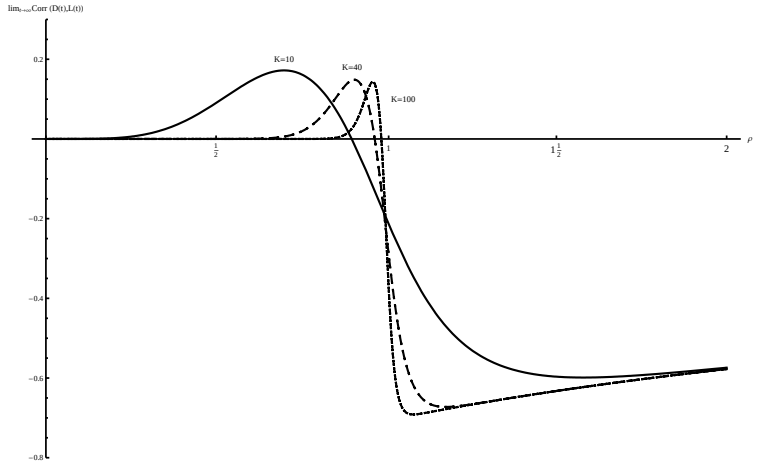

Figure 6.3: M/M/1/K: The limiting correlation between entrances/outputs and overflows as a function of ρ.

Figure 6.3 displays the limiting correlation coefficient for various buffer sizes. Note also the following properties:

- The limiting correlation is continuous in ρ for all $\rho > 0$.
- For fixed K, as $\rho \to \infty$, the limiting correlation increases to 0.
- For fixed ρ we have:

$$\lim_{K\to\infty} \lim_{t\to\infty} \mathrm{Corr}(D(t), L(t)) = \begin{cases} 0 & \rho < 1 \\ -\frac{1}{4} & \rho = 1 \\ -\frac{1}{\sqrt{1+\rho}} & \rho > 1 \end{cases}$$

- For finite K, let $\hat{\rho} := \arg\max_{0<\rho<1} \bar{R}_{\rho,K}$. Then $\hat{\rho}$ converges to 1 as $K \to \infty$, and it is numerically observed that the maximum value converges to $\lim_{K\to\infty} \bar{R}_{\hat{\rho},K} \approx 0.139772$.
- Similarly, let $\breve{\rho} := \arg\min_{\rho>1} -\bar{R}_{\rho,K}$. Then $\breve{\rho}$ converges to 1 as $K \to \infty$, and the minimum value converges to $\lim_{K\to\infty} -\bar{R}_{\breve{\rho},K} = -\frac{1}{\sqrt{2}}$.

- Summarizing, informally, we see that the limiting correlation attains 3 different values at the vicinity of $\rho = 1$ for large K:

$$\lim_{t\to\infty} \text{Corr}(D(t), L(t)) \approx \begin{cases} 0.139772 & \rho = 1^- \\ -\frac{1}{4} & \rho = 1 \\ -\frac{1}{\sqrt{2}} & \rho = 1^+ \end{cases}$$

The y-intercept of the Linear Asymptote of Var$(D(t))$

We now analyze the y-intercept, $\bar{B}_{\mathcal{D}}$ according to formula (6.12) (take $\mathbf{Q} = \mathbf{\Lambda}$, $\boldsymbol{\eta} = \boldsymbol{\pi}$ and $\mathbf{D} = \mathbf{D}_{\mathcal{D}}$). Figure 6.4 presents $\bar{B}_{\mathcal{D}}$ as a function of λ for $K = 10$ and $K = 20$. Interestingly, $\bar{B}_{\mathcal{D}}$ appears to be maximized when $\rho = 1$ and the value increases with K. Note that when $\rho \neq 1$ our calculations indicate that $\bar{B}_{\mathcal{D}}$ decreases to 0 as $K \to \infty$.

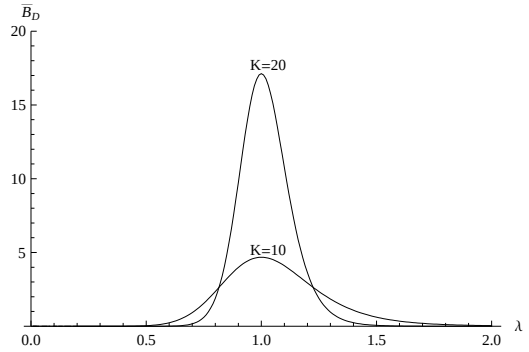

Figure 6.4: M/M/1/K: $\bar{B}_{\mathcal{D}}$ as a function λ when $\mu = 1$.

The values of $\bar{B}_{\mathcal{D}}$ in Figure 6.4 were evaluated numerically (each point on the curve requires inversion of $(\mathbf{\Lambda} - \mathbf{1}\boldsymbol{\pi})$ to obtain $\mathbf{\Lambda}^-$). In the balanced case the stationary distribution of the queue lengths (6.20) is discrete uniform, and we can obtain explicit expressions for the elements of $\mathbf{\Lambda}^-$ and in turn have an explicit expression for $\bar{B}_{\mathcal{D}}$. The following proposition derives this expression and shows that $\bar{B}_{\mathcal{D}}$ for $\rho = 1$ increases quadratically with K.

Proposition 6.3. *For the M/M/1/K queue with $\rho = 1$ and $K \geq 2$:*

$$\bar{B}_{\mathcal{D}} = \frac{7K^4 + 28K^3 + 37K^2 + 18K}{180K^2 + 360K + 180}$$

Proof. For $\gamma > 0$ and $K \geq 2$, define the $K \times K$ matrix:

$$\mathbf{A}_K^\gamma = \begin{pmatrix} 1+\gamma & 1-\gamma & & & & & & 1 \\ 1-\gamma & 1+2\gamma & \ddots & & & & & \\ & \ddots & 1+2\gamma & \ddots & & & & \\ & & \ddots & \ddots & \ddots & & & \\ & & & \ddots & 1+2\gamma & \ddots & & \\ & & & & \ddots & 1+2\gamma & 1-\gamma & \\ 1 & & & & & 1-\gamma & 1+\gamma & \end{pmatrix} \tag{6.24}$$

By Lemma 6.2, its inverse, $(\mathbf{A}_K^\gamma)^{-1}$, is also a symmetric matrix with elements given by:

$$\tilde{a}_{i,j} = \frac{i^2 - i + (K+1-j)^2 - (K+1-j)}{2K\gamma} - \frac{K^3 - K - 6\gamma}{6K^2\gamma} \qquad i \leq j \tag{6.25}$$

Now observe that $\mathbf{\Lambda}^- = -(K+1)(\mathbf{A}_{K+1}^{\lambda(K+1)})^{-1}$ and find the elements of $\mathbf{\Lambda}^-\mathbf{\Lambda}^-$ by multiplication. These are rather complicated expressions, we omit the details. We now have:

$$\begin{aligned}\bar{B}_{\mathcal{D}} &= 2\left(\lambda\frac{K}{K+1}\right)^2 - 2\frac{1}{K+1}\mathbf{1}'\mathbf{D}\mathbf{\Lambda}^-\mathbf{\Lambda}^-\mathbf{D}\mathbf{1} \\ &= 2\left(\lambda\frac{K}{K+1}\right)^2 - 2\frac{\lambda^2}{K+1}(1,\ldots,1,0)\mathbf{\Lambda}^-\mathbf{\Lambda}^-(0,1,\ldots,1)' \qquad (6.26)\end{aligned}$$

The bilinear form in the second term is a summation of all entries of the matrix $\mathbf{\Lambda}^-\mathbf{\Lambda}^-$ except for the first column and last row. The resulting expression is:

$$(1,\ldots,1,0)\mathbf{\Lambda}^-\mathbf{\Lambda}^-(0,1,\ldots,1)' = -\frac{7K^4+28K^3-(360\lambda^2-37)K^2+18K}{360\lambda^2(K+1)}$$

Plugging this in (6.26) and simplifying we obtain the result. □

Lemma 6.2. *The elements of the inverse of the matrix (6.24) are as in (6.25).*

Proof. Examine the matrix multiplication $\mathbf{R}_K^\gamma {\mathbf{R}_K^\gamma}^{-1}$. Denote the entries of the resulting matrix by $\tilde{i}_{i,j}$. Since both matrices are symmetric it is enough to verify that $\tilde{i}_{i,j}$ are elements of the identity matrix for $i \le j$. We split our calculation into five cases and utilize the identity $\sum_{l=1}^K \tilde{r}_{l,j} = \frac{1}{K}$ [3]:

$i = j = 1$:

$$\begin{aligned}\tilde{i}_{1,1} &= (1+\gamma)\tilde{r}_{1,1} + (1-\gamma)\tilde{r}_{2,1} + \sum_{l=3}^K \tilde{r}_{l,1} \\ &= \gamma(\tilde{r}_{1,1} - \tilde{r}_{1,2}) + \frac{1}{K} \;=\; 1\end{aligned}$$

$i = j = 2,\ldots,K-1$:

$$\begin{aligned}\tilde{i}_{i,i} &= \sum_{l=1}^{i-2}\tilde{r}_{l,i} + (1-\gamma)\tilde{r}_{i-1,i} + (1+2\gamma)\tilde{r}_{i,i} + (1-\gamma)\tilde{r}_{i+1,i} + \sum_{l=i+2}^K \tilde{r}_{l,i} \\ &= -\gamma(\tilde{r}_{i-1,i} - 2\tilde{r}_{i,i} + \tilde{r}_{i,i+1}) + \frac{1}{K} \;=\; 1\end{aligned}$$

$i = j = K$:

$$\begin{aligned}\tilde{i}_{K,K} &= \sum_{l=1}^{K-2}\tilde{r}_{l,K} + (1-\gamma)\tilde{r}_{K-1,K} + (1+\gamma)\tilde{r}_{K,K} \\ &= -\gamma(\tilde{r}_{K-1,K} - \tilde{r}_{K,K}) + \frac{1}{K} \;=\; 1\end{aligned}$$

$i = 1$ and $j = 2,\ldots,K$:

$$\begin{aligned}\tilde{i}_{1,j} &= (1+\gamma)\tilde{r}_{1,j} + (1-\gamma)\tilde{r}_{2,j} + 1\sum_{l=3}^K \tilde{r}_{l,j} \\ &= \gamma(\tilde{r}_{1,j} - \tilde{r}_{2,j}) + \frac{1}{K} \;=\; 0\end{aligned}$$

$i = 2,\ldots,K-1$

and $j = i+1,\ldots,K$:

[3] The evaluation of this identity as well as several other steps requires some tedious algebraic calculations involving finite sums.

$$\begin{aligned}\tilde{i}_{i,j} &= \sum_{l=1}^{i-2}\tilde{r}_{l,j} + (1-\gamma)\tilde{r}_{i-1,j} + (1+2\gamma)\tilde{r}_{i,j} + (1-\gamma)\tilde{r}_{i+1,j} + \sum_{l=i+2}^{K}\tilde{r}_{l,j} \\ &= -\gamma(\tilde{r}_{i-1,j} - 2\tilde{r}_{i,j} + \tilde{r}_{i+1,j}) + \frac{1}{K} \;=\; 0\end{aligned}$$

□

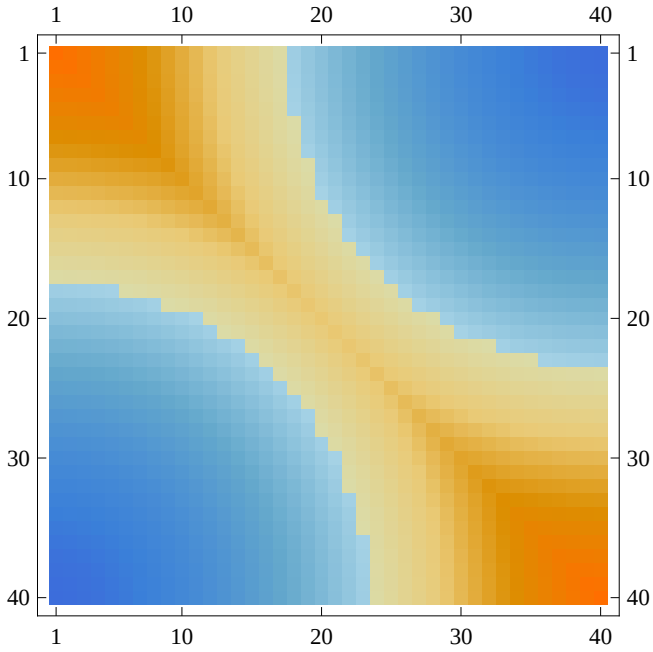

Figure 6.5: Matrix plot of the inverse matrix. $K = 40$. The blue pixels imply negative values and the brown pixels imply positive values.

The non-constructiveness of the above proof requires some comment: By inspecting examples (see Figure 6.5), we noticed that the differences between adjacent elements of ${\mathbf{R}_K^\gamma}^{-1}$ increase linearly by $\frac{1}{K\gamma}$ as the indexes get farther from the top right corner or bottom left corner (see Figure 6.5). More specifically:

$$\tilde{r}_{i,j} = \tilde{r}_{1,K} + (\sum_{l=1}^{K-j} l + \sum_{l=1}^{i-1} l)\frac{1}{K\gamma} \qquad i \le j \tag{6.27}$$

In addition we observe that sum of all elements of ${\mathbf{R}_K^\gamma}^{-1}$ is 1 (this may also be observed by multiplying $(\mathbf{\Lambda} - \mathbf{1}\pi)(\mathbf{\Lambda} - \mathbf{1}\pi)^{-1}$ from the left with π and from the right with $\mathbf{1}$ and remembering that π is uniform). We are thus able to sum (6.27) on all elements, equate to 1 and solve for $\tilde{r}_{1,K}$ and obtain formula (6.25) which is quadratic in the distances of the indexes i, j from the top right corner of the matrix. Note also that $\mathbf{R}_K^\gamma$ is a simple case of a Toeplitz-plus-Hankel matrix. A formula of the generating function of such a matrix is given in Heining and Rost (1988).

Note also that the other results of this section for the case $\rho = 1$ may also be obtained by using Lemma 6.2 instead of using Theorem 6.1. [4]

[4] A third method to obtain these results (only for the case $\rho = 1$) is by conditioning on the occupation time and using the conditional variance formula.

$\mathbf{Var}(D(t))$ in the Short Range

We now present numerical examples and results of the variance function for finite t. While our main finding of this chapter is that balancing reduces Var$(D(t))$ in the long range (BRAVO), there is no guarantee that it has the same effect in the short range. In fact, Figure 6.4 hints that balanced systems may have a higher variance function then unbalanced systems in the short range since the y-intercept of their linear asymptote is higher.

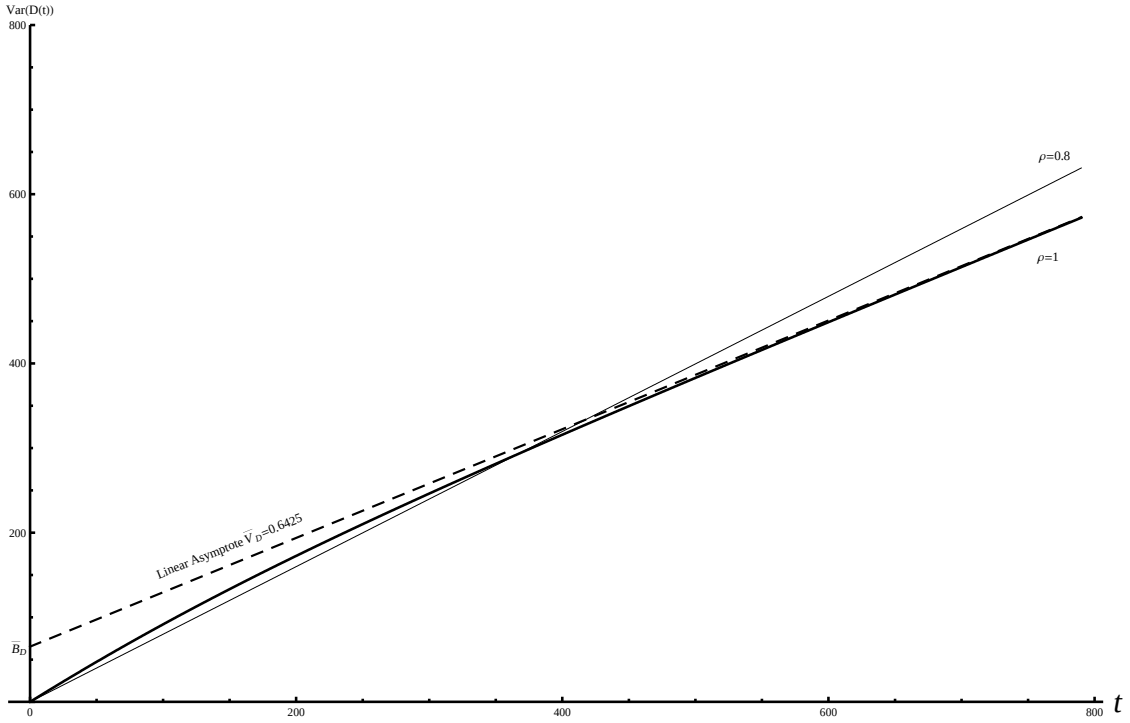

Figure 6.6: M/M/1/40: $Var(D(t))$ for $\mu = 1$ and two different arrival rates. Heavy curve is for $\lambda = 1$ (balanced). Light curve is for $\lambda = 0.8$ (unbalanced). Dashed line is linear asymptote of balanced case.

This is illustrated in Figure 6.6. Here we compare the variance function, Var$(D(t))$, of a balanced system to that of a system with $\rho = 0.8$. We plot the variance function (heavy curve) and its linear asymptote (dashed line) for the balanced system, and the variance function for the unbalanced system (light curve). Both are calculated for $K = 40$, using formula (6.10). It is observed that for the balanced system, the slope of the variance function is steeper than the asymptotic variance rate for small t and nears the asymptotic variance rate of approximately $\frac{2}{3}$ as time progresses. On the contrary the slope of the variance function of an unbalanced system almost equals the asymptotic variance (approximately 0.8, with negligible intercept) from the outset. As a result, the unbalanced system has a slightly lower variance function for values of t smaller than approximately 350.

To further understand the short-term behavior we performed extensive calculations in which we compared the variance of the output process of balanced M/M/1/K queues, $\mathcal{D}_1 = \{D_1(t),\, t \geq 0\}$, to that of unbalanced queues, with arrival rates λ and service rates $\mu = 2 - \lambda$, given by

$\mathcal{D}_\lambda = \{D_\lambda(t),\ t \geq 0\}$. We define:

$$\bar{T}_\lambda := \inf\{t > 0 \mid \text{Var}(D_1(t)) \leq \text{Var}(D_\lambda(t))\}$$

Stated informally, $\bar{T}_\lambda$ is a measure of the time it takes the BRAVO effect to "kick-in" when comparing a balanced system to an unbalanced one. We evaluated $\bar{T}_\lambda$ only for λ for which $\bar{V}_{\mathcal{D}_1} < \bar{V}_{\mathcal{D}_\lambda}$. It is infinite otherwise. The range of these λ's varies with K and as $K \to \infty$ the range converges to $(\frac{2}{3},\ \frac{4}{3})$. Figure 6.7 shows $\bar{T}_\lambda$ as a function of λ for $K = 10,\ 20,\ 30$. We observe the following:

- $0 < \bar{T}_\lambda < \infty$ for all $\lambda \neq 1$ that satisfy $\bar{V}_{\mathcal{D}_1} < \bar{V}_{\mathcal{D}_\lambda}$. The fact that $\bar{T}_\lambda \neq 0$ shows that indeed the unbalanced systems have a lower variance in the short range (during the time interval $[0, \bar{T}_\lambda)$).

- For fixed $\lambda \neq 1$, $\bar{T}_\lambda$ increases with K. In fact, for λ far enough from 1, a simple approximation for $\bar{T}_\lambda$ may be achieved by calculating the intersection of the linear asymptote of the balanced system (it is given by Corollary 6.1 and Proposition 6.3) and an approximation of the linear asymptote of an unbalanced system taking the y-intercept to be 0 and the asymptotic variance rate to be λ. According to this approximation, $\bar{T}_\lambda$ increases quadratically with K.

- For $\lambda = 1^-$ and $\lambda = 1^+$ we observe $\bar{T}_\lambda \approx K + 1$ and the approximation quickly becomes accurate when K increases. Note that $\bar{T}_1$ is trivially 0 and thus there is a singularity in the function $\bar{T}_\lambda$ at $\lambda = 1$. We do not have any intuitive explanation for the value of $K + 1$ at the moment.

6.5 More on BRAVO

For the M/M/1/K queue, our intuition for BRAVO is as follows: Since the asymptotic variance rate of the transitions $\mathcal{M}$ and the outputs $\mathcal{D}$ are the same up to a constant we can gain intuition by considering the transitions process. Now it can be seen that the rate of transitions incurred on states $\{1, \ldots, K-1\}$ is $\lambda + \mu$ while the rates of transitions on the edge states, 0 and K are λ and μ respectively. Observing the steady state distribution, (6.20), we see that when $\lambda = \mu$ the system spends very little time on the edge states and thus the "modulation" between rates $\lambda + \mu$ and λ or μ is minimal. On the contrary when $\lambda \neq \mu$ the system often switches between an edge state and a non-edge state and thus there is substantial "modulation" in the transition rates and as a result the variance of the transition process is greater.

This intuition does not immediately carry over to more complex systems but the BRAVO effect does. We now show some examples.

M/M/c/K

The M/M/c/K queue with $1 \leq c \leq K$ is an example a of a birth-death queue with monotone rates (the birth rates are constant and the death rates are increasing). While Theorem 6.1 is

applicable to this system, the calculation of the normalization constant of the stationary distribution does not simplify and thus we are not able to obtain simple a formula for $\bar{V}_{\mathcal{D}}$ except for the case $c = 1$ (Section 6.4). Nevertheless, the computation of $\bar{V}_{\mathcal{D}}$ using the formula of 6.1 is simpler and more efficient than using the matrix formula (6.11).

Figure 6.8 shows that the BRAVO effect appears in the M/M/c/K queue: in this case "balancing" implies setting $\lambda = c\mu$. The thick curve is for the Erlang loss system ($c = K$) with $K = 40$ to which we compare other systems. It is apparent that as the number of servers decreases, the asymptotic variance rate normalized by the number of servers increases. Alternatively, keeping the number of servers equal to the buffer size and decreasing the number of servers causes a decrease in the asymptotic variance rate normalized by the number of servers. We do not yet have an intuitive explanation for BRAVO in the M/M/c/K.

Non Exponential Distributions

We now consider some examples of GI/G/1/K using phase-type distributions (cf. Breuer and Baum (2005)). We let the inter-arrival and/or service time distributions be generated by sequences of i.i.d Erlang random variables, $\{E_1,\ E_2, \ldots\}$, and i.i.d hyper-exponential random variables, $\{H_1,\ H_2,\ \ldots\}$:

$$\begin{aligned} E_1 &\sim \text{Erlang}(2,2) \\ H_1 &\sim \begin{cases} \exp(\frac{1}{2}) & \text{w.p } 1/3 \\ \exp(2) & \text{w.p } 2/3 \end{cases} \end{aligned}$$

Note that: $\mathbb{E}[E_1] = \mathbb{E}[H_1] = 1$, the SCV of E_1 is $\frac{1}{2}$ and the SCV of H_1 is 2. We denote the queueing systems with the four possible combinations of inter-arrival and service distributions by: E/E/1/K, H/H/1/K, E/H/1/K and H/E/1/K. In all our examples we set $\mu = 1$ and scale the corresponding sequences of inter-arrival or service times by $\frac{1}{\lambda}$.

These are simple examples of PH/PH/1/K queues and are represented by a CTMC with $2 + 4K$ states (in this example, both E_1 and H_1 are PH distributions with 2 phases). Now using formula (6.11) for various values of λ we obtain Figure 6.9. The solid curves are for the E/E/1/K and H/H/1/K cases. The dashed curves are for the E/H/1/K and H/E/1/K cases.

When $\lambda \gg \mu$ we expect the asymptotic variance rate to be determined by the service distribution. This is because the server is almost always busy and thus we almost have a renewal output process with asymptotic variance $\mu\, c_S^2$ (where c_S^2 is the SCV of the service distribution). Similarly, when $\lambda \ll \mu$, we expect the asymptotic variance rate to be determined by the inter-arrival distribution. This is because the overflow rate is very small and thus $A(t) \approx Q(t) + D(t)$. Now, since $Q(t)$ is bounded, $\bar{V}_{\mathcal{D}} \approx \bar{V}_{\mathcal{A}} = \lambda\, c_A^2$ (where c_A^2 is the SCV of the inter-arrival distribution).

Now consider the case where $\lambda \approx \mu$: In the E/E/1/K and H/H/1/K systems (same SCV for inter-arrival and service distributions) we clearly observe the BRAVO effect: these curves have a pronounced local minimum at the vicinity of $\rho = 1$.

In the E/H/1/K and H/E/1/K systems, we observe a "smoothed step" in $\bar{V}_{\mathcal{D}}$ at the vicinity of $\rho = 1$ between the values, 2 and $\frac{1}{2}$ (approximately for finite K). This is due to the fact that

for $\lambda \ll \mu$ we have $\bar{V}_{\mathcal{D}} \approx \lambda\, c_A^2$ and for $\lambda \gg \mu$ we have $\bar{V}_{\mathcal{D}} \approx \mu c_S^2$ and $c_A^2 \neq c_S^2$ and is not directly due to the BRAVO effect. Nevertheless, we believe that traces of the BRAVO effect appear in the local minima (marked by '$*$' in the figure) at $\lambda \approx 0.9$ for the E/H/1/K case and $\lambda \approx 1.1$ for the H/E/1/K case.

A further observation is that when $c_A^2 = c_S^2$, the BRAVO effect appears to have the same "magnitude" as that of the M/M/1/K case: a reduction of the asymptotic variance rate by a factor of $\frac{2}{3}$ for large K. This observation is demonstrated in Figure 6.10 where we summarize results of several PH/PH/1/K systems with service and inter-arrival distributions having SCV: $\frac{1}{2},\ 1,\ \frac{6}{5},\ \frac{3}{2},\ 2$. These are calculated using Erlang, exponential and hyper-exponential distributions as before.

Discussion

The results presented here were motivated by the practical question of calculating the asymptotic variance rate of the output of finite birth and death queues. We found that this variance rate is optimized when the input rate and service rate are balanced, BRAVO. In deriving these results we discovered some unexpected phenomena, for which we do not yet have sufficient explanations.

Firstly, there is the "$\frac{2}{3}$ phenomenon", which we proved for $M/M/1/K$: in summary, when $\rho = 1$ and $K \to \infty$ the asymptotic variance rates of the outputs and of the overflows are the same and equal $\frac{2}{3}\lambda$ and this is possibly true for any choice of distribution of service and inter-arrival times as long as $c_A^2 = c_S^2$. We note that the value of $\frac{2}{3}$ for the asymptotic variance rate of the overflow process has been well known, as in the formula (6.22) which is due to Berger and Whitt. See also Theorem 5.7.4 of Whitt (2002), as well as Williams (1992). A well known fact is that the asymptotic variance of integrated Brownian motion with $\sigma^2 = 2$ is $\frac{2}{3}$ (cf. Parzen (1962)). We do not see an immediate connection here but suspect there may be one.

Further surprises which our analytic and numeric results show took the form of singularities that occur in the M/M/1/K queue at the point $\rho = 1$ when $K \to \infty$: (a) The y-intercept of the linear asymptote of the variance function is maximized and approaches a delta function. (b) The limiting correlation coefficient between the outputs and the overflows exhibits a sharp change of sign. (c) The graph of $\bar{T}_\lambda$ has a singular point, dropping from the values of $\approx K + 1$ to 0. It is plausible that all these are closely related, and may hold for general inter-arrival and service distributions. The details are yet to be discovered.

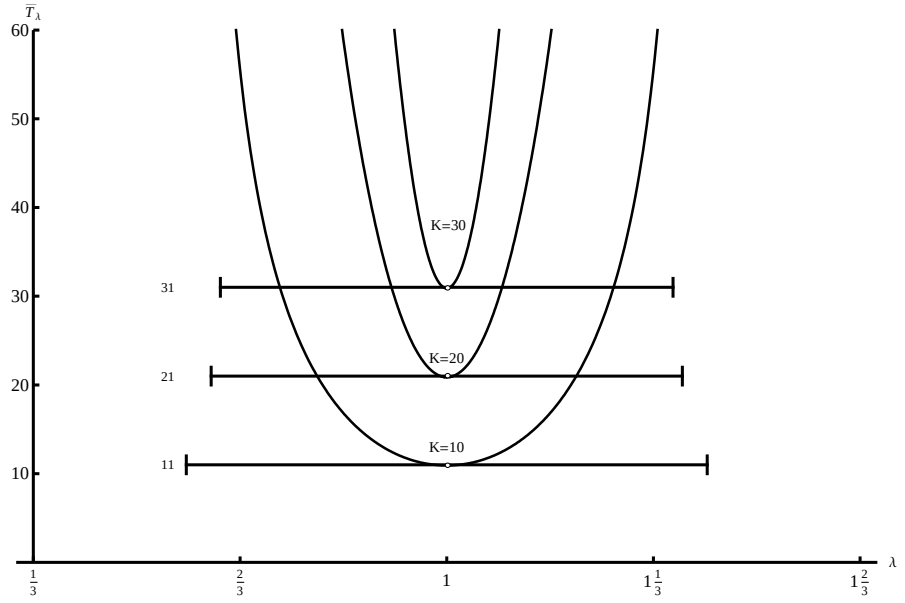

Figure 6.7: $\bar{T}_\lambda$ for $K = 10,\ 20,\ 30$. The horizontal intervals are exactly at heights 11, 21, 31 showing that $\bar{T}_{1^-} = \bar{T}_{1^+} \approx K + 1$. The horizontal intervals also specify the range of values for which $\bar{V}_{\mathcal{D}_1} \leq \bar{V}_{\mathcal{D}_\lambda}$.

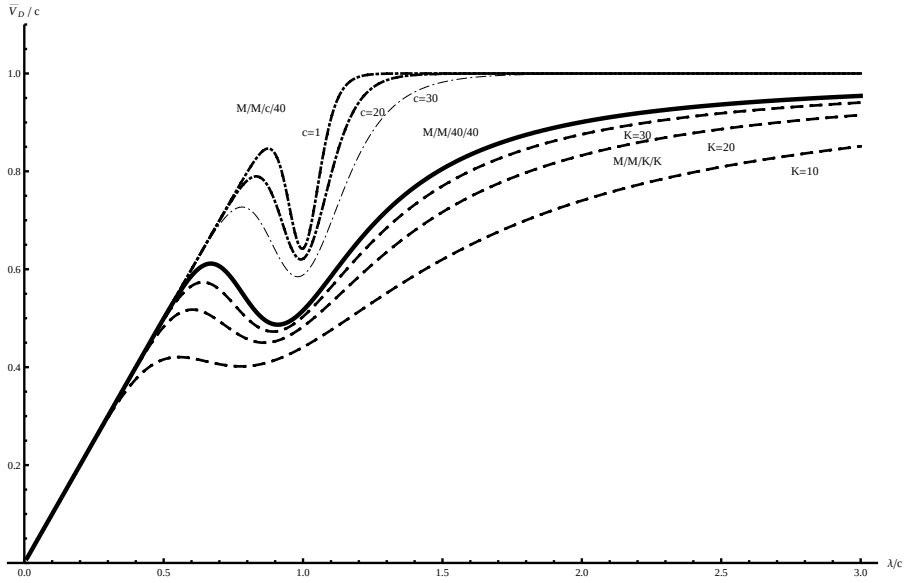

Figure 6.8: M/M/c/K: $\frac{\bar{V}_\mathcal{D}}{c}$ as a function of $\frac{\lambda}{c}$ when $\mu = 1$.

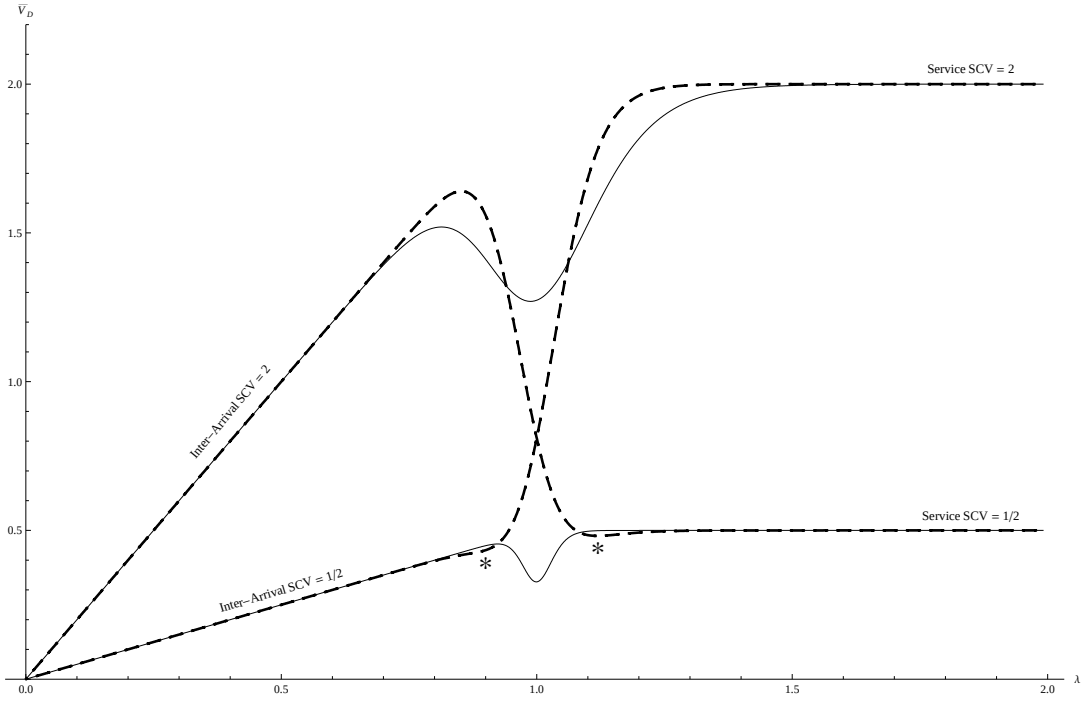

Figure 6.9: PH/PH/1/40: $\bar{V}_{\mathcal{D}}$ as a function of λ when $\mu = 1$ for four combinations of inter-arrival and service times distributions.

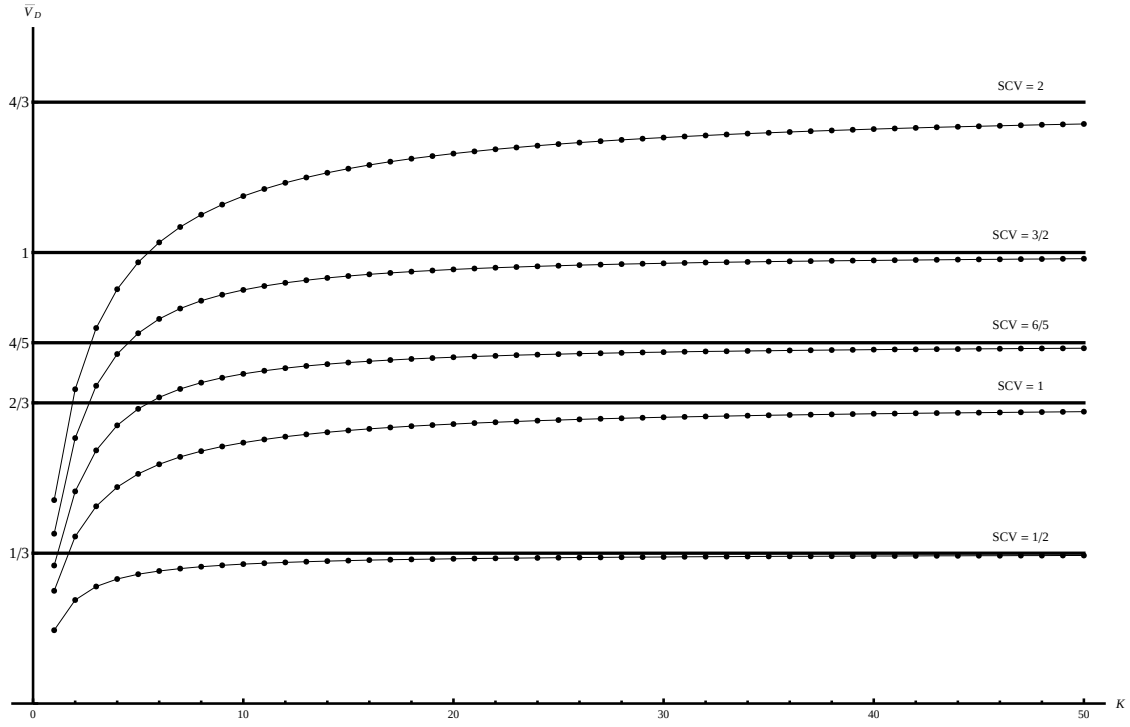

Figure 6.10: PH/PH/1/K: $\bar{V}_{\mathcal{D}}$ as a function of K for $\lambda = \mu = 1$, for various values of the SCVs of inter-arrival and service times. The horizontal lines are at $\frac{2}{3}$SCV.

CHAPTER 7

DIFFUSION SCALE ANALYSIS OF OUTPUTS

In this final chapter we present a method for finding diffusion limits of output processes of queueing networks with infinite virtual queues. Such diffusion limits are useful for to obtaining expressions for the asymptotic variance rate of outputs. Most of the analysis is for the example push-pull network discussed in Chapters 2, 4 and 5. In addition we present results for infinite supply re-entrant lines. Some of the contents of this chapter was published in Nazarathy and Weiss (2008c), along with the results of Chapter 4.

In Section 7.1 we repeat the definition of the push-pull model once again. This is here for convenience. We now make the additional assumption, (A3) that the processing times have a second moment. In Section 7.2 we present a diffusion limit theorem for the push-pull network. We continue in Section 7.3 where we discuss the covariance structure of the push-pull output processes and compare them to the KSRS network. In Section 7.4 we present diffusion limits for the output process of infinite supply re-entrant lines. In Section 7.5 we discuss the fact the diffusion limits presented are actually insensitive to the exact policy used.

7.1 Push-Pull Model, Again in Brief

We have already defined the push-pull network in Chapter 2 and again in Chapter 4. For convenience we briefly repeat the definition as in Chapter 4 with the addition of a third assumption regarding second moments of the processing times.

The *push-pull network* consists of two servers, numbered 1, 2 and two types of jobs numbered 1, 2 each of which is processed by both servers. Type 1 is processed by server 1 and then by server 2, while type 2 is first processed by server 2 and then by server 1. We call the first step a push of each type a *push activity* and the second step a *pull activity*. We denote by $Q_i(t), i = 2, 4$ the number of jobs in the two queues at time t (including the job in process), and by $D_i(t), i = 1, 2, 3, 4$ the number of jobs that have completed activity i in the time interval $[0, t]$. When $Q_4(t) > 0$, server 1 can either pull, by serving a type 2 job from $Q_4(t)$ or push, by serving a

type 1 job from the infinite supply. When $Q_4(t) = 0$ server 1 can still always push jobs of type 1. Hence, server 1 never needs to idle. Similarly for server 2. The long term average processing time for activity i is $1/\mu_i$, $i = 1, 2, 3, 4$. Let θ_i, $i = 1, 2, 3, 4$ be the long term fraction of time spent in activity i and let ν_i be the long term average rate of the departure process D_i, $i = 1, 2, 3, 4$. Then as explained previously we get:

$$\nu_1 = \nu_2 = \frac{\mu_1\mu_2(\mu_3 - \mu_4)}{\mu_1\mu_3 - \mu_2\mu_4}, \quad \nu_3 = \nu_4 = \frac{\mu_3\mu_4(\mu_1 - \mu_2)}{\mu_1\mu_3 - \mu_2\mu_4}.$$

We consider preemptive resume head of the line policies for the inherently stable case and inherently unstable case:

Inherently stable network: When $\mu_1 < \mu_2$ and $\mu_3 < \mu_4$, service of each type of jobs alone, by its second server, is a stable single server queue. In this case the policy which we use is preemptive resume head of the line priority for pull activities 4 and 2 over push activities 1 and 3. We refer to this as *Case 1*, and to the policy as *pull priority policy*.

Inherently unstable network: When $\mu_1 > \mu_2$ and $\mu_3 > \mu_4$, service of each type of jobs alone, by both servers results in an unstable single server queue. A policy that works here is that while $Q_2(t)$ is below some threshold level server 1 will push work to server 2, and server 1 will only pull from $Q_4(t)$ when $Q_2(t)$ is above the threshold, with a similar rule for server 2. We use a linear threshold to determine pull or push preemptive head of the line priority. We define a family of such policies, each determined by a pair of constants κ_1, κ_2 which satisfy $\kappa_1 > \frac{\mu_3}{\mu_1}$, $\kappa_2 > \frac{\mu_1}{\mu_3}$:

Server 1: Priority to pull activity 4 over push activity 1 if $0 < Q_4(t) < \kappa_1 Q_2(t)$.

Server 2: Priority to pull activity 2 over push activity 3 if $0 < Q_2(t) < \kappa_2 Q_4(t)$.

We refer to this as *Case 2*, and to the policy as *linear threshold policy*, see Figure 4.2.

We assume that the processing durations of the jobs in activity $i = 1, 2, 3, 4$ are drawn from a sequence of positive random variables: $\xi_i = \{\xi_i^j,\ j = 1, 2, \ldots\}$. The assumptions that we make regarding the processing durations are the same as in Chapter 4 with the addition of assumption (A3) which requires existence of second moments, with squared coefficients of variation c_i^2:

$$(A1) \quad \lim_{n\to\infty} \frac{\sum_{j=1}^n \xi_i^j}{n} = \frac{1}{\mu_i}, \text{ a.s.}$$
$$\text{for some } \mu_i \in (0, \infty),\ i = 1, 2, 3, 4.$$

$$(A2) \quad \begin{cases} (a) & \xi_i,\ i = 1, 2, 3, 4 \\ & \text{are mutually independent i.i.d.} \\ (b) & P(\xi_i^1 \geq x) > 0 \text{ for all } x > 0,\ i = 1, 3. \\ & \exists k_0^i > 0,\ q_i(\cdot) \geq 0 \text{ with } \int_0^\infty q_i(x)dx > 0 : \\ & P(\xi_i^1 + \ldots + \xi_i^{k_0^i} \in dx) \geq q_i(x)dx,\ \ i = 1, 3. \\ (b') & \text{Compact sets are petite.} \end{cases}$$

$$(A3) \quad \mu_i^2 \text{Var}(\xi_i^1) = c_i^2,$$
$$\text{for some } c_i^2 \in [0, \infty),\ \ i = 1, 2, 3, 4.$$

We associate counting processes with each activity i:

$$S_i(t) = \sup\{n : \sum_{j=1}^{n} \xi_i^j \leq t\}, \quad t \geq 0.$$

We denote by $T_i(t)$, $i = 1, 2, 3, 4$, the total time that the server allocates to the processing of activity i during the interval $[0, t]$. We require that $T_i(0) = 0$ and that $T_i(\cdot)$ be nondecreasing. Under our policies of full utilization, the servers never idle, thus:

$$T_1(t) + T_4(t) = t, \qquad T_2(t) + T_3(t) = t. \tag{7.1}$$

Note that $T_i(\cdot)$ are Lipschitz, and are therefore absolutely continuous. Thus their derivative exists almost everywhere with respect to Lebesgue measure on $[0, \infty)$. The number of jobs that have completed processing of activity i by time t is $D_i(t) = S_i(T_i(t))$. Let $Q_i(0)$, $i = 2, 4$ be the initial queue lengths. The number of jobs at time t is:

$$Q_i(t) = Q_i(0) + D_{i-1}(t) - D_i(t), \quad i = 2, 4. \tag{7.2}$$

We further require that $Q_i(\cdot) \geq 0$ for $i = 2, 4$.

In Chapter 4 we studied the network under fluid scaling by considering the six dimensional network process $Y(t) = (Q(t), T(t))$ parameterized by $n = 1, 2, ...$ as follows: For each n set the initial queue lengths as $Q^n(0)$, and let $Y^n(t)$ be the network process starting from this initial condition, where all the Y^n share the same sequences of random processing times $\xi_i, i = 1, 2, 3, 4$. Denote by $Y^n(t, \omega)$ the realization of the n'th network process for some ω in the sample space. We defined *fluid scalings* as:

$$\bar{Y}^n(t, \omega) = \frac{Y^n(nt, \omega)}{n}.$$

A function $\bar{Y}(t) = (\bar{Q}(t), \bar{T}(t))$ is said to be a *fluid limit* of our network if there exists a sequence of integers $r \to \infty$ and a sample path ω such that:

$$\bar{Y}^r(\cdot, \omega) \to \bar{Y}(\cdot), \text{ u.o.c.}$$

Under the above conditions (without requiring assumption (A3)) we have shown in Chapter 4:

(R1) $\bar{T}^n(t) \to \bar{T}(t) = \theta t$ and $\bar{D}^n(t) \to \bar{D}(t) = \nu t$ u.o.c as $n \to \infty$.

(R2) $Q_i(t), i = 1, 2, 3, 4$ has a stationary limiting distribution.

Result (R1) was obtained in Corollary 4.1 and result (R2) is an immediate consequence of the positive Harris recurrence shown in Theorem 4.2. We shall now use the results (R1) and (R2) to obtain a diffusion limit.

7.2 A Diffusion Limit for the Push-Pull Network

We assume (A1), (A2) and (A3) and consider the behavior of the push-pull network under diffusion scaling. We find that the queues are 0 on the diffusion scale, and the output processes $D_i(t)$ converge under diffusion scaling to Brownian motions. We calculate the parameters of these, including the asymptotic variance of the outputs and the covariances between the output streams.

Define diffusion scalings for $n = 1, 2, \ldots$. First denote

$$\bar{S}(t) = \lim_{n\to\infty} \bar{S}^n(t) = \lim_{n\to\infty} \frac{S(nt)}{n} = \mu t,$$

where the limit exists a.s. u.o.c. by Assumption (A1). Further use the fluid limit processes of Section 4.3, Corollary 4.1. The diffusion scalings are:

$$\begin{array}{ll} \hat{S}_i^n(t) = \frac{S_i(nt)-\bar{S}_i(nt)}{\sqrt{n}}, & \hat{T}_i^n(t) = \frac{T_i(nt)-\bar{T}_i(nt)}{\sqrt{n}}, \\ \hat{D}_i^n(t) = \frac{D_i(nt)-\bar{D}_i(nt)}{\sqrt{n}}, & \hat{Q}_i^n(t) = \frac{Q_i(nt)}{\sqrt{n}}. \end{array} \tag{7.3}$$

Note that in this analysis we use a fixed $Q(0)$, which does not change with n.

Define the 10 dimensional diffusion scaled process:

$$\hat{X}^n(t) = (\hat{D}^n(t), \hat{T}^n(t), \hat{Q}^n(t))$$

The following theorem describes the diffusion limit for our model.

Theorem 7.1. *Consider the push-pull network, under Assumptions (A1–A3), for Case 1 under pull priority policy, and for Case 2 under linear threshold policy. Then as $n \to \infty$, $\hat{X}^n \Rightarrow \hat{X}$, where $\hat{X}(t)$ is a 10 dimensional driftless Brownian motion. Furthermore,*

$$\begin{array}{rcl} \hat{D}_1^n(t) - \hat{D}_2^n(t) = \hat{Q}_2^n(t) & \Rightarrow & 0, \\ \hat{D}_4^n(t) - \hat{D}_3^n(t) = \hat{Q}_4^n(t) & \Rightarrow & 0, \end{array} \tag{7.4}$$

$$\hat{T}_1^n(t) + \hat{T}_4^n(t) = \hat{T}_3^n(t) + \hat{T}_2^n(t) = 0, \tag{7.5}$$

and the variances and covariances of the limiting Brownian motions are given by:

$$Var(\hat{D}_2(1)) = \frac{\mu_1\mu_2}{(\mu_1\mu_3 - \mu_2\mu_4)^3}\Big(\mu_1\mu_2\mu_3\mu_4(c_3^2+c_4^2)(\mu_1-\mu_2)+(\mu_1^2\mu_3^2c_2^2+\mu_2^2\mu_4^2c_1^2)(\mu_3-\mu_4)\Big), \tag{7.6}$$

$$Cov(\hat{D}_2(1), \hat{D}_4(1)) = -\frac{\mu_1\mu_2\mu_3\mu_4}{(\mu_1\mu_3 - \mu_2\mu_4)^3}\Big((\mu_1\mu_3c_4^2+\mu_2\mu_4c_3^2)(\mu_1-\mu_2)+(\mu_1\mu_3c_2^2+\mu_2\mu_4c_1^2)(\mu_3-\mu_4)\Big), \tag{7.7}$$

with a symmetric expression for $Var(\hat{D}_4(1))$. Similar expressions for variances and covariances of $\hat{T}_2(\cdot)$, $\hat{T}_4(\cdot)$ may be read off from (7.12).

Proof. The equalities (7.4) and (7.5) follow immediately from (7.2) and (7.1). The convergence to 0 in (7.4) follows from result (R2), since $Q_i(t)$ has a limiting stationary distribution, therefore

$Q_i(nt)$ converges to this limiting distribution as $n \to \infty$, and dividing by $\sqrt{n}$ implies converges to 0 in probability and therefore also weakly.

The rest of the proof and the calculations are straightforward:

$$\begin{aligned}
\hat{D}_i^n(t) &= \tfrac{D_i(nt)-\bar{D}_i(nt)}{\sqrt{n}} \\
&= \tfrac{S_i(n\bar{T}_i^n(t))-\bar{S}_i(n\bar{T}_i^n(t))}{\sqrt{n}} + \tfrac{\bar{S}_i(n\bar{T}_i^n(t))}{\sqrt{n}} - \tfrac{\bar{D}_i(nt)}{\sqrt{n}} \\
&= \hat{S}_i^n(\bar{T}_i^n(t)) + \mu_i \tfrac{T_i(nt)-\bar{T}_i(nt)}{\sqrt{n}} + \mu_i \tfrac{\bar{T}_i(nt)}{\sqrt{n}} - \tfrac{\bar{D}_i(nt)}{\sqrt{n}} \\
&= \hat{S}_i^n(\bar{T}_i^n(t)) + \mu_i \hat{T}_i^n(t) + \theta_i\mu_i\sqrt{n}t - \theta_i\mu_i\sqrt{n}t,
\end{aligned}$$

where all we did is to add and subtract quantities, use the definitions (7.3), and use $\bar{S}_i(t) = \mu_i t$ (by Assumption (A1)), and $\bar{T}_i(t) = \theta_i t$, $\bar{D}_i(t) = \nu_i t = \mu_i\theta_i t$ by result (R1).

Define $\hat{P}_i^n(t) = \hat{S}_i^n(\bar{T}_i^n(t))$, $i = 1, 2, 3, 4$, then summarizing the above and also using similar calculations (for (7.9) and (7.10)) we obtain:

$$\hat{D}_i^n(t) = \hat{P}_i^n(t) + \mu_i \hat{T}_i^n(t), \quad i = 1, 2, 3, 4, \tag{7.8}$$

$$\hat{Q}_i^n(t) = \hat{D}_{i-1}^n(t) - \hat{D}_i^n(t), \quad i = 2, 4, \tag{7.9}$$

$$\hat{T}_2^n(t) = -\hat{T}_3^n(t), \qquad \hat{T}_4^n(t) = -\hat{T}_1^n(t). \tag{7.10}$$

Now using (7.8)–(7.10):

$$\begin{bmatrix} \hat{D}_2^n(t) \\ \hat{D}_4^n(t) \\ \hat{T}_2^n(t) \\ \hat{T}_4^n(t) \end{bmatrix} = \mathbf{A}\,\hat{P}^n(t) + \mathbf{B} \begin{bmatrix} \hat{Q}_2^n(t) \\ \hat{Q}_4^n(t) \end{bmatrix}, \tag{7.11}$$

where

$$\mathbf{A} = \frac{1}{\mu_1\mu_3 - \mu_2\mu_4} \begin{bmatrix} -\mu_2\mu_4 & \mu_1\mu_3 & \mu_1\mu_2 & -\mu_1\mu_2 \\ \mu_3\mu_4 & -\mu_3\mu_4 & -\mu_2\mu_4 & \mu_1\mu_3 \\ -\mu_4 & \mu_4 & \mu_1 & -\mu_1 \\ \mu_3 & -\mu_3 & -\mu_2 & \mu_2 \end{bmatrix},$$

and

$$\mathbf{B} = \frac{1}{\mu_1\mu_3 - \mu_2\mu_4} \begin{bmatrix} \mu_2\mu_4 & -\mu_1\mu_2 \\ -\mu_3\mu_4 & \mu_2\mu_4 \\ \mu_4 & -\mu_1 \\ -\mu_3 & \mu_2 \end{bmatrix}.$$

By the functional central limit theorem for renewal processes and the continuous mapping theorem (cf. Glynn (1990)) we have $\hat{P}^n(t) \Rightarrow \hat{P}(t)$ where $\hat{P}(t)$ is a 4 dimensional driftless Brownian motion with a diagonal covariance matrix $\mathbf{\Lambda}$, having entries

$$\mathrm{Var}(\hat{P}_i(1)) = \mu_i c_i^2 \theta_i,\ i = 1, 2, 3, 4.$$

Incorporating the above with the weak convergence of $\hat{Q}^n$ to 0, we have that $(\hat{D}_2^n(t), \hat{D}_4^n(t), \hat{T}_2^n(t), \hat{T}_4^n(t))$ converges to a driftless Brownian motion process with covariance matrix:

$$\mathbf{\Gamma} = \mathbf{A}\mathbf{\Lambda}\mathbf{A}'. \tag{7.12}$$

□

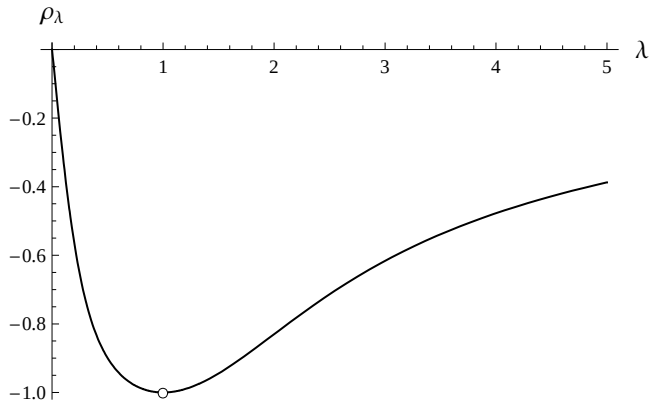

Figure 7.1: The correlation between outputs of a symmetric push-pull network.

7.3 Negative Covariance of Outputs of the Push-Pull Network

It is evident from (7.7) that $\text{Cov}(\hat{D}_2(t), \hat{D}_4(t)) < 0$. Also, when all activity processing times have the same squared coefficient of variation c^2, then both the variance and the covariance in (7.6,7.7) are linear in c^2.

In Figure 7.1 we illustrate the negative correlation between the output processes of our network. We plot as a function of λ:

$$\rho_\lambda = \frac{\text{Cov}(\hat{D}_2(1), \hat{D}_4(1))}{\sqrt{\text{Var}(\hat{D}_2(1))\text{Var}(\hat{D}_4(1))}}, \tag{7.13}$$

for symmetric push-pull networks with parameters $c_i^2 = c^2, i = 1, 2, 3, 4$, $\mu_2 = \mu_4 = 1$, $\mu_1 = \mu_3 = \lambda$.

Our analysis applies to all $\lambda \neq 1$. When $\lambda = 1$ we have a completely balanced network (as defined in Section 2.5) and with our policies, under diffusion scaling the queues do not converge to 0, so the analysis in this chapter does not apply.

Note that for $1/2 < \lambda < 2$, i.e when the ratio of processing times for each type of job on the two servers is not too far from 1, we get $-1 < \rho_\lambda < -0.8$, so the negative correlation is very high. Most surprisingly, as $\lambda \to 1$ the correlation approaches -1, and we are close to complete resource pooling Dai and Lin (2006).

When λ is very small or very large the correlation approaches zero. This is intuitively clear, since each server is now spending almost all of its time on just one type of job, and so the fluctuations in D_2 depend mostly on the processing times of jobs of type 1, and the fluctuations of D_4 will depend mostly on the processing times of jobs of type 2, and hence they will be almost independent.

Comparing to the KSRS Network

The Kumar-Seidman Rybko-Stolyar multi-class queueing network (see Chapter 1) differs from our push-pull network in that instead of infinite supply of jobs there are two stochastic arrival streams of jobs of type 1 and of type 2, with long term average arrival rates α_1, α_3. In that case

there are 4 queues $Q_i(t)$ of jobs waiting for activities $i = 1, 2, 3, 4$ in the network, and the offered loads for servers 1 and 2 are $\rho_1 = \alpha_1/\mu_1 + \alpha_3/\mu_4$ and $\rho_2 = \alpha_3/\mu_3 + \alpha_1/\mu_2$ respectively.

We have already compared the queue level behaviour of KSRS and push-pull in Chapter 4. We now compare the behavior of the output processes, $D_i(t)$, $i = 1, 2, 3, 4$ in the KSRS network and in the push-pull network, under diffusion scaling.

In the KSRS network with $\rho_i < 1$, $i = 1, 2$ the diffusion scaled queue lengths will be 0. Therefore on a diffusion scale, jobs of type 1 have arrivals, departures from queue 1, and departures from queue 2, which are all identical Brownian motions. Similarly for type 2. In particular, the diffusion scaled flow of jobs of type 1 and of jobs of type 2 will be independent. This fully describes the diffusion scale behavior, for fixed $\rho_i < 1$.

Under balanced heavy traffic the behavior of the output processes of the KSRS network seems to be much more complex. The four queue length processes will be reflected Brownian processes, and will affect the diffusion scaled output processes. To the best of our knowledge the behavior of the output processes in that case has not been investigated. We note that even the output process of a single server queue, under balanced heavy traffic, poses some as yet unanswered questions (cf. Harrison and Williams (1992) and Chapter 5).

In contrast to that, in the push-pull network, operated with our policies, under full utilization, the diffusion scaled queue lengths are 0. As a result we can analyze the output processes of the two types of jobs. What we find is that the output processes of jobs of types 1 and 2 that leave the network converge under diffusion scaling to two standard Brownian motions, but these two Brownian motions are highly negatively correlated.

7.4 A Diffusion Limit for Re-Entrant Line Outputs

We now consider the general infinite supply re-entrant line that was surveyed in Section 2.4. We apply the same type of analysis of Section 7.2 to obtain a diffusion limit for the output process of this queueing network.

Our model consists of K consecutive steps on I servers where the first step has an IVQ and the other steps have standard buffers. Generally we have that $I < K$, thus at least some of the servers are "revisited" by jobs. The set of steps performed on server i is denoted by C_i. All processing times are assumed to be independent and we further assume that the processing times of each step are identically distributed with mean $m_k, k = 1, \ldots, K$ and rates $\mu_k = m_k^{-1}$. The steps of the first server (with the IVQ) are C_1. We denote the *flow rate* to be:

$$\lambda^* = \frac{1}{\sum_{k \in C_1} m_k},$$

and further denote,

$$\rho_i = \lambda \sum_{k \in C_i} m_k, \;\; i = 2, \ldots, I$$

and assume that $\rho_i < 1, i = 2, \ldots, K$. We also require the technical assumption that the processing times of the first step (IVQ) are unbounded. Under these assumptions, Guo and Zhang (2007) have shown that the LBFS policy maintains the associated general state space Markov

process, positive Harris recurrent [1]. We further assume that processing times of server 1 steps have a finite second moment and denote their variance by $\sigma_k^2, k \in C_1$.

We are interested in the output counting process from the last buffer, denoted by $D(t)$. We use $Q_k^+(t), k \in C_1$ to denote the number of jobs in the queues that are downstream to buffer k, i.e:

$$Q_k^+(t) = \sum_{j=k+1}^{K} Q_k(t), \;\; k \in C_1,$$

where as usual, $Q_k(t)$ is the queue level of buffer k. Note that $Q_K^+(t) = 0$. We shall denote by $D_k(t), k \in C_1$ the output counting process from buffer k and we thus have:

$$D_k(t) = Q_k^+(t) + D(t), k \in C_1. \tag{7.14}$$

We further denote by $T_k(t), k \in C_1$ the amount of time that server 1 spends on buffer k during the time [0,t] and thus:

$$\sum_{k \in C_1} T_k(t) = t. \tag{7.15}$$

As in previous sections and chapters, we use the renewal primitives $S_k(t)$ and thus $D_k(t) = T_k(S_k(t))$. We also define fluid and diffusion scalings of all of the quantities in the same manner that was defined in equation (7.3). Note that the long run proportion of time which server 1 spends on step $k \in C_1$ is λm_k, i.e.

$$\lim_{n \to \infty} \bar{T}_k^n(t) = \lambda m_k t, \;\; k \in C_1$$

Here is our result regarding the asymptotic variance of the output process which we obtain from its diffusion scaling:

Theorem 7.2. *Consider an infinite re-entrant line operating under the LBFS policy with $\rho_i < 1, i = 2, \ldots, I$. Then the diffusion scaled output process,*

$$\hat{D}^n(t) = \frac{D(nt) - \lambda nt}{\sqrt{n}},$$

converges weakly to a drift less Brownian motion, $\hat{D}(t)$ with variance parameter:

$$Var(\hat{D}(1)) = \frac{\sum_{k \in C_1} \sigma_k^2}{(\sum_{k \in C_1} m_k)^3} \tag{7.16}$$

Proof. Using the exact same calculations as in the proof of Theorem 7.1 we have:

$$\hat{D}_k^n(t) = \hat{S}_k^n(\bar{T}^n(t)) + \mu_k \hat{T}_k^n(t) \;\; , k \in C_1, \tag{7.17}$$

And further (by applying the definition of the diffusion scalings):

$$\sum_{k \in C_1} \hat{T}_k^n(t) = 0. \tag{7.18}$$

[1] Actually in Guo and Zhang (2007) it is also required that the processing time of the first step be spread out, but this assumption may be relaxed to only requiring unboundedness – personal communication, Weiss, Zhang, 2008.

Summing over the equations of (7.17) and along with (7.18), we obtain:

$$\sum_{k\in C_1} \frac{\hat{D}_k^n(t)}{\mu_k} - \sum_{k\in C_1} \frac{\hat{P}_k^n(t)}{\mu_k} = 0, \tag{7.19}$$

where as in the proof of Theorem 7.1, $\hat{P}_k^n(t) = \hat{S}_k^n(\bar{T}_k^n(t))$, $k \in C_1$. Further, from the dynamics of the network and using the definitions of the fluid scalings:

$$\hat{D}_k^n(t) = \hat{Q^+}_k^n(t) + \hat{D}^n(t), k \in C_1. \tag{7.20}$$

Now substituting the equations (7.20) in (7.19) and solving for $\hat{D}^n(t)$ we obtain:

$$\hat{D}^n(t) = \lambda \sum_{k\in C_1} m_k \hat{P}_k^n(t) + \sum_{k\in C_1} b_k \hat{Q}_k^n(t), \tag{7.21}$$

where $b_k, k = 1, \ldots, K$ are some constants (expressions of m_k). Now as in Theorem 7.1, we have that $\hat{P}_k^n(t), k \in C_1$ converge to independent drift less Brownian motions with,

$$\mathrm{Var}(\hat{P}_k(1)) = \lambda \frac{\sigma_k^2}{m_k^2}.$$

In addition since the network is positive Harris recurrent we have that $\hat{Q}_k^n(t) \Rightarrow 0$. The linear transformation of this $|C_1|$ dimensional, random vector into the 1 dimensional output process yields the result. □

Note that if $C_1 = \{1, \ldots, K\}$ (the system is re-entrant through a single server) then the output process is actually a renewal process (given that the system started empty) with inter-output times having mean $\sum_{k\in C_1} m_k$ and variance $\sum_{k\in C_1} \sigma_k^2$. In this case, the asymptotic variance of the above theorem immediately follows. Our theorem shows that even when the output is not renewal (as is the case when there is more then one server) then the asymptotic variance rate of the output still only depends on the first server and is equal to that of the renewal output case.

7.5 Insensitivity to Policy

The expression for the asymptotic variance rate of the push-pull, (7.6) has appeared before in Chapter 5, (5.6) for the case of operations 1 and 3 being exponential. Our new result is much stronger not only because of the general processing times but also because it is for both the pull priority policy, case 1 and the linear threshold policy, case 2 and this is in contrast to the result of Chapter 5 which is only for the pull priority policy, case 1. In-fact, our current result holds for any policy that operates at the solved flow rate ν_2, ν_4 and maintains a stable system. By following the proof of Theorem 7.1 it can be verified that all that is required from the policy is that results (R1) and (R2) are maintained. In particular, the calculations for Case 1 and Case 2 are the same.

The same holds for the re-entrant line results of Section 7.4. While the results were stated for the LBFS policy for which there exist positive Harris recurrence results, our analysis holds for any policy that maintains the system stable.

We thus reach the surprising conclusion that the diffusion scale output processes $\hat{D}(t)$ do not depend on the policy, so long as it is fully utilizing and stabilizing.

Note that instead of (R2) we may use a seemingly weaker condition: $\hat{Q}^n(t) \Rightarrow 0$. Unfortunately, we have been unable to show this weak convergence without obtaining positive Harris recurrence. For example, it would have been nice to skip the minorization proofs of Section 4.5 and thus stop at a weaker result of positive recurrence (without Harris). But to the best of our knowledge, we cannot use such a result to obtain the desired weak convergence of the scaled process [2].

[2]This is based on personal communication with Serguei Foss, 2008.

BIBLIOGRAPHY

Adan, I. and Weiss, G. (2005). A two node Jackson network with infinite supply of work. *Probability in the Engineering and Informational Sciences*, **19**(2), 191–212.

Adan, I. and Weiss, G. (2006). Analysis of a simple Markovian re-entrant line with infinite supply of work under the LBFS policy. *Queueing Systems*, **54**(3), 169–183.

Adan, I., Wessels, J., and Zijm, W. (1993). A compensation approach for two-dimensional Markov processes. *Advances in Applied Probability*, **25**(4), 783–817.

Albin, S. (1984). Approximating a point process by a renewal process, II: Superposition arrival processes to queues. *Operations Research*, **32**(5), 1133–1162.

Anderson, E. (1981). A new continuous model for job-shop scheduling. *International Journal of Systems Science*, **12**(12), 1469–1475.

Anderson, E. and Nash, P. (1987). *Linear Programming in Infinite-Dimensional Spaces: Theory and Applications*. John Wiley & Sons.

Araghi, M. and Balcıoglu, B. (2008). A new renewal approximation for certain autocorrelated processes. *Operations Research Letters*, **36**(1), 133–139.

Artalejo, J. (2000). G-networks: A versatile approach for work removal in queueing networks. *European Journal of Operational Research*, **126**(2), 233–249.

Asmussen, S. (2003). *Applied Probability and Queues*. Springer-Verlag.

Ata, B. and Lin, W. (2008). Heavy traffic analysis of maximum pressure policies for stochastic processing networks with multiple bottlenecks. *Preprint*.

Avram, F., Bertsimas, D., and Ricard, M. (1995). Fluid models of sequencing problems in open queueing networks; an optimal control approach. *Institute for Mathematics and Its Applications*, **71**, 199.

Baccelli, F. and Foss, S. (1994). Ergodicity of Jackson-type queueing networks. *Queueing systems*, **17**(1), 5–72.

Baccelli, F., Massey, W., and Towsley, D. (1989). Acyclic fork-join queuing networks. *Journal of the ACM (JACM)*, **36**(3), 615–642.

Balcıoglu, B., Jagerman, D., and Altıok, T. (2008). Merging and splitting autocorrelated arrival processes and impact on queueing performance. *Performance Evaluation*, **65**(9), 653–669.

Balsamo, S., de Nitto Persone, V., and Onvural, R. (2001). *Analysis of Queueing Networks with Blocking*. Kluwer Academic Publishers.

Barnes, J. and Disney, R. (1990). Traffic processes in a class of finite Markovian queues. *Queueing Systems*, **6**, 311–326.

Baskett, F., Chandy, K., Muntz, R., and Palacios, F. (1975). Open, closed, and mixed networks of queues with different classes of customers. *J. Assoc. Comput. Mach*, **22**(2), 248–260.

Bellman, R. (1953). Bottleneck problems and dynamic programming. *Proceedings of the National Academy of Sciences of the United States of America*, **39**(9), 947–951.

Berger, A. W. and Whitt, W. (1992). The Brownian approximation for rate-control throttles and the G/G/1/C queue. *Discrete Event Dynamic Systems: Theory and Applications*, **2**, 7–60.

Bitran, G. and Dasu, S. (1993). Approximating Nonrenewal Processes by Markov Chains: Use of Super-Erlang SE Chains,". *Operations Research*, **41**(5), 903–923.

Boxma, O. (1988). Sojourn times in cyclic queues - the influence of the slowest server. *In Computer Performance and Reliability, Eds. G. Iazeolla, P.J. Courtois and O.J. Boxma. (North-Holland)*.

Bramson, M. (1994). Instability of FIFO queueing networks with quick service times. *Annals of Applied Probability*, **4**(3), 693–718.

Bramson, M. (1998a). Stability of two families of queueing networks and a discussion of fluid limits. *Queueing Systems*, **28**(1-3), 7–31.

Bramson, M. (1998b). State space collapse with application to heavy traffic limits for multiclass queueing networks. *Queueing Systems*, **30**, 89–148.

Bramson, M. (2008). *Stability of Queueing Networks*. Springer.

Branford, A. J. (1986). On a property of finite-state birth and death processes. *J. Appl. Prob.*, **23**, 859–866.

Breuer, L. and Baum, D. (2005). *An Introduction to Queueing Theory and Matrix-Analytic Methods*. Springer.

Brown, M. and Solomon, H. (1974). A second order approximation for the variance of a renewal reward process. *Stochastic Processes and their Applications*, **3**, 301–314.

Burke, P. (1956). The output of a queuing system. *Operations Research*, **4**(6), 699–704.

Caldentey, R. (2001). Approximations for Multi-Class Departure Processes. *Queueing Systems*, **38**, 205–212.

Chandramohan, J., Foley, R. D., and Disney, R. L. (1985). Thinning of point processes – covariance analysis. *Adv. Appl. Prob.*, **17**, 127–146.

Chen, H. and Yao, D. (1993). Dynamic scheduling of a multiclass fluid network. *Operations Research*, **41**(6), 1104–1115.

Chen, H. and Yao, D. (2001). *Fundamentals of Queueing Networks: Performance, Asymptotics, and Optimization*. Springer.

Chen, M., Pandit, C., and Meyn, S. P. (2003). In Search of Sensitivity in Network Optimization. *Queueing Systems*, **44**(4), 313–363.

Chen, R. and Meyn, S. (1999). Value iteration and optimization of multiclass queueing networks. *Queueing Systems*, **32**(1), 65–97.

Cinlar, E. and Disney, R. L. (1967). Stream of overflows from a finite queue. *Operations Research*, **15**(1), 131–134.

Ciprut, P., Hongler, M., Salama, Y., and de Lausanne, E. (1999). On the variance of the production output of transfer lines. *Robotics and Automation, IEEE Transactions on*, **15**(1), 33–43.

Cohen, J. (1982). *The Single Server Queue*. North-Holland.

Connors, D., Feigin, G., and Yao, D. (1994). Scheduling semiconductor lines using a fluid network model. *Robotics and Automation, IEEE Transactions on*, **10**(2), 88–98.

Cox, D. and Isham, V. (1980). *Point Processes*. Chapman and Hall.

Dai, J. G. (1995). On positive Harris recurrence of multiclass queueing networks: A unified approach via fluid limit models. *The Annals of Applied Probability*, **5**(1), 49–77.

Dai, J. G. (1996). A fluid-limit model criterion for instability of multiclass queueing networks. *Ann. Appl. Probab*, **6**(3), 751–757.

Dai, J. G. and Harrison, J. M. (1992). Reflected Brownian motion in an orthant: numerical methods for steady-state analysis. *Ann. Appl. Probab*, **2**(1), 65–86.

Dai, J. G. and Lin, W. (2005). Maximum pressure policies in stochastic processing networks. *Operations Research*, **53**(2), 197–218.

Dai, J. G. and Lin, W. (2006). Asymptotic optimality of maximum pressure policies in stochastic processing networks. *Preprint*.

Dai, J. G. and Weiss, G. (1996). Stability and instability of fluid models for re-entrant lines. *Mathematics of Operations Research*, **21**(1), 115–134.

Daley, D. (1976). Queueing output processes. *Adv. Appl. Prob.*, **8**, 395–415.

Dao-Thi, T. and Mairesse, J. (2006). Zero-automatic networks. In *Proceedings of the 1st international conference on Performance evaluation methodolgies and tools*. ACM Press New York, NY, USA.

Davis, M. H. A. (1984). Piecewise-deterministic Markov processes: A general class of non-diffusion stochastic models. *Journal of Royal Statistical Society. Series B.*, **46**(3), 353–388.

Disney, R. L. and de Morais, P. R. (1976). Covariance properties for the departure process of M/E_k/1/N queues. *AIIE Transactions*, **8**(2), 169–175.

Disney, R. L. and Kiessler, P. C. (1987). *Traffic Processes in Queueing Networks – A Markov Renewal Approach*. The Johns Hopkins University Press.

Disney, R. L. and Konig, D. (1985). Queueing networks: A survey of their random processes. *SIAM Review*, **27**(3), 335–403.

Fischer, W. and Meier-Hellstern, K. (1992). The Markov-modulated Poisson process (MMPP) cookbook. *Performance Evaluation*, **18**, 149–171.

Fleischer, L. and Sethuraman, J. (2003). Approximately optimal control of fluid networks. *Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 56–65.

Ganesh, A., O'Connell, N., and Wischik, D. (2004). *Big Queues*. Springer.

Gans, N., Koole, G., and Mandelbaum, A. (2003). Telephone call centers: Tutorial, review, and research prospects. *Manufacturing and Service Operations Management*, **5**(2), 79–141.

Gershwin, S. B. (1993). Variance of output of a tandem production system. *in: Queueing Networks with Finite Capacity,* eds R. Onvural and I. Akyildiz, Proceedings of the Second International Conference on Queueing Networks with Finite Capacity (Elsevier, Amsterdam).

Glynn, P. W. (1990). Diffusion approximations. *In Handbooks in Operations Research, Vol 2, D.P. Heyman and M.J. Sobel (eds.), North-Holland, Amsterdam*, pages 145–198.

Goemans, M. and Williamson, D. (1996). The primal-dual method for approximation algorithms and its application to network design problems. *Approximation algorithms for NP-hard problems table of contents*, pages 144–191.

Guo, Y. (2008). Fluid model criterion for instability of re-entrant line with infinite supply of work. *Preprint*.

Guo, Y. and Yang, J. (2007). Stability of a 2-station-5-class re-entrant line with infinite supply of work. *J.Systems Sci&Comp*, **21**(2), 283–295.

Guo, Y. and Zhang, H. (2006). On the stability of a simple re-entrant line with infinite supply. *OR Transactions*, **10**(2), 75–85.

Guo, Y. and Zhang, H. (2007). Positive Harris recurrence of re-entrant lines with infinite supply. *Preprint*.

Halfin, S. and Whitt, W. (1981). Heavy-Traffic Limits for Queues With Many Exponential Servers. *Operations Research*, **29**(3), 567–588.

Harrison, J. M. (1978). The diffusion approximation for tandem queues in heavy traffic. *Adv. Appl. Probab*, **10**, 886–905.

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.)*, pages 147–186.

Harrison, J. M. (1996). The BIGSTEP approach to flow management in stochastic processing networks. *Stochastic Networks: Theory and Applications*, **4**, 147–186.

Harrison, J. M. (2000). Brownian models of open processing networks: Canonical representation of workload. *Ann. Appl. Probab*, **10**(1), 75–103.

Harrison, J. M. (2002). Stochastic networks and activity analysis. *Analytic Methods in Applied Probability*, **207**, 53–76.

Harrison, J. M. (2003). A broader view of Brownian networks. *Annals of Applied Probability*, **13**(3), 1119–1150.

Harrison, J. M. and Reiman, M. (1981). Reflected Brownian motion on an orthant. *Ann. Probab*, **9**(2), 302–308.

Harrison, J. M. and Van Mieghem, J. (1997). Dynamic control of Brownian networks: state space collapse and equivalent workload formulations. *Ann. Appl. Probab*, **7**(3), 747–771.

Harrison, J. M. and Williams, R. (1992). Brownian models of feedforward queueing networks: Quasireversibility and product form solutions. *Ann. Appl. Probab*, **2**(2), 263–293.

Haverkort, B. (1995). Approximate Analysis of Networks of PH| PH| 1| Jf Queues: Theory & Tool Support. *Quantitative Evaluation of Computing and Communication Systems: 8th International Conference on Modelling Techniques and Tools for Computer Performance Evaluation, Performance Tools' 95, 8th GI/ITG Conference on Measuring, Modelling, and Evaluating Computing, and Communication Systems, MMB'95, Heidelberg, Germany, September 20-22, 1995: Proceedings*.

He, Q. and Neuts, M. F. (1998). Markov chains with marked transitions. *Stochastic Processes and their applications*, **74**, 37–52.

Heindl, A. and Telek, M. (2002). Output models of MAP/PH/1(/K) queues for an efficient network decomposition. *Performance Evaluation*, **49**(1-4), 321–339.

Heining, G. and Rost, K. (1988). On the inverses of Toeplitz-plus-Hankel matrices. *Linear Algebra and Its Applications*, **106**, 39–52.

Henderson, S. G., Meyn, S. P., and Tadic, V. B. (2003). Performance evaluation and policy selection in multiclass networks. *Discrete Event Dynamic Systems*, **13**(1-2), 149–189.

Hendricks, K. B. (1992). The output processes of serial production lines of exponential machines with finite buffers. *Operations Research*, **40**(6), 1139–1147.

Hendricks, K. B. and McClain, J. O. (1993). The output processes of serial production lines of general machines with finite buffers. *Management Science*, **39**(10), 1194–1201.

Iglehart, D. and Whitt, W. (1970). Multiple Channel Queues in Heavy Traffic. I. *Advances in Applied Probability*, **2**(1), 150–177.

Jackson, J. R. (1957). Networks of waiting lines. *Operations Research*, **5**(4), 518–521.

Jackson, J. R. (1963). Jobshop-like queueing systems. *Management Science*, **10**(1), 131–142.

Keilson, J. (1979). *Markov Chain Models – Rarity and Exponentiality*. Springer-Verlag.

Kelly, F. (1975). Networks of queues with customers of different types. *J. Appl. Probab*, **12**(3), 542–554.

Kelly, F. (1976). Networks of queues. *Adv. Appl. Prob*, **8**(2), 416–432.

Kelly, F. (1979). *Reversibility and Stochastic Networks*. John Wiley & Sons.

Kelly, F. (1991). Loss networks. *Ann. Appl. Probab*, **1**(3), 319–378.

Kelly, F. and Laws, C. (1993). Dynamic routing in open queueing networks. *Queueing Systems*, **13**, 47–86.

Kleinrock, L. (1974). *Queueing Systems. Volume I, Theory*. New York: Wiley.

Kopzon, A. (2006). *The push pull system: a queueing network with two machines that feed each other*. Ph.D. thesis, The University of Haifa.

Kopzon, A. and Weiss, G. (2002). A push pull queueing system. *Operations Research Letters*, **30**(6), 351–359.

Kopzon, A., Nazarathy, Y., and Weiss, G. (2008). A push pull system with infinite supply of work. *Preprint*.

Kumar, P. (1993). Re-entrant lines. *Queueing Systems*, **13**(1), 87–110.

Kumar, P. and Seidman, T. (1990). Dynamic instabilities and stabilization methods in distributed real-time scheduling of manufacturing systems. *IEEE Transactions on Automatic Control*, **AC-35**(3), 289–298.

Kushner, H. (2001). *Heavy Traffic Analysis of Controlled Queueing and Communication Networks*. Springer.

Latouche, G. and Ramaswami, V. (1999). *Introduction to Matrix Analytic Methods in Stochastic Modeling*. PA:SIAM.

Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A. H. G., and Shmoys, D. B. (1993). Sequencing and scheduling, algorithms and complexity. *In S.C. Graves, A.H.G. Rinnooy Kan, and P. Zipkin, editors,* Logistics of Production and Inventory*, volume 4 of* Handbooks in Operations Research and Management Science*, Amsterdam. North Holland.*

Lemoine, A. (1978). Networks of queues, a survey of weak convergence results. *Management Science*, **24**(11), 1175–1193.

Levy, Y. and Yechiali, U. (1975). Utilization of idle time in an M/G/1 queueing system. *Management Science*, **22**(2), 202–211.

Maglaras, C. (1999). Dynamic scheduling in multiclass queueing networks: Stability under discrete-review policies. *Queueing Systems*, **31**(3), 171–206.

Maglaras, C. (2000). Discrete-review policies for scheduling stochastic networks: Trajectory tracking and fluid-scale asymptotic optimality. *Ann. Appl. Probab*, **10**(3), 897–929.

Meyn, S. P. (2001). Sequencing and routing in multiclass queueing networks part i: Feedback regulation. *SIAM Journal on Control and Optimization*, **40**(3), 741–776.

Meyn, S. P. (2003). Sequencing and routing in multiclass queueing networks part ii: Workload relaxations. *SIAM Journal on Control and Optimization*, **42**(1), 178–217.

Meyn, S. P. (2008). *Control Techniques for Complex Networks*. Cambridge University Press.

Meyn, S. P. and Down, D. (1994). Stability of generalized Jackson networks. *The Annals of Applied Probability*, **4**(1), 124–148.

Meyn, S. P. and Tweedie, R. (1993a). *Markov Chains and Stochastic Stability*. Springer-Verlag.

Meyn, S. P. and Tweedie, R. L. (1993b). Stability of Markovian processes II: Continuous-time processes and sampled chains. *Advances in Applied Probability*, **25**(3), 487–517.

Meyn, S. P. and Tweedie, R. L. (1993c). Stability of Markovian processes III: Foster-Lyapunov criteria for continuous-time processes. *Advances in Applied Probability*, **25**(3), 518–548.

Miltenburg, G. J. (1987). Variance of the number of units produced on a transfer line with buffer inventories during a period of length T. *Naval Research Logistics*, **34**, 811–822.

Mitchell, K. and van de Liefvoort, A. (2003). Approximation models of feed-forward G/G/1/N queueing networks with correlated arrivals. *Performance Evaluation*, **51**(2-4), 137–152.

Naryana, S. and Neuts, M. F. (1992). The first two moment matrices of the counts for the Markovian arrival process. *Stochastic Models*, **8**(3), 459–477.

Nazarathy, Y. (2001). *Evaluation of on-line scheduling rules for high volume job shop problems, a simulation study*. Master's thesis, The University of Haifa.

Nazarathy, Y. and Weiss, G. (2008a). The asymptotic variance rate of finite capacity birth-death queues. *Queueing Systems*, **59**(2), 135–156.

Nazarathy, Y. and Weiss, G. (2008b). Near optimal control of queueing networks over a finite time horizon. *Annals of Operations Research. To Appear.*

Nazarathy, Y. and Weiss, G. (2008c). Positive Harris recurrence and diffusion scale analysis of a push pull queueing network. *Proceedings of Valuetools 2008.*

Neuts, M. F. and Li, J. (2000). The input/output process of a queue. *Appl. Stochastic Models Bus. Ind.*, **16**, 11–21.

Parthasarathy, P. R. and Sudhesh, R. (2005). The overflow process from a state-dependent queue. *International Journal of Computer Mathematics*, **82**(9), 1073–1093.

Parzen, E. (1962). *Stochastic Processes*. Holden-Day.

Pourbabai, B. (1987). Approximation of the overflow process from a G/M/N/K queueing system. *Management Science*, **33**(7), 931–938.

Prabhu, N. (1998). *Stochastic Storage Processes: Queues, Insurance Risk, Dams, and Data Communication*. Springer.

Pullan, M. (1993). An algorithm for a class of continuous linear programs. *SIAM Journal on Control and Optimization*, **31**, 1558.

Reiman, M. (1984). Open queueing networks in heavy traffic. *Math. Oper. Res.*, **9**(3), 441–458.

Reynolds, J. F. (1975). The covariance structure of queues and related processes - a survey of recent work. *Adv. Appl. Prob.*, **7**, 383–415.

Rudemo, M. (1973). Point processes generated by transitions of markov chains. *Adv. Appl. Prob.*, **5**, 262–286.

Rybko, A. and Stolyar, A. (1992). Ergodicity of stochastic processes describing the operation of open queueing networks. *Problems of Information Transmission*, **28**(3), 3–26.

Sadre, R., Haverkort, B., and Ost, A. (1999). An efficient and accurate decomposition method for open finite and infinite buffer queueing networks. *Proceedings of the Third International Workshop on Numerical Solution of Markov Chains*, pages 1–20.

Serfozo, R. (1999). *Introduction to Stochastic Networks*. Springer.

Shah, S. and Nahrstedt, K. (2002). Predictive location-based qos routing in mobile ad hoc networks. *In Proceedings of IEEE International Conference on Communications*.

Sigman, K. (1990). The stability of open queueing networks. *Stoch. Proc. Applns*, **35**, 11–25.

Smith, W. (1955). Regenerative Stochastic Processes. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences (1934-1990)*, **232**(1188), 6–31.

Stroustrup, B. (2000). *The C++ Programming Language: Language, Library and Design Tutorial*. AT&T.

Tan, B. (1999). Variance of the output as a function of time: Production line dynamics. *European Journal of Operational Research*, **177**(3), 470–484.

Tan, B. (2000). Asymptotic variance rate of the output in production lines with finite buffers. *Annals of Operations Research*, **93**, 385–403.

Tassiulas, L. (1995). Adaptive back-pressure congestion control based on local information. *Automatic Control, IEEE Transactions on*, **40**(2), 236–250.

Taylor, L. and Williams, R. (1993). Existence and uniqueness of semimartingale reflecting Brownian motions in an orthant. *Probability Theory and Related Fields*, **96**, 283–317.

van Doorn, E. A. (1984). On the overflow process from a finite Markovian queue. *Performance Evaluation*, **4**, 233–240.

Walrand, J. (1988). *An Introduction to Queueing Networks*. Prentice Hall.

Wein, L. (1992). Scheduling networks of queues: heavy traffic analysis of a multistation network with controllable inputs. *Operations Research*, **40**(2), 312–334.

Weiss, G. (1999). Scheduling and control of manufacturing systems – a fluid approach. *Proceedings of the 37 Allerton Conference*, pages 577–586.

Weiss, G. (2004). Stability of a simple re-entrant line with infinite supply of work – the case of exponential processing times. *J. Oper. Res. Soc. Jpn.*, **47**(4), 304–313.

Weiss, G. (2005). Jackson networks with unlimited supply of work. *Journal of Applied Probability*, **42**(3), 879–882.

Weiss, G. (2008). A simplex based algorithm to solve separated continuous linear programs. *Mathematical Programming*, **115**, 151–198.

Whitt, W. (1982). Approximating a point process by a renewal process, I: Two basic methods. *Operations Research*, **30**(1), 125–147.

Whitt, W. (1983a). Performance of the queueing network analyzer. *Bell System Technical Journal*, **62**(9), 2817–2843.

Whitt, W. (1983b). The queueing network analyzer. *The Bell Systems Technical Journal*, **62**(9), 2779–2815.

Whitt, W. (1992). Asymptotic formulas for Markov processes with applications to simulation. *Operations Research*, **40**(2), 279–291.

Whitt, W. (1994). Towards better multi-class parametric-decomposition approximations for open queueing networks. *Annals of Operations Research*, **48**, 221–248.

Whitt, W. (1995). Variability Functions for Parametric-decomposition Approximations of Queueing Networks. *Management Science*, **41**, 1704–1715.

Whitt, W. (2002). *Stochastic Process Limits*. Springer New York.

Whitt, W. (2004). Heavy traffic limits for loss proportions in single-server queues. *Queueing Systems*, **46**, 507–536.

Williams, R. (1998). Diffusion approximations for open multiclass queueing networks: sufficient conditions involving state space collapse. *Queueing Systems*, **30**, 27–88.

Williams, R. J. (1992). Asymptotic variance parameters for the boundary local times of reflected Brownian motion on a compact interval. *Journal of Applied Probability*, **29**(4), 996–1002.

Wolff, R. (1989). *Stochastic Modeling and the Theory of Queues*. Prentice Hall.

Wolfram, S. (1999). *The Mathematica Book, version 4*. Wolfram Media.

APPENDIX A

THE PRONETSIM SIMULATION PACKAGE

In this short chapter we briefly describe a software package which we have developed and used for most of the simulation examples presented in this thesis. We call it PRONETSIM, which stands for *Processing Network Simulator*. The current version, described here is called V0.5 and is still preliminary[1]. The source, executable and example files of the current and future versions are available at the web-page:

`http://www.stat.haifa.ac.il/~yonin/PRONETSIM/pronetsim.html`

In its current form, PRONETSIM is *non user-friendly* and typically requires performing some minor code alteration if one has a specific simulation task in mind. This implies the user is required to edit the C++ code in some development environment and re-compile. Nevertheless, we hope that this "lack of packaging" will not deter interested researchers from considering to use this package because we believe that its requirements specification is broad enough to cover a rich array of queueing network research questions and its design is quite professional from a software engineering point of view.

The software is written in ANSI/ISO C++ (cf. Stroustrup (2000)) and is currently compiled using Microsoft Visual C++ (for MS-Windows operating systems). Compilation for other machines should be straight forward but has not been attempted. The design of PRONETSIM is a UML driven object oriented design and the implementation attempts to be as efficient as possible.

The end goal of PRONETSIM is to simulate processing networks of quite a general form. This implies a model that generalizes the stochastic processing networks suggested in Harrison (2000, 2002, 2003). The main generalizations are the inclusion of infinite virtual queues and finite buffers with overflows. Currently (in the preliminary version V0.5), only MCQN+IVQ are fully supported (and have been tested) along with the ability to have finite buffers with

[1] We intend to continue to develop this package and bring it to the maturity level, similar to another software package that we developed, the JSSP: `http://www.stat.haifa.ac.il/~yonin/thesis/jobshopsim/shopsim.html`.

overflows. The more general stochastic processing network model, which has been taken into account in the design, but not fully implemented and tested, is not restricted to a one to one matching between activities and classes as in the MCQN and is useful for implementing discretionary routing and other more general settings. In addition, the current implementation does not handle preemption with resume, only preemption with a re-start of the job. This distinction is irrelevant for exponential processing times.

In contrast to the quite general model that PRONETSIM is designed to handle, the state representation of PRONETSIM is quite slick and does not maintain a representation of individual jobs but just counts of the number of jobs in each class[2]. As a consequence, the software may not be used for the following: Simulation of some scheduling policies such as FIFO and PS, output analysis of sojourn time distributions, efficient simulation of many-server systems, simulation of fluid queues and simulation of policies that are dependent on actual workloads. In spite all of these weaknesses, PRONETSIM has been very useful for obtaining most of the simulation results presented in this thesis as well as some more results presented in Kopzon *et al.* (2008).

The software is run from a command line as follows:

```
pronetsim file_name
```

Here `file_name` stands for a name of the *input file* (may include a path) which specifies which simulation to perform along with some auxiliary parameters. As an output, PRONETSIM creates an output file whose name is controlled by parameters of the input file. The output file is a textual file that contains a single nested Mathematica style list (cf. Wolfram (1999)).

The continuation of this chapter is structured as follows: Section A.1 defines the simulation model. Section A.2 describes the input file. Section A.3 describes the structure of the list in the output file. Additional information, is at the web-page mentioned above.

A.1 Model Description

The PRONET simulator is a discrete event simulator that is designed to simulate controlled processing networks of discrete material flow and continuous time. The basic "physical" entities in the simulated world are *buffers*, *jobs* and *resources*. The actions that are performed are called *activities* and they basically move jobs out of some buffers and into others. Activities require resources to operate.

Buffers

There are 4 possible buffer types:

Source Buffers – These are IVQs. They have an infinite amount of jobs in them. The job state of each such buffer is represented by a non-positive integer. Whenever a job is removed, the job state is decremented.

[2]This can also be changed in future versions but it would require some alterations to the core of the software design.

Sink Buffers – These are the duals of the IVQs. Jobs are only added to these buffers and their job state is represented by a non-negative integer that is increased. They are used to model the outside of the network (they must be used for open-networks).

Standard Buffers – These are "plain old queues". They do not have a size limit and jobs are taken from them and put in them upon completion of activities.

Finite Buffers – These are queues of finite capacity. When an activity attempts to put a job in such a queue and it is full, the job moves to an alternative overflow buffer. Overflows are allowed to recurse: the alternative buffer may also be a finite buffer that happens to be full and another overflow occurs, and so forth.

Activities

Activities are the entities that "occur" while simulated time is progressing. They are the entities that drive the simulation. The following is associated with each activity:

Effected Buffers – The activity associates a positive or negative integer with each buffer in a subset of the buffers list (source buffers are only associated with negative integers and sink buffers are only associated with positive integers). When the activity is complete, the integer is added to the state of each buffer in the subset. Typically an activity will have -1 and $+1$ associated with buffers and this implies that upon completion of the activity a job moves from one buffer to the other.

Utilized Resources – The activity indicates which resource are needed for it's operation. There is currently no processor sharing allowed so when an activity is in operation it "grabs" it's associated resources and other activities that need those resources may not operate during that time.

Processing Duration – A sequence of real positive values is associated with the activity. This is typically achieved by associating a mean processing time along with a specification of the distribution (e.g. Exponential, Deterministic or data from a file). Note that the simulation allows for concurrent event completions. This typically occurs when deterministic processing times are used.

Note that when an activity begins it freezes jobs in its input buffers so that if an additional activity was to begin, it will not be able to work on those frozen jobs. This is as described in Dai and Lin (2005).

Resources

Resources don't really have an active role in the simulation, they rather act as "semaphores" for the activates - two activities that share a resource may not operate simultaneously. Sometimes an activity may be preempted and in doing so, the resources that the activity uses are released.

Policies

Scheduling policies or rules are the controls of the simulated network. The code for this rule specifies which activities to schedule, based on the network state or some other decision rules. Ultimately (in future versions of this software), the interface for the scheduling policies will be well defined so that they can be coded as plug-ins (i.e. almost as input to the simulation). Currently, the policies are implemented in the *Control Policies* module and if the user wishes to add an additional policy, that module should be changed. In a preemptive system, a policy may decide to preempt activities in addition to scheduling activities. In this case, an important detail is to code when the policy "wakes up" during the discrete event simulation. Typically, this is at times during which one or more activities finish, but alternatively it can be at other times which are specifically coded.

A.2 Input File

The input file is organized like a windows type INI file. This file format is typically used by applications running on Microsoft windows to specify parameters to application programs[3]. Several example input files are at the web-site. The file format is composed of lines of the form:

```
attribute = value
```

In each such line the user specifies the values that certain attributes should take. The ini file is thus simply a collection of lines with each line being an attribute specification. To make things slightly more organized, there are also lines that contain section headers of the form:

```
[section name]
```

These section names precede groups of lines and thus group attributes into categorizes. Table A.1 lists the sections.

Section Name	**Description**
`runs`	Specifies how many runs to perform, and the duration of each run.
`model`	Defines the processing network to be simulated: The resources, activities, buffers and their inter-connections.
`processing times`	Defines the mean processing times for each of the activities and specifies their distributions. Also defines how to obtain seeds for the random variable generation (for non-deterministic processing times)
`policy`	Defines the scheduling policy to be used, along with parameters for the policy if any.
`logging`	Specifies which information to record during a run and what to output.

Table A.1: Sections of PRONETSIM input file.

Tables A.2 – A.6 define all of the attributes that may be specified, grouped into sections. Some attributes are required and others are optional. If an optional attribute has a default value, it gets this value if it does not appear in the input file. Options that are to be inputed as lists should be specified as Mathematica style lists and nested lists (i.e, use '{' and '}').

[3]PRONETSIM is not restricted to Microsoft systems, INI files are pure ASCII files.

Attribute Name	Description
`num runs`	Contains a positive integer value that specifies how many runs to perform. Default value is 1.
`time horizon`	Contains a positive real value that specifies the time horizon for each run. As an alternative, may contain a list of length `num runs` which specifies the duration for each run.
`initial conditions`	Contains a list containing two lists of non-negative integer values. The first list specifies initial conditions for the standard buffers. The second list specifies initial conditions for the finite buffers. As an alternative, may contain a list of such initial conditions with the length of `num runs` to specify initial conditions for each run. Default value is 0 for initial conditions if not specified.

Table A.2: Attributes of "runs" section.

A.3 The Output File Format

The output file contains a Mathematica style list in which each entry represents a run. Each run is a list that is organized as follows:

```
{run_descriptor, run_stats, log}
```

`run_descriptor` specifies information regarding the run. This is useful if there are many runs that are output into some sort of database. In the current version, this field is not made fully generic and must be handled by changing the code.

`run_stats` is reserved to specify information such as the time it took to perform the simulation, exceptions, memory usage, etc. It currently only records the number of second the simulation took.

`log` contains the simulated output of the run and we now describe it in detail. It is a Mathematica style list that is composed of sublists. The structure is as follows:

```
{
   {"Realization",rlist},
   {"Histogram",histList},
   {"Means",meanList},
   {"Samples",sampleList}
}
```

The strings `"Realization"` etc. are outputed for readability and may be ignored if processing the output in Mathematica or another application program. `rlist` is a list that contains a full dump of the realization (or an empty list if the realization is not to be dumped). `histList` is a list that contains a histogram if specified. `meanList` is a list with two sublists. The first is the means of the standard buffers, the second is the means of the finite buffers. `sampleList` is a list that contains samples of the queue levels as specified by the `partial samples` attribute. There is always one sample which is a sample at the end of the simulation run.

Attribute Name	Description
predefined model	If not specified, a pre defined model is not assumed. If specified, may take one of the following values: MODEL_SINGLE_SERVER_QUEUE, MODEL_SINGLE_SERVER_FINITE_QUEUE, MODEL_3_BUFFER_RLINE, MODEL_3_BUFFER_INFINITE_RLINE, MODEL_5_BUFFER_RLINE, MODEL_KSRS_NETWORK, MODEL_PUSH_PULL_NETWORK, MODEL_SIMPLE_ROUTING, MODEL_SINGLE_SERVER_WITH_IVQ, MODEL_DECOUPLED_PUSH_PULL_NETWORK, MODEL_GUO_5_CLASS_INFINITE_RLINE. The details regarding these models are in the source files PredefinedModels.h and PredefinedModels.cpp. In version V0.5 the user must use a predefined model (i.e. To add a new model, the source code must be modified).
num sources num sinks num standard buffers num finite buffers num resources num activities activities of resource effects of activities finite buffer sizes overflow destinations	These fields are still not supported in V0.5. They are intended to be used to specify a non-predefined model.

Table A.3: Attributes of "model" section.

Attribute Name	Description
distributions	A list of distributions, 1 per activity, that may currently take one of these values: DETERMINSITIC_PROCESSING_TIME, EXPONENTIAL_PROCESSING_TIME, ERLANG_PROCESSING_TIME, UNIFORM_PROCESSING_TIME, HYPER_EXPONENTIAL_2_PROCESSING_TIME, INPUT_FILE_PROCESSING_TIME.
distribution parameters	A list of lists, 1 per activity. The entry in each list is a list that specifies the distribution parameters other than the mean. For deterministic or exponential distributions, leave an empty list ({}). For Erlang distributions specify the number of phases ($\{k\}$). For uniform distributions specify the width around the mean. The hyper exponential 2 distributions is a mixture of two exponentials so specify for it the mean of the first and second exponentials. If INPUT_FILE_PROCESSING_TIME is specified then the parameter is a name of a file that contains a Mathematica style list of processing times. These times are to be scaled by the mean.
means	A list of the means, 1 per activity.
seed	If this value is -1, then a time seed is used. It is assured that a different seed is used between runs. Otherwise, a list of seeds should be specified, 1 seed per simulation run.

Table A.4: Attributes of "processing times" section.

Attribute Name	Description
policy	Some of the policies that are currently implemented are: PP_QUEUE_BALANCING_POLICY, PP_FIXED_THRESHOLD_POLICY, KSRS_QUEUE_THRESHOLDS_POLICY, FIXED_PRIORITY_POLICY. Some of these will only work with certain models (e.g. Push-Pull or KSRS). The FIXED_PRIORITY_POLICY will work with any model. It sets an absolute priority ordering on the buffers (priority doesn't change based on state).
policy parameters	A list of parameters to be passed to the policy. This is policy dependent. An important case is the FIXED_PRIORITY_POLICY. In this case, pass a permutation of the activity indexes $1,\ldots,$num activities. The first activity in this permutation has highest priority and the last has lowest priority.
preemption type	Indicates if premption is allowed and the kind of premption. Possible values are: NO_PREEMPTION, PREEMPTION_RESUME, PREEMPTION_RESTART.

Table A.5: Attributes of "policy" section.

Attribute Name	Description
full dump	If set to true then every detail of the realization is recorded.
partial samples	A list of times during which queue state samples should be recorded.
histogram	If set to true, then a histogram of the queue states is recorded.
running means	If set to true, then running means of queue levels (for finite and standard) buffers are recorded.
output file	The name of the output file to use. If not specified the output is put in the subdirectory RunDB (must exist) under a unique file name (file name is based on time of first run).

Table A.6: Attributes of "logging" section.

אוגרים וריטואלים אינסופיים. אנו שומרים על יציבות הסטיות באמצעות הפעלה של מדיניות לחץ מקסימאלי. התוצאה המרכזית בהקשר זה מראה שמדיניות זו היא אסימפטוטית אופטימאלית כאשר מספר הפריטים אשר עוברים עיבוד גדל ביחד עם מהירות העיבוד.

כפי שתואר לעיל, הנושא השני בהקשר של בקרה דן ברשת דחוף ומשוך. רשת זו מאופיינית ע״י שני שרתים ושני סוגים של עבודות אשר מבוצעות על ידי השרתים בכווני זרימה הפוכים. לרשת זמני עיבוד סטוכסטיים בעלי התפלגויות כלליות. רשת זו דומה במבנה לרשת Kumar-Seidman, Rybko-Stolyar (KSRS). ההבדל הוא שלרשת KSRS, יש זרמי הגעה סטוכסטיים בעוד שלרשת דחוף ומשוך יש כמות עבודה לא מוגבלת ולכן ברשת דחוף ומשוך, כל אחד מהשרתים יכול לעבוד ללא הפסקה. בניגוד לרשת KSRS, ברשת דחוף ומשוך ניתן למצוא מדיניות אשר מנצלות את השרתים ללא הפסקה ועדיין אינן יוצרות גודש בתורים. אנו מראים שתהליך המרקוב המתאים לרשת זו הוא מתמיד חיובית הריס. אנליזה זו עושה שימוש בתוצאות הנוזלים של Dai (1995) ומכלילות אותם למקרה של תורים וריטואלים אינסופיים.

בהקשר של תהליך היציאה של תור פשוט מסוג לידה-מוות עם מרחב מצבים סופי, אנו מפתחים נוסחא עבור קצב השונות האיסמפטוטי מהצורה $\lambda^* + \sum v_i$. כאן λ^* הוא קצב היציאה ו - v_i הם ביטויים המבוססים על קצבי הלידה והמוות. אנו מראים שאם קצבי הלידה הינם לא עולים וקצבי המוות אינם לא יורדים עם גודל התור (כמצוי בהרבה מערכות תורים), אז הערכים של v_i הינם שליליים ממש ולכן אינדקס הפיזור הגבולי של תהליכי ספירה (Limiting Index of Dispersion of Counts) הינו קטן ממש מאחד. במקרה של תור M/M/1/K, הנוסחה אשר אנו מציגים מסתכמת לביטוי סגור אשר מראה תופעה מפתיעה: כאשר המערכת מאוזנת (קצב השרות וקצב הגעת הלקוחות שווים), אז אינדקס הפיזור הגבולי של תהליך היציאה הוא מינימאלי. המצב דומה עבור תור M/M/c/K, מערכת הפסד ארלנג (Erlang Loss System), וגם מספר תורים מסוג PH/PH/1/K. בכל המקרים הללו ישנה ירידה מודגשת של קצב השונות האסימפטוטי כאשר הפרמטרים של המערכת מאוזנים.

בחזרה לתהליכי היציאה של רשת דחוף ומשוך, אנו מתעניינים בקצב השונות האיסמטוטי וגם בקוואריאנס בין תהליכי היציאה. דרך אחת לחשב זאת היא ע״י קרוב דיפוזיה. בהקשר זה אנו מראים שהתורים מתאפסים כאשר לוקחים גבולות דיפוזיה ואנו מחשבים את הפרמטרים של תהליך בראוני גבולי של תהליכי היציאה. קרוב הדיפוזיה מראה ששני תהליכי היציאה הינם בעלי קורלציה שלילית גבוהה. מסקנה נוספת הנובעת מהחישוב היא העובדה שקצב השונות האיסמפטוטי אינו מושפע מהמדיניות כל עוד שהמדיניות מאפשרת ניצולת מלאה ותורים יציבים.

על בקרה של רשתות תורים
וקצב השונות האסימפטוטי של יציאות

יוני נצרתי

תקציר

בעבודה זו אנו מטפלים במספר נושאים הקשורים לבקרה של רשתות תורים ואנליזה של קצב השונות האסימפטוטי של תהליכי יציאה. בהקשר של בקרת רשתות תורים, אנו תחילה דנים בבעיה של בקרה אופטימאלית של רשת מרובת מחלקות על פני אופק זמן סופי ביחס לעלויות החזקה באוגרים. שיטת הבקרה והתוצאות אשר אנו מציגים פורסמו ב Nazarathy and Weiss (2008b). לאחר מכן אנו מנתחים את היציבות של רשת לדוגמא בעלת אוגרים וריטואלים אינסופיים (Infinite Virtual Queues) אשר אנו מכנים רשת דחוף ומשוך (Push-Pull). ניתן לבקר רשת זו באופן המאפשר לשרתים לעבוד ללא הפסקה תוך כדי שמירה על גודל חסום סטוכסטית של כמות העבודה באוגרים. ב Kopzon et al. (2008) בוצע ניתוח של רשת זו בהנחת זמני עיבוד חסרי זיכרון. כאן אנו מרחיבים את התוצאות לזמני עיבוד כללים. התוצאות מתבססות על מסגרת "יציבות באמצעות נוזלים" להוכחת התמדה חיובית הריס (Harris) של תהליכי מרקוב המתארים רשתות תורים. תוצאות אלו פורסמו ב Nazarathy and Weiss (2008c).

התבוננות בהתנהגות לדוגמא של תהליכי היציאה של רשת דחוף ומשוך הובילה אותנו לחקור את "רמת האקראיות" של תהליך היציאה של רשת זו. מדד ראשון מעניין בהקשר זה הוא קצב השונות האסימפטוטי: קצב גידול הלינארי של פונקצית השונות של תהליך ספירה על פני זמן. חישובים של מדד זה, הראו התנהגות מעניינת בתורים פשוטים מסוג לידה-מוות עם מרחב מצבים סופי. בהקשר זה אנו מציגים נוסחא סגורה לקצב השונות האסימפטוטי של תהליך היציאה. תוצאות אלו פורסמו ב Nazarathy and Weiss (2008a). לאחר הדיון בתורים פשוטים, אנו חוזרים לדון ברשת דחוף ומשוך ומציגים ביטויים עבור קצב השונות האסיפמטוטי באמצעות קירובי דיפוזיה העושים שימוש בתוצאת היציבות אשר הוזכרה קודם.

שיטת הבקרה אשר אנו מציגים עבור רשת תורים מרובת מחלקות על פני אופק זמן סופי משלבת מספר עקרונות: תוכניות לינאריות מופרדות רציפות (Separated Continuous Linear Programs), אוגרים וריטואלים אינסופיים, ומדיניות לחץ מקסימאלי (Maximum Pressure Policies). אנו מקרבים את רשת התורים באמצעות רשת נוזלים ומנסחים בעיית אופטימיזציה עבור הנוזלים. בעיה זו ניתנת לפתרון ע"י תוכנית ליניארית מופרדת רציפה. פתרון הנוזלים האופטימאלי, מחלק את אופק הזמן למקטעים בעלי קצב זרימת נוזלים קבוע. לאחר מכן אנו משתמשים במדיניות אשר מאפשרת לרשת התורים לעקוב אחר פתרון הנוזלים. בהקשר זה אנו ממדלים את הסטיות בין רשת התורים ורשת הנוזלים באמצעות מספר רשתות תורים בעלי

ברצוני להקדיש עבודה זו לזכרה של סבתה סופיה, האישה המתוקה שזרעה בי זרעים של חוזק, הן באמצעות האהבה הטהורה אשר סיפקה לי עד ימייה האחרונים והן בדרך עקיפה דרך גידולה המסור של אמי לאה ובאופן מסוים גם אבי משה, זוג ההורים הכי נהדרים שאפשר לדמיין, לפעמים כה נהדרים שאפילו קשה לדמיין.

סבתה שרדה באומץ את שואת היהודים של המאה ה 20, הצליחה בדרך נס לפגוש את סבא נחום האהוב, וביחד גידלו שתי בנות מופלאות אשר הביאו שבעה נכדים לעולם, כולם אנשים איכותיים וטובים אשר יישארו קרובים לליבי כל עוד הוא פועם. בנוסף, עד היום נולדו לסבתא שבעה נינים, את רובם אמנם לא פגשה, שתיים מהם הן אמילי וקיילי, ילדות הזהב אשר הגדירו בשבילי מחדש את המושג אהבה, כמשהו אבסולוטי וללא התניות, והרבה מכך תודות לאימם, אשת הברזל והפרחים שאני אוהב כל כך, כרמל.

סבתה לעולם לא פגשה את אמילי, נסיכת הנסיכות שלי, אשר נולדה מספר חודשים לאחר מותה, בתחילת תקופת הדוקטורט, וגם לא את קיילי בת השנה, כוכב הזהב המתוקה עלי האדמות. סבתא גם לא הייתה מאמינה שאסיים את הדוקטורט, ולאור מה שהכירה בעודה בחיים כנראה שצדקה, לא הייתי עושה זאת ללא הרוגע והנחת אשר קיבלתי מאמילי וקיילי ומכרמל וגם לא הייתי עושה זאת ללא מטריית העזרה רחבת ההיקף אשר קיבלתי מהורי טובי הלב ומאחי נדב ואחיותיי נעמה וענת.

אז יללה, לאחר כל הפוצי שמוצי הזה: החיים זה דבר קצר, צריך לתת גז עד ההקדשה הבאה.

על בקרה של רשתות תורים
וקצב השונות האסימפטוטי של יציאות

מאת: יוני נצרתי

בהדרכת: פרופסור גדעון וייס

חיבור לשם קבלת התואר "דוקטור לפילוסופיה"

אוניברסיטת חיפה
הפקולטה ל מדעי החברה
החוג ל סטטיסטיקה

This page is back of Hebrew cover – throw away

על בקרה של רשתות תורים
וקצב השונות האסימפטוטי של יציאות

יוני נצרתי

חיבור לשם קבלת התואר "דוקטור לפילוסופיה"

אוניברסיטת חיפה
הפקולטה למדעי החברה
החוג לסטטיסטיקה

נובמבר, 2008