CANONICAL APPROXIMATION IN THE PERFORMANCE ANALYSIS OF DISTRIBUTED SYSTEMS

Eugene Pinsky

CUCST-251-86

CANONICAL APPROXIMATION IN THE PERFORMANCE ANALYSIS OF DISTRIBUTED SYSTEMS

Eugene Pinsky

Submitted in partial fulfillment of the
requirements for the degree
of Doctor of Philosophy
in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY
1986

ABSTRACT

CANONICAL APPROXIMATION IN THE PERFORMANCE ANALYSIS OF DISTRIBUTED SYSTEMS

Eugene Pinsky

The problem of analyzing distributed systems arises in many areas of computer science, such as communication networks, distributed databases, packet radio networks, VLSI communications and switching mechanisms. Analysis of distributed systems is difficult since one must deal with many tightly-interacting components. The number of possible state configurations typically grows exponentially with the system size, making the exact analysis intractable even for relatively small systems.

For the stochastic models of these systems, whose steady-state probability is of the product form, many global performance measures of interest can be computed once one knows the normalization constant of the steady-state probability distribution. This constant, called the system partition function, is typically difficult to derive in closed form. The key difficulty in performance analysis of such models can be viewed as trying to derive a good approximation to the partition function or calculate it numerically.

In this Ph.D. work we introduce a new approximation technique to analyze a variety of such models of distributed systems. This technique, which we call the method of Canonical Approximation, is similar to that developed in statistical physics to compute the partition function. The new method gives a closed-form approximation of the partition function and of the global performance measures. It is computationally simple with complexity independent of the system size, gives an

excellent degree of precision for large systems, and is applicable to a wide variety of problems. The method is applied to the analysis of multihop packet radio networks, locking schemes in database systems, closed queueing networks, and interconnection networks.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

I sincerely thank my academic and thesis advisor, Professor Yechiam Yemini for his guidance, interest, encouragement and criticism throughout my graduate studies at Columbia. I would also like to express my sincere gratitude to the members of my Ph.D. thesis committee: Professor Michael Foster of Columbia University, Professor Zvi Galil of Columbia University, Dr. Steven Lavenberg of IBM T.J. Watson Research Center, Professor R. Mazumdar of Columbia University, Dr. Debasis Mitra of Bell Labs, Professor Mischa Schwarz of Columbia University and Professor Moshe Sidi of the Technion for many helpful comments, discussions and suggestions.

I am grateful to all the faculty, students and staff in the Department of Computer Science for their encouragement and support. My special thanks to my friends, especially Galina Datskovsky, Mark Moerdler, Moti Yung, Kevin Mathews, Cecile Paris, Bruce Abramson and others for their friendship and support.

Most of all, I would like to thank my family for their constant encouragement, support and love which has made this work possible.

The research leading to this thesis was supported in part by an IBM Fellowship, by the Department of Defense Advanced Research Project Agency under contract N00039-84-C-0165, the National Science Foundation, under contract MCS-81-10319, and the New York State Center for Advanced Technology, under contract CAT(83)-8.

To my family

with love

CHAPTER 1

INTRODUCTION

1.1. MOTIVATION

This work focuses on the area of performance evaluation of distributed systems. Such systems include multiprocessor interconnection networks, distributed database systems, closed queueing networks, packet radio networks and other systems. The significance of these systems and of their analysis is best summarized by P. Heidelberger and S. Lavenberg ([Heid84]):

"... Analytic performance modeling has been an extremely useful tool for evaluating the performance of computing systems of the 1970's and early 1980's. However, computing systems are rapidly advancing and analytic modeling techniques must advance with them in order to maintain relevance in the late 1980's and into the 1990's. Distributed processing, parallel processing, and radically new computer architectures present significant modeling challenges and opportunities. Distributed processing systems will become commonplace. These systems will serve large numbers of users, consist of many devices and will incorporate large databases, both centralized and distributed. High levels of performance will be the key .... Further work is needed in database modeling .... Particular attention needs to be paid to modeling distributed databases .... Parallel processing systems will also become widespread as hardware costs continue to decline .... There have been very few techniques developed to analyze such systems and much work remains to be done. In meeting these challenges approximations will play a key role. If possible, computable error bounds need to be developed .... A consistent and comprehensive framework for validating approximations needs to be developed and applied."

Thus, the problem of analyzing distributed systems is of fundamental im-

portance to the whole field of Computer Science. Major new advances in this area are required. This work provides a step in that direction.

1.2. THE PROBLEM

Consider a system of distributed agents sharing a distributed resource. It is usually the case that there are more agents than resources. Therefore, *interference* among agents is *inevitable*. For example, in database systems, transactions interfere with each other by locking subsets of the items. In multiprocessor interconnection networks, an established connection may block (and thus interfere with) other connections. In packet radio networks, transmissions share the broadcast medium and thus may interfere with one another if in progress at the same time.

Given the statistics of generation and duration of service requests, we would like to compute a number of performance measures, such as average concurrency, utilization and throughput. In addition, we would like to be able to show the dependence of these measures on system parameters (load, size, etc). If such dependence can be established analytically, we can solve problems involving optimum design (for example, achieving optimal load to maximize certain performances). Among the key performance measures are the following:

- A measure of average concurrency. For multiprocessor interconnection networks this is the bandwidth - average number of accessed memory modules. For database systems it is the average number of transactions in the system. For packet radio networks it is the capacity, i.e. the maximum number of successful transmissions per link.

- Non-Blocking Probability - the probability that a new activity is not blocked.

- Distribution of traffic and interference. The operation of a computer system can exhibit "interference locality". For example, in multiprocessor systems a processor

usually requests access to a particular (so-called "favorite") memory module. How can we model this phenomenon of "interference locality" and show its influence on global performance measures ?

The existence of interference causes two difficulties for the performance analysis of these systems. Firstly, since only certain configurations of concurrent activities are feasible (e.g. certain transmissions can coexist concurrently in a packet radio network) there is a problem of adequately describing the possible concurrent states. Secondly, the systems usually cannot be decomposed into independent, easy-to-analyze components. These two problems - *combinatorics of concurrency* and *tight-coupling* make the analysis of interference systems difficult.

In this thesis, we consider distributed systems modelled by time-reversible Markov chains ([Kell80]). For such models, the steady-state probability distribution is of the product form. This means that if a state of a system is described by k random variables $n_1, \ldots, n_k$, then the steady-state probability distribution $P(n_1, \ldots, n_k)$ is given by $H_1(n_1) \cdots H_k(n_k)/Z_N$. Here Z_N is the normalization constant Z_N such that all probabilities $P(n_1, \ldots, n_k)$ sum up to 1. This constant Z_N is called the system *partition* function ([Mitr84, Yemi83]). The number N is a measure of the system size, such as the size of the database, etc.

For these product-form solution models, a number of performance measures (e.g. average concurrency, throughput, utilization, non-blocking probability) can be computed once we know the partition function Z_N. This partition function is a generating function, with one term for each permissible concurrency level, and each term being given a weight related to the combinatorics of concurrency of the corresponding configurations. In other words, the partition function specifies how the possible configurations are partitioned among the different concurrency levels.

Typically, the partition function Z_N cannot be computed in closed form except for some simple cases. Therefore, the key difficulty in the performance analysis of these systems can be viewed as trying to derive a good approximation

to Z_N or to calculate it numerically.

It is interesting to note an analogy here to statistical physics. The main theory of statistical mechanics states that the global performance measures of a physical system are computed once we know its partition function. The partition function of a physical system specifies how the system configurations are distributed among different energy levels. It can be shown from this analogy that concurrency is analogous to energy, load is a "measure" of temperature, etc. One can even introduce a new performance measure of "pressure" which measures an average rate of blocking experienced by activities in progress ([Yemi83]).

1.3. THIS WORK

This thesis presents an approximation method to evaluate the system partition function Z_N in a closed-form. The basic idea is as follows:

To calculate the partition function Z_N for a system of size N, compute its generating function $Z_G(t)$:

$$Z_G(t) = \sum_{N=0}^{\infty} Z_N t^N$$

For many product-form solution models, it is often easier to find $Z_G(t)$ in closed form than it is to find the partition function. This is due to the fact that typically one establishes a recursive relation among the Z_N. Such a relation usually yields an algebraic equation for $Z_G(t)$. The partition function Z_N can be approximated from $Z_G(t)$ by the saddle point approximation ([Cops65]). The method is similar to that used in statistical physics to show the equivalence of canonical and grand canonical ensembles ([Path84]). This equivalence allows us to use the method to compute the average performance measures using the obtained

approximation of the partition function (see Appendix 2). By analogy with physics we will call $Z_G(t)$ the *grand partition function* and the method itself Canonical Approximation.

Using canonical approximation, we can show that the partition function admits the following representation:

$$Z_N = \frac{F(t_N)}{t_N^{N+1}}[1 + \epsilon_N]$$

where relative error ϵ_N is typically of order $O(1/N)$. In the above expression, t_N is the smallest (positive) point at which the derivative (with respect to t) of $\log[Z_G(t)/t^{N+1}]$ vanishes†, and $F(t_N)$ is related to the second derivative of the grand partition function at t_N.

The computational and space complexity‡ of evaluating t_N and $F(t_N)$ is *independent* of N. From this representation of the partition function and the concept of ensemble equivalence (see Appendix 2), we will be able to derive the closed-form approximations for a number of performance measures that do not require an explicit calculation of Z_N. This is very important in practice: the explicit calculation of Z_N may lead to very unstable algorithms (e.g. overflow), since Z_N can be a very large number (outside of a floating point range of a computer) even for moderate values of N. Canonical approximation, on the other hand, allows the design of *stable* algorithms to compute these measures for practically any system size N. The computational and space complexity of computing the performance measures is *independent* of N, whereas the precision increases with N.

Using canonical approximation, we solve the following classes of problems:

---

† It can be shown (see Chapter 3) that t_N is the saddle point of the surface $f(t) = \log[Z_G(t)/t^{N+1}]$ drawn in the complex plane.

‡ In this thesis, we assume the real number model of computation with the cost of an arithmetic operation to be unity. This is the standard model of computational complexity for scientific calculations.

• Multihop packet radio networks. Although a general markovian model of multihop packet radio networks is available ([Boor80, Toba82]), the computation of performance measures (primarily, the nodal throughput) has been done only for very small (the number of nodes $N = 10$) networks due to the complexity of the numerical algorithms involved. In this work, several networks operating under CSMA and C-BTMA are analyzed and compared using canonical approximation. The closed form expressions are obtained in terms of a root of a simple polynomial equation. Not only is it possible to write down explicit formulae for the nodal throughput for any network size N, but it is also possible to identify (analytically) the loads when a particular multi-access protocol gives a better performance. Moreover, we introduce a method to compute the approximation of the nodal throughput assuming the zero capture. The only known methods for the analysis of these under zero capture are simulation ([Toba85]) or explicit construction of the underlying state space ([Braz85]), which is computationally intractable unless N is very small.

• Locking Models of Database Systems. The evaluation of global performance measures (e.g. average concurrency and non-blocking probability for each transaction class) for a database of N items and p classes using static locking ([Mitr84, Lave84]) is reduced to solving a simple polynomial equation. Simple closed-form expressions for global performance measures are obtained for any range of parameters. This is in contrast with the previously known algorithms, which require an explicit computation of the partition function and whose computational complexity is of order $O(Np)$ ([Mitr84]). The previous asymptotic ($N \mapsto \infty$) analysis ([Lave84, Mitr84]) requires negligible computation but assumes very low traffic. We will show that in heavy traffic, there is a a range of load parameters beyond which the database will not have any large transactions. This phenomenon will be explained using the analogy from the condensation of an imperfect gas.

• Single Class Closed Queueing Networks. The computation of performance measures (e.g. average queue length, utilization, and average delay at each node) for a single-class markovian network with M nodes and N customers is usually done

by iterative numerical algorithms, whose computational and space complexity increases with N and M ([Buze73, Reis79]). Canonical approximation gives simple closed-form expressions for these performance measures. The computational complexity of the new algorithms, based on these expressions, is of order $O(M)$. There are no additional storage requirements. These algorithms are numerically stable and convergent with high accuracy even for very moderate values of M and N. The asymptotic analysis ($N \mapsto \infty$) requires negligible computation and can be implemented on a pocket calculator.

• Interconnection Networks. The method is applied to analyze the crossbar interconnection network for a wideband digital switch. The previous analysis of the crossbar has been done in the context of multiprocessor systems ([Bhan75, Bhuy83, Pate81, Stre70]). For these systems, the crossbar interconnection network operates in the synchronous mode and interference is caused only when there are two or more connection requests to the same output line (i.e. only memory interference). The crossbar networks for a wideband switch, analyzed in this thesis, operate in asynchronous packet switching mode. Unlike the previous models of multiprocessor crossbar networks, there is an additional interference when there are two or more requests to the same input line, making it much more difficult to analyze. We will consider such a model with different classes of arrivals requests, characterized by different statistics of arrivals, service times and access patterns. Both "equally likely" and "favorite output" access patterns are analyzed. Canonical approximation gives simple closed-form expressions for the average bandwidth and non-blocking probability for each class of arrival requests. The computation of these performance measures is reduced to solving a simple cubic equation.

The main contribution of the present work is the introduction of a computational method to with the following characteristics:

- Closed-form Approximation
- Simple Computationally

- Complexity Independent of the System Size
- Wide Applicability
- Excellent Degree of Precision for Large-Scale Systems

1.4. ORGANIZATION OF THIS WORK

This thesis consists of 8 chapters and 2 appendices. For the sake of clarity, some of the proofs are given in the last sections (chapter appendices) of the corresponding chapters. All of the theorems and lemmas are ended with a "blackslug" sign ∎.

Chapter 2 presents a unifying survey of some performance analysis problems arising in distributed systems. The survey will focus on the interrelationships between concurrency and interference of processes. Problems to be considered include: analysis of database locking protocols, analysis of multihop packet radio networks, and analysis of multiprocessor interconnection networks.

Chapter 3 presents a model capturing the interference phenomena in a variety of distributed systems. After presenting the modeling assumptions, we write the general form of the solution. From the general form of the solution we can draw an interesting analogy to statistical physics. Details of this analogy can be found in the appendix. We then give a rigorous derivation of the canonical approximation method and the derivation of the relative error. The method is reduced to calculating the saddle-point (if $Z_G(t)$ is entire) or the smallest pole (if $Z_G(t)$ is meromorphic). The complexity is ***independent*** of the system size under consideration whereas the accuracy increases for larger size. The chapter concludes with an analysis of a classical machine interference model using the canonical approximation method.

Chapter 4 presents the application of the canonical approximation method to analyze the performance of some multiple access protocols for multihop packet radio networks. The general model of packet radio networks developed by Boorstyn and Tobagi ([Boor80, Toba82]) is used. Such a model can capture a variety of multiple access protocols. However, so far only very small networks have been analytically tractable. In this chapter we analyze linear array multihop packet radio networks operating under CSMA with perfect capture and C-BTMA. Not only are we able to compute the capacities but we can also identify (analytically) the loads for which a particular protocol should be used. We also introduce a new approximation method to analyze these systems under zero capture. This is important since the only existing methods today are simulation ([Toba85]) or an explicit construction of an underlying Markov chain.

Chapter 5 presents the analysis of static locking policies for a database system ([Lave84, Mitr84]). For a database of N items, p classes of transactions (here a class consists of all the transactions which require locks on the same number of items) each with its own statistics of generation and duration of access requests (static locking model of a database), canonical approximation is used to get closed-form expressions for the average concurrency of each transaction class, the non-blocking probabilities, and other database performance measures. The computation is based on finding the smallest root of a simple polynomial. The obtained closed-form solution is thus independent of N. This contrasts with the known algorithm ([Mitr84]) which is of $O(Np)$. The previous asymptotic analysis for these systems ([Lave84, Mitr84]) was done under the assumption of low traffic. We analyze the behavior of the database in "heavy" traffic and show that there is a range of load parameters beyond which large transactions would not be present. This is similar to that of a condensation and is explained using an analogy to imperfect gas theory.

Chapter 6 presents the analysis of single-class closed queueing networks. The method is numerically stable and computes the closed-form expressions for the average performance measures without an explicit calculation of Z_N. The problem

is reduced to solving a simple algebraic equation. It is shown to provide a very good approximation even for very small networks, whereas its complexity is independent of the number of customers. This in contrast to some of the other methods based on the convolution ([Buze73]) or mean-value analysis ([Reis79]), which are typically iterative and linear (complexity of solution) in the number of customers. Both load-dependent and load-independent cases are analyzed. For the network with only load-independent servers and a large number of customers, a new "pocket calculator" program to compute the performance measures is presented.

Chapter 7 presents the analysis of the crossbar interconnection networks. The analysis is presented under the assumptions of asynchronous packet switching, interference at the inputs and outputs, and possibility of favorite memory requests. The model is motivated by interconnection networks to be used in future wideband communication systems and to operate differently from those used in multiprocessor systems. The computation of global averages does not explicitly require the computation of the partition function. It is reduced to solving a simple quadratic or cubic equation.

Chapter 8 summarizes and concludes this thesis and discusses topics for further research.

Appendix 1 presents the analogy between an interference model introduced in Chapter 3 and a rigid-sphere model of a lattice gas in statistical physics.

Appendix 2 presents the idea of ensemble equivalence in statistical mechanics. This equivalence suggested the introduction of a method of approximation in statistical physics. The canonical approximation introduced in this thesis is motivated by that method.

CHAPTER 2

A SURVEY OF RELATED WORK

2.1. INTRODUCTION

The objective of this chapter is to provide a survey of performance analysis problems arising in distributed systems. Problems to be considered include: analysis of database locking protocols, analysis of multihop packet radio networks, and analysis of multiprocessor interconnection switches. This chapter focuses on the interrelationships between concurrency and interference of processes in such systems.

Consider a set of distributed agents sharing a common resource. At any moment of time, a given agent is either idle or busy utilizing the resource. Concurrent access to a shared resource is typically constrained by possible contention. In database systems, for example, agents are transactions contending over the lockable items. In multiprocessor interconnection networks, agents are connection requests contending over the shared switches. In packet radio networks, agents are transmissions contending over the shared broadcast medium. A key advantage of a distributed system over a centralized one is the performance gained by concurrency of activities. Interference determines the level of concurrency that a system can achieve. Understanding interference and its impact on performance thus plays a key role in the design of an efficient distributed system.

In analyzing distributed systems one needs to distinguish the synchronous mode and asynchronous mode of operation. Under the synchronous mode of operation, time is divided into slots. An agent may become active only at the beginning of a time slot and is typically served during the slot. Under the asynchronous mode of operation, an agent may become active at any time. In this case, one usually

assumes Poisson distribution of arrivals. The service time is random and is usually assumed to be distributed exponentially.

The dynamics of agents in the systems above can be described as follows: An agent becomes active and tries to access the resource. An acquisition protocol is used to determine the conditions under which the agent can obtain the resource. Access can be denied because the resource is already used by another interfering agent. For example, in database systems, an arriving transaction cannot proceed if some of its requested items are exclusively locked by some other transactions. In packet radio networks, a node senses that the shared channel is busy and thus knows that it might interfere if it starts transmitting. If an agent is blocked, it retries later. If the agent is allowed to proceed, it uses the resource for some time. When the agent holds the resource, it may try to minimize the potential interference from other agents. For example, with the help of a busy tone signal, it may warn potentially interfering agents to remain idle. After using the resource, the agent becomes idle.

The acquisition protocol regulates interference among contending agents. It is thus a key element in accomplishing greater concurrency and with it, improved performance. Therefore, the design of the acquisition protocol is key to accomplishing effective performance of a distributed resource sharing system.

From the statistics of generation and duration of service requests and the acquisition protocol, one would like to

(a) compute a number of performance measures such as the average concurrency and blocking probability.

(b) determine the dependency of these measures on system parameters (load, system size).

Moreover, one would like to be able to compare different acquisition protocols and choose the one optimizing the concurrency. Unfortunately, the complex

combinatorics of admissible concurrent activities makes the analysis of interference difficult even for very small systems.

Whenever there are more tasks than resources, interference and hence queues are inevitable. It is therefore natural to model these systems as a network of interconnected queues. Such a model typically provides a conceptually and computationally appealing mechanism to capture interaction among contending agents. Queueing networks models are used to calculate a number of performance measures. Key advances in performance analysis can be seen as breakthroughs in queueing theory. For a review of computational algorithms for queueing networks, see [Heid84, Brue80]. We should just mention that there is a broad class of models called product-form (also called separable or decomposable) solution networks. Networks in this class are characterized by the so-called BMCP theorem ([Bask75]). Efficient computational algorithms are known only for separable networks. It is now generally believed ([Heid84]) that the class of product-form solution networks will not be significantly extended beyond the BMCP networks.

2.2. MULTIPROCESSOR INTERCONNECTION NETWORKS

2.2.1.GENERAL OVERVIEW AND PERFORMANCE ISSUES

The first class of systems to be considered in this chapter is Multiprocessor Interconnection Networks. These networks provide communication between processors and memory modules in multi-processor computer systems (Figure 2.2.1.1). Research towards supercomputer and fault-tolerant systems has spurred interest in high performance multiprocessor systems, requiring a high capacity interconnection network.

An interconnection network cannot typically provide a dedicated link between any processor-memory or processor-processor pair. This is prohibitively ex-

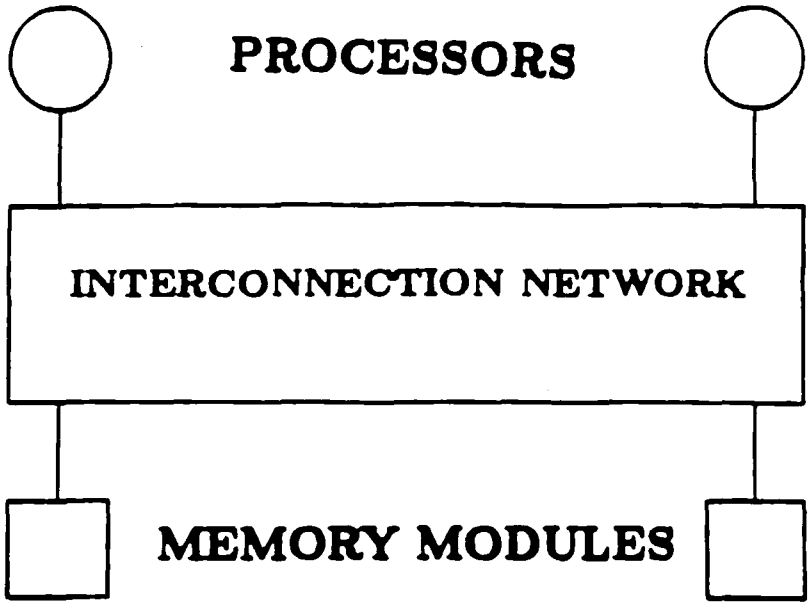

Figure 2.2.1.1: Multiprocessor Interconnection Network.

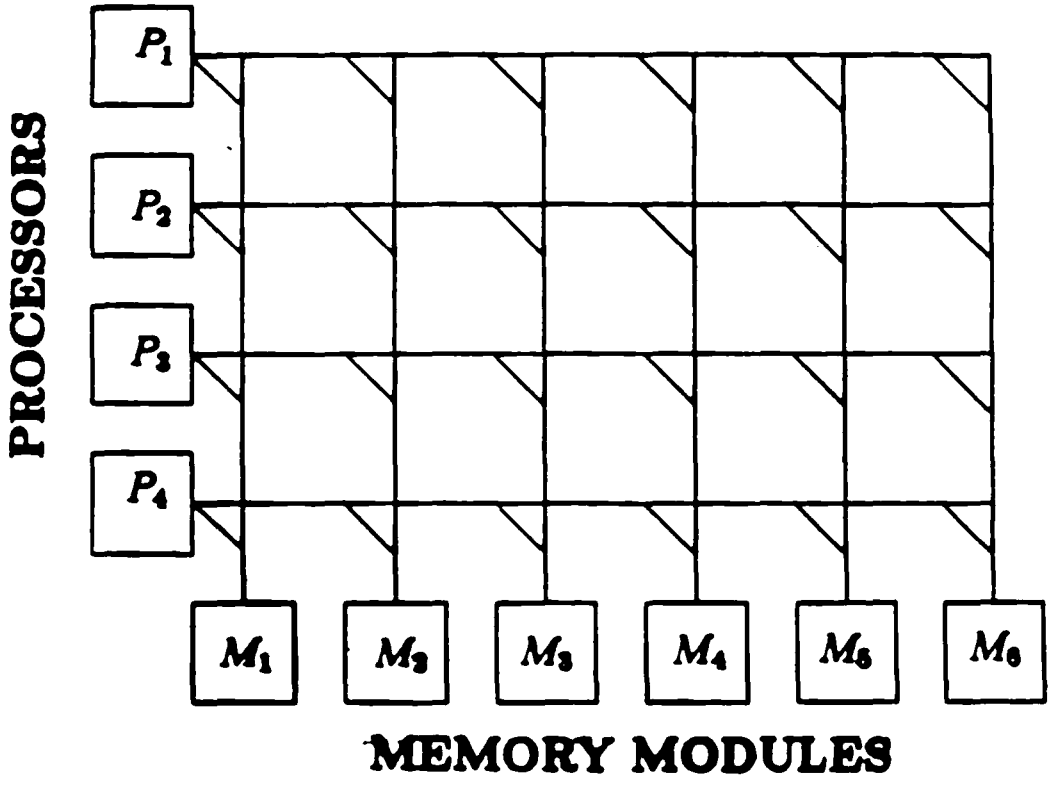

Figure 2.2.1.2: 4 × 6 Crossbar Interconnection Network.

pensive even for modest-size networks. Therefore, an established communication path may block some other paths. For example, consider an 4×6 crossbar interconnection network (Figure 2.2.1.2). This network consists of 24 switching elements and allows all possible one-to-one and simultaneous connections between all 4 processors and all 6 memory modules. When two or more processors try to access the same memory, only one of them will be connected and the rest will be blocked or rejected.

Interconnection networks may be classified into two types: non-blocking and blocking. In a blocking network, interference can arise from simultaneous requests to use a common path or access the same destination. In such a case, interference occurs within a roùting switch. Examples of blocking networks include shuffle-exchange, delta, and banyan, described later in this chapter. In a non-blocking network, interference can arise only from simultaneous requests to the same memory module, but not from an attempt to use any one switch in a network. This type of interference is usually referred to as the "memory interference". An example of such a network is a crossbar switch.

What are the key performance measures in analyzing the impact of interference in multiprocessor interconnection networks ?

- The most important performance measure is the Bandwidth (BW) - the expected number of memory requests accepted per memory cycle.
- The average delay D to access a memory module and get the required data.
- The probability P that a request generated by a processor is blocked.
- The distribution of traffic and interference throughout the system. In a practical situation, a processor is likely to address a particular memory module most of the time, except when an interprocessor communication is necessary. If processor P_i communicates more often with memory module M_i, one calls M_i a favorite memory of processor of P_i. In such a case, interference is not uniform throughout the system.

How does the phenomena of "interference locality" affect system performance ?

There are several different physical forms available for multiprocessor interconnection networks. A good survey is given in [Thur82]. For the purposes of this discussion, one can divide them into three groups: crossbar interconnection networks, bus interconnection networks, and multi-stage interconnection networks.

2.2.2. CROSSBAR INTERCONNECTION NETWORKS

Crossbar switch has been analyzed more extensively than other interconnection networks ([Bhan73, Bhan75, Bhan77, Bhuyan83, Pate81, Stre70]). Perhaps the first analytic results were reported by Bhandarkar ([Bhan73]) who analyzed an $N \times M$ crossbar switch under the assumption that service times are exponentially distributed. However, most memory systems do not have an exponentially distributed cycle time, although techniques such as interleaving, cache memories, and read-modify-write memory access suggest that this exponential assumption may be a good approximation. The generation of requests is uniform, in that a processor is equally likely to request connection to any memory module. The crossbar is modelled as a queueing network with M nodes. A processor is a customer that is either receiving service (accessing some memory module) or queued at some node (queued to access some memory module). A state of the network is specified by the number of queued requests for each memory module. With the above assumptions, it is shown that the crossbar can then be described as a closed Jackson network with M servers and N customers circulating with uniform routing probabilities $\frac{1}{M}$. The steady state distribution

$$P(\text{any state}) = \left[\binom{N+M-1}{M-1}\right]^{-1} \qquad (2.2.2.1)$$

indicates that all states have the same probability. If one sets $N = M$ then the

bandwidth is asymptotically

$$E(\text{busy processors}) \mapsto \frac{N}{2}, \qquad N \gg 1 \tag{2.2.2.2}$$

Thus, no matter how many processors are used, one can expect half of them to be active.

In the above analysis, it was assumed that the cycle time is exponentially distributed. However, for real systems it is more realistic to assume that the cycle time is fixed. If the service time is fixed, the crossbar can be analyzed by a simple Markov Chain analysis ([Bhan75, Bhan77]). The analysis of a crossbar network with N processors and M memory modules is similar to the analysis of an occupancy problem of N balls and M urns ([Fell66]). Processor requests (balls) are assigned to the M memories (urns) at the beginning of each memory cycle. At the end of the cycle one processor request (ball) is removed from each memory (urn). If there are k memories with queued processor requests (k non-empty urns) during cycle C, then k new processor requests (balls) are available for assignment for the $(C+1)$-th cycle.

Under these assumptions, the number of distinct states is given by the number of ways in which N requests (balls) can be distributed among M memories (urns) and is given by $\binom{N+M-1}{M-1}$. Unlike the previous analysis, not all states are equally likely. Assuming that all processors behave identically, one can show that a number of distinct states are equivalent. For example, for a 3×2 crossbar, if the state $(2,1)$ means two connection requests to first memory module and one request to the second memory module, then the states $(2,1)$ and $(1,2)$ are equivalent. The problem can be described by reduced states (all equivalent states are represented by one aggregated state). Using this reduction in the number of states, a computational algorithm is presented. The algorithm computes the reduced states, generates the transition probabilities for each reduced state and then computes the bandwidth. However, despite the reduction, the number of reduced states corresponds to the number of different ways in which the number N can be partitioned into M parts.

This number (for $N \leq M$) is asymptotic to $\frac{1}{4\pi\sqrt{3}}\exp\left[\sqrt{\frac{2N\pi}{3}}\right]$. The number of reduced states is exponential in N, and thus the solution method is intractable, except for very small N.

Strecker ([Stre70]) has analyzed the crossbar under the assumption of removing the queued processors from all the memory modules and reassigning them randomly for the next cycle. The state of the system at any cycle can be considered independent of that in the previous cycle. This is unlike the previous analysis ([Bhan75, Bhan77]) when only those processors which received service in the previous cycle are reassigned for the next cycle. Although such an assumption may seems unrealistic, the results obtained by Strecker for the crossbar suggest that this is a good approximation. The analytic results are within 8% of the exact Markov chain model of Bhandarkar ([Bhan75, Bhan77]). More important, such an assumption makes the analysis very easy. The distribution of processors accessing the memory modules is given by a binomial distribution: for an $N \times M$ crossbar, the probability $P(i)$ that there are i requests in a unit cycle is

$$P(i) = \binom{N}{i}\left(\frac{1}{M}\right)^{i}\left(1-\frac{1}{M}\right)^{N-i} \qquad (2.2.2.3)$$

From this, one can easily derive the expected number of busy memory modules to be

$$BW = M[1-\left(1-\frac{1}{M}\right)^{N}] \rightarrowtail M(1-\exp(-\frac{M}{N})) \qquad (2.2.2.4)$$

In particular, if $M = N$ the bandwidth $BW \approx 0.63N$. The bandwidth is linear in N. It is interesting to note that this expression is similar to the expression (2.2.2.2) for the crossbar bandwidth $BW \approx 0.5N$ under the assumption of exponential service time.

In the above analyses, it is assumed that a processor generates an access request at the beginning of each memory cycle. It is more realistic to assume, however, that a processor generates an access request at the beginning of a cycle with some probability. Such a model was analyzed by Patel ([Pate81]), who used

the assumptions of Strecker, but with the assumption of the fixed probability m that a processor generates a request during a cycle. The following expressions for bandwidth (BW) and non-blocking probability P were obtained:

$$BW = M - M\left(1 - \frac{m}{M}\right)^N \mapsto M(1 - e^{-\frac{mM}{N}}) \qquad (2.2.2.5)$$

$$P = \frac{M}{mN} - \frac{M}{mN}\left(1 - \frac{m}{M}\right)^N \mapsto \frac{M}{mN}(1 - e^{-\frac{mN}{N}}) \qquad (2.2.2.6)$$

Note that $m = 1$ corresponds to the above results of Strecker ([Stre70]). The above asymptotic approximations are good within 1% of the simulation results when $M, N > 30$ and within 5% when $M, N > 8$. Note that for a fixed $\frac{N}{M}$ the bandwidth increases linearly.

Bhuyan and Lee ([Bhuy83]) have extended the above analysis of an $N \times M$ crossbar to show the impact of "interference locality". In real systems, a processor accesses a particular memory module most of the time, with occasional requests to other memory modules. To capture this phenomena in the model, the processors are divided into two groups (it is assumed that $N > M$). Group A consists of N processors each of which accesses its "favorite" memory module with probability x. The remaining $N - M$ processors in group B are equally likely to address any of the M memory modules with probability $\frac{1}{M}$. If x is the probability of favorite memory access, then the bandwidth is shown to be given by

$$BW = M\left[1 - (1 - xm)\left(1 - m\frac{1-x}{M-1}\right)^{M-1}\left(1 - \frac{m}{M}\right)^{N-M}\right] \qquad (2.2.2.7)$$

Note that if $x = \frac{1}{M}$

$$BW = M\left[1 - \left(1 - \frac{m}{M}\right)^N\right] \qquad (2.2.2.8)$$

which is identical to equation (2.2.2.3) for an equally likely case. Bandwidth comparison is given in Figure 2.2.2.1. It is not surprising to note that the bandwidth

for a favorite memory case is higher than for an equally likely case: most of the time the processors will try to access their favorite memory modules. As a result, there is less interference, which in turn gives a higher bandwidth.

Crossbar switches for the connection of many processors and many memory modules are becoming less and less attractive due to their complexity (e.g. number of switching elements) and high cost, as compared to the cost of processors and memory modules. In fact, for a large crossbar multiprocessor system, the crossbar interconnection network would probably cost more than the rest of the system combined. Moreover, the bandwidth provided by such an interconnection network exceeds the requirements of most applications. To circumvent the high cost of crossbar interconnection networks, some "loosely coupled" systems have been proposed. In these systems, sharing of the main memory is somewhat restricted: some memory accesses may be fast and direct while others may be slow, indirect and even involve the operating system intervention. The interconnection networks for these "loosely coupled" systems are either multi-bus or multi-stage interconnection networks.

2.2.3. BUS INTERCONNECTION NETWORKS

Let us start by a simple example of a multi-bus architecture as given in Figure 2.2.3.1. In such a system a processor P_i uses its private memory PM_i most of the time and occasionally accesses the global memory modules GM_i via the system global busses.

Interference arises in such multi-bus systems because of the processor's contention for both memory modules and busses. In a system with N processors, M memories and b busses, if the number of busses $b \geq \min(N, M)$, then interference is caused by the sharing of memory modules. In such a system, a processor can always find a bus to access a free memory. Note that such a system is different from a crossbar switch. In a crossbar switch, for any processor and any memory module,

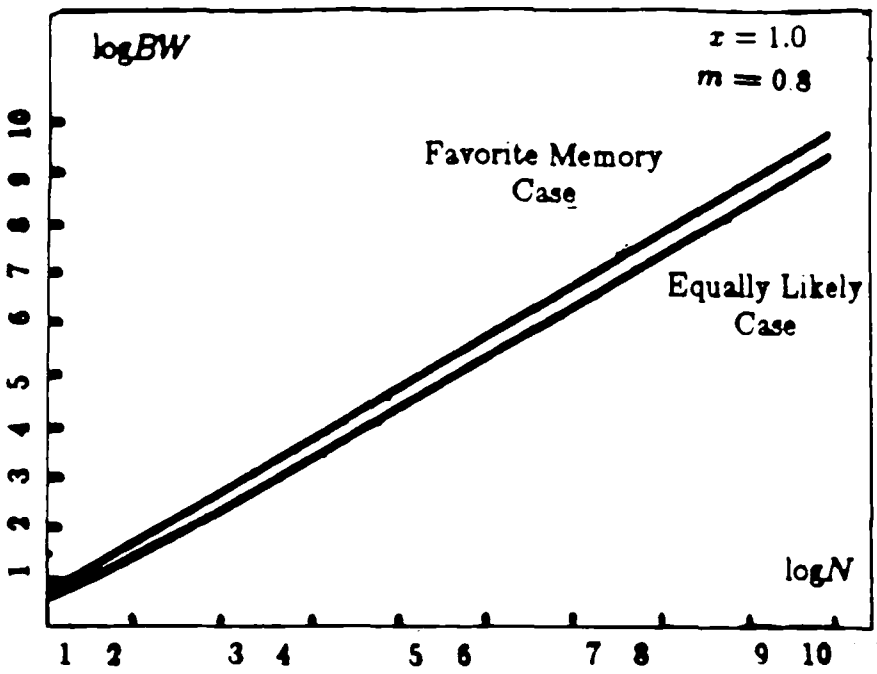

Figure 2.2.2.1: log(BW) for $N \times N$ Crossbar Network.

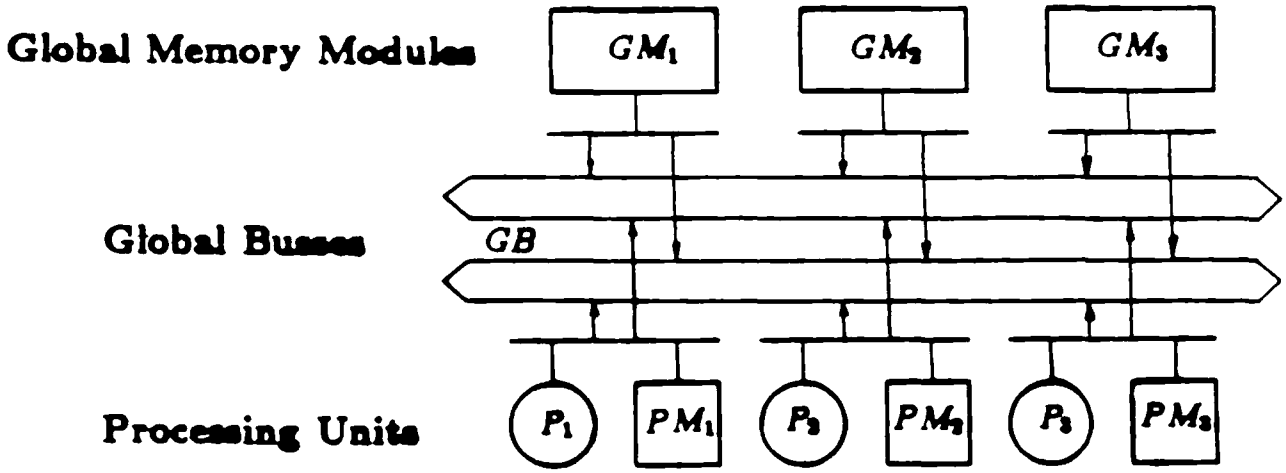

Figure 2.2.3.1: A 3-Processor, 3-Memory, 2-Bus Architecture.

there is a unique path connecting them. In a multi-bus system, a processor can use any global bus to access any memory module. In general, it is expensive to have $b > \min(N, M)$ busses unless one wants some redundancy in the interconnection network for reliability. Moreover, it is not necessary to have that many busses since most memory accesses are made to the private memories without the use of the busses.

In fact, if $b > \min(N, M)$, the model becomes relatively easy to analyze. There is no interference for the use of the busses. Interference arises only when there are two or more requests for the same memory module. Such a model was by analyzed by Ravi ([Ravi72]) to study the bandwidth and interference in interleaved memory systems. In that model, N processors issue a request at random to M memory modules at the beginning of each memory cycle. It is assumed that $N < M$ (The model with $N \geq M$ was considered by Hellerman ([Hell67]) and analyzed in a similar fashion). If the request cannot be satisfied, it is queued at the corresponding memory module. It is assumed that at most N requests can be queued. From these above assumptions, the following expression for the bandwidth is obtained

$$BW = \sum_{k=1}^{N} \frac{k! k S(N,k) \binom{M}{k}}{M^N} \qquad (2.2.3.1)$$

where $S(N, k)$ denotes the Stirling number of the second kind†.

It can be seen from 2.2.3.1 that if one increases M but keeps N fixed, there is less interference (requests are uniformly distributed, thus less chance of a request to the same memory module) and thus a higher memory bandwidth.

The above models completely ignore interference over the use of busses. In real bus interconnection networks interference is typically due to the contention over

† The Stirling number $S(N, k)$ of the second kind is defined as the number of ways of putting N different things into k like cells, with no cells empty ([Rior78]).

the use of the bus rather than over a memory module. Therefore, it is reasonable to assume that whenever a processor issues a memory request, if a bus is available, the requested memory is also available. In other words, one can assume that interference in such systems is mainly due to busses. As noted by Marsan ([Mars82a]), to account for the interference of both busses and memory modules would make the analysis intractable. Simultaneous possession of busses and memory modules for which individual queues exist makes queueing models of multi-bus architectures fall outside the range of applicability of the BCMP theorem. Such queueing models are thus not in a product-form and are computationally intractable.

One can, however, analyze the lower and upper bounds of system performance ([Mars82a]). An upper bound can be obtained by assuming that interference is due only to busses, and thus memory interference may be completely neglected. Processors are assumed to issue requests for busses according to the Poisson distribution with rate λ. The busses are used for an exponentially distributed period with mean μ. If no bus is available, the processor request is queued. If there are b busses and N processors, the model corresponds to an $M/M/b$ queueing system with a finite number of customers ([Klei75]). It is illustrated in Figure 2.2.3.2. The steady-state probability Φ_k that there are k requests in the queue ($N-k$ active processors) is given by

$$\Phi_k = \begin{cases} \frac{1}{G}\binom{N}{k}\left(\frac{\lambda}{\mu}\right)^k & 0 \le k \le b \\ \frac{1}{G}\binom{N}{k}\frac{k!}{b!b^{k-b}}\left(\frac{\lambda}{\mu}\right)^k & b \le k \le N \end{cases} \qquad (2.2.3.2)$$

Here G is the normalization constant of the probabilities Φ_k.

From the above expression, one can easily calculate the upper bound for the bandwidth

$$BW = \sum_{k=0}^{N}(N-k)\Phi_k \qquad (2.2.3.3)$$

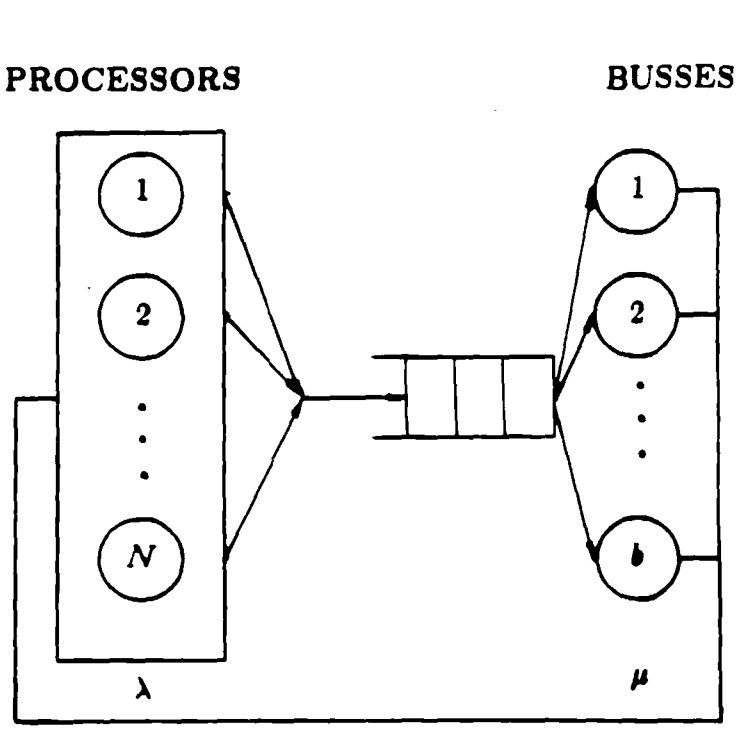

Figure 2.2.3.2: Upper Bound Queueing Model.

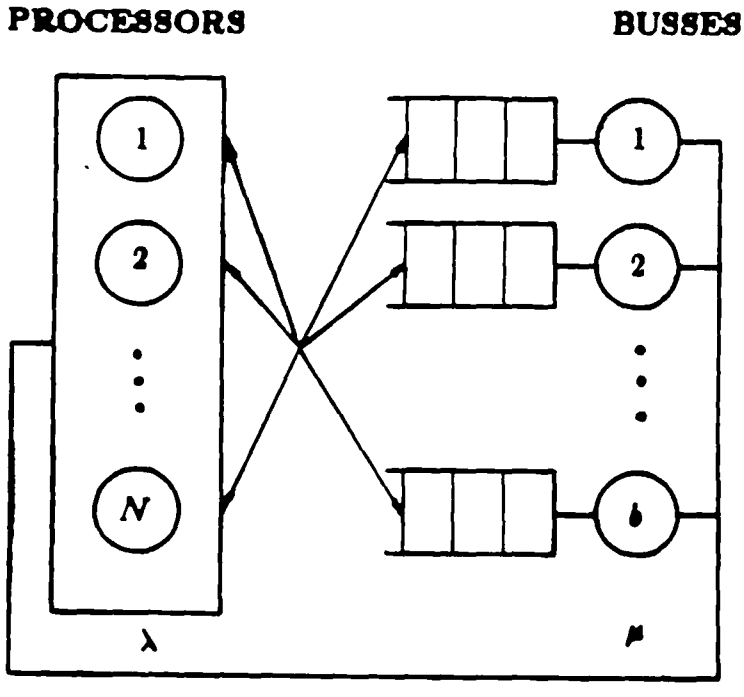

Figure 2.2.3.3: Lower Bound Queueing Model.

To obtain a lower bound, a number of pessimistic assumptions should be introduced. One assumes that the number of shared resources is equal to the number of busses. This hypothesis is pessimistic since the number of resources is reduced and the simplified model will have a higher level of interference than in real systems. Processors issue requests for the shared resources after exponentially distributed active periods. Each resource is selected at random with the probability $1/b$. If the resource is available (the bus is certainly available - there are as many busses as resources), it is accessed for an exponentially distributed time. Then the bus and the resource are released. The approximate queueing network corresponding to this model contains one infinite server station and b single server stations with exponential service times (Figure 2.2.3.3). The steady-state probability distribution Π_k that there are k queued processor requests ($N-k$ processors are active) is given by

$$\Pi_k = \frac{1}{G}\left(\frac{\lambda}{b\mu}\right)^k \binom{N}{k} \frac{(b+k-1)!}{(b-1)!} \qquad (2.2.3.4)$$

where G is the normalization constant for the probabilities Π_k.

The resulting expression of the lower bound for the bandwidth can be found as follows:

$$BW = \sum_{k=0}^{N} (N-k)\Pi_k \qquad (2.2.3.5)$$

Although the lower and upper bounds are available (equations 2.2.3.2 - 2.2.3.3), more precise analyses for the general multi-bus architectures are not yet available.

Some single-bus architectures have been analyzed by Marsan, Balbo, Conte and Grigoretti ([Mars82b, Mars82d]). In the first model (Figure 2.2.3.4), processor P_i can access its private memory PM_i by the local bus LB_i. There is a common memory (CM) external to all processors and available only through the global bus

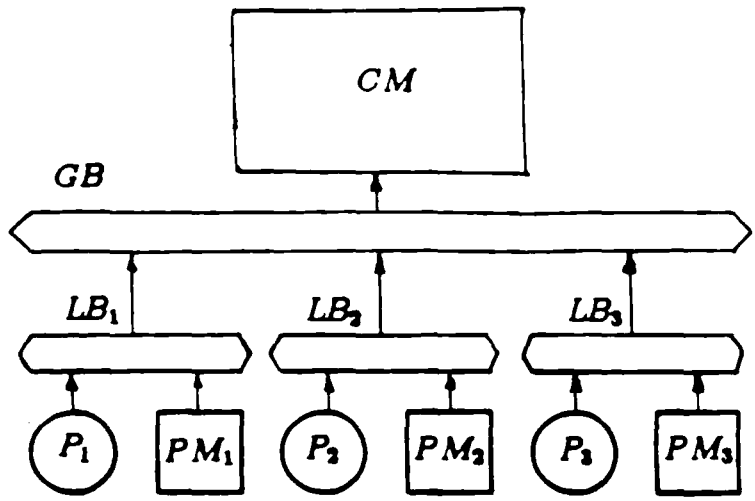

Figure 2.2.3.4: First Single-Bus Model.

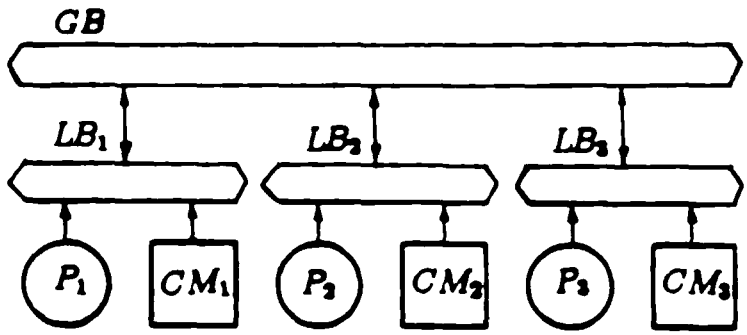

Figure 2.2.3.5: Second Single-Bus Model.

(GB). Interference arises each time a message is written in (read from) common memory. Only one processor P_i can access common memory at a time. If the bus is busy, it has to wait. Assuming that both the time between requests and the memory access time are exponentially distributed, the model corresponds to a "machine repairman" model ([Alle78]). In this model, the traffic of messages flowing out of a processor balances that flowing into the same processor. Thus, the processor activity is interrupted with a rate which is twice the rate of generation of messages. In the second model, the common memory CM is divided into sub-memories CM_i. Processor P_i can connect to CM_i by the local bus LB_i. A processor P_i accesses a "non-local" memory CM_j $(i \neq j)$ by using its own local bus LB_i, the global bus GB and the local bus LB_j connected CM_j (Figure 2.2.3.5.). Because of the blocking phenomenon due to one processor accessing the local memory of another one, the second model cannot be modeled as a simple queueing system (even with the assumptions of Poisson generation of requests and exponentially distributed memory access times). The problem is solved by explicitly constructing a corresponding Markov chain.

These architectures are compared for the same processor communication load. For light loads, the first model outperforms the second, despite the fact that it generates twice as many communications with the global bus than for the second model. This can be explained by the fact that with light loads, the average queueing delay is quite low, making the additional interference introduced by the first model negligible. With the second model, on the other hand, every access to an external common memory preempts a processor, whose probability of being active on its local memory is very high under light load conditions.

The comparison of performance under low interference (light load) is important. Well-designed multiprocessor systems should operate in this region if the problem decomposition into tasks and the task allocation to resources is aimed at reducing communication overhead. From this perspective, the first model is a good choice, considering the simplicity of organization. For heavily loaded systems, it is

better to use local common memories.

Goyal and Robinson ([Goya84]) present the analysis of two generic classes of multi-bus computer systems. The first system, called BIP (billion instructions per second), consists of a very few high performance processors. The primary motivations are improved throughput, reliability, and availability. The second system, called KMIP (K million instructions per second), consists of hundreds of low speed processors connected to a large central memory. The examples of such systems are airline reservation systems.

The BIP system is given in Figure 2.2.3.6. The model and analysis of KMIP system follows the same lines. There are N processors connected to M shared memory modules by B busses. A processor serves a job for an average of $1/m$ time units before a miss in local memory occurs. On such a miss, a processor references a shared memory module with an equal probability $1/M$. The processor waits till the requested page is transferred to its local memory. It is assumed that a page request takes an average of w units of time to obtain the required shared memory modules and an arbitrary bus. The page request waits in a queue till all the previous page requests to those shared memory modules are satisfied. It takes a constant t units of time to access and transfer a page from the shared memory to the local memory. Finally, every page request causes an additional (in parallel) write request with probability m_w.

It can be shown that the queueing system described above is closed queueing network, which is always stable and has an equilibrium state. The queueing model for the BIP system is presented in Figure 2.2.3.7. If U_p is the processor utilization, and M' is the average number of busy memory modules, it can be shown that

$$\begin{cases} M' = [NU_p m(1+m_w)][t] \\ U_p = \frac{1}{1+m(1+m_w)(w+t)} \\ m' = [U_p m(1+m_w)][(w+t)] \\ M' = M[1-(1-\frac{m'}{M})^N] \end{cases} \tag{2.2.3.6}$$

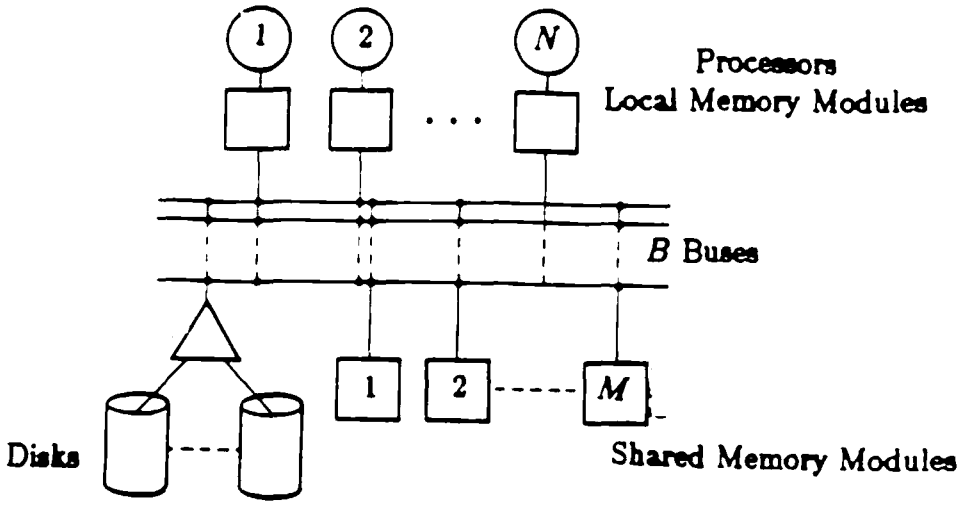

Figure 2.2.3.6: The BIP System.

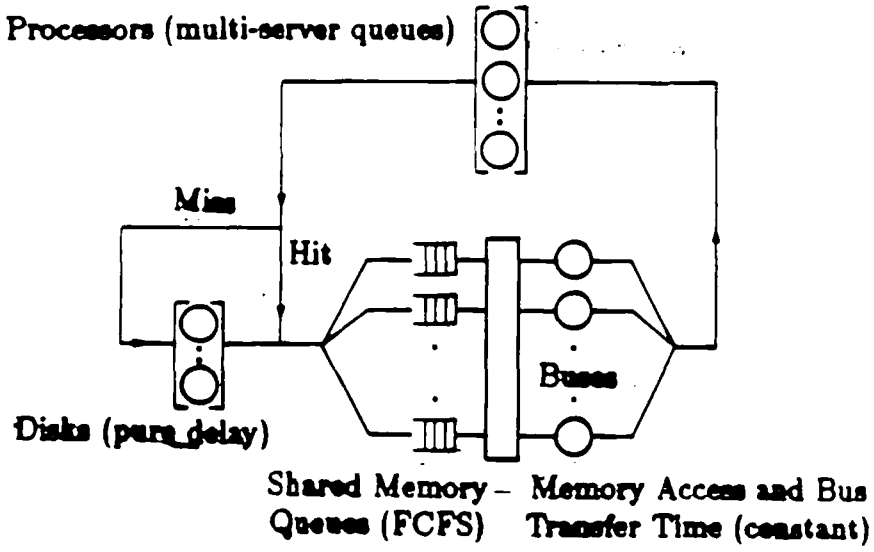

Figure 2.2.3.7: Queueing Model for the BIP System.

This gives 4 equations in 4 unknowns (M', U_p, w, m'), which can be reduced to a single nonlinear equation in U_p. Therefore, the performance of the system can be evaluated by solving this nonlinear equation. The following approximation is suggested: Since m'/M is the probability that a shared memory module is addressed from the local memory, the probability that i shared memory modules are requested is

$$P(i) = \binom{M}{i} \left[1 - \left(1 - \frac{m'}{M}\right)^N\right]^i \left[\left(1 - \frac{m'}{M}\right)^N\right]^{M-i} \qquad (2.2.3.7)$$

If $B \geq M$ then the equation 2.2.3.7. also gives the probability that i shared memory modules are busy. If $B < M$, the maximal possible number of busy modules is B. Therefore, the average number of busy memory modules for an arbitrary number of busses can be approximated as

$$M' \approx \sum_{i=0}^{M} \binom{M}{i} \left[1 - \left(1 - \frac{m'}{M}\right)^N\right]^i \left[\left(1 - \frac{m'}{M}\right)^N\right]^{M-i} \min(i, B) \qquad (2.2.3.8)$$

The utilization of an individual bus is given by $U_B \approx M'/B$. The authors call this the *truncation* method.

The analytical results are within 5% of the simulation results. The most critical design parameter is the bandwidth of a single bus. If the required bandwidth cannot be made available on a single bus, it may be traded to some extent for an increased number of buses and shared memory modules. The proposed model allows us to study the effect of bus failures on system performance by varying the number of busses B. This is unlike some of the previous work, which assumed a fully functional interconnection network for the analysis.

Many of the assumptions are not met by real life bus-shared systems. Access times may not be of exponential duration, processors may not evenly share the

load, and there may be more interference in some resources than in others. The uniform distribution of requests (the uniform distribution of interference) is definitely an optimistic assumption: if there is more interference at a particular bus, this resource becomes a system bottleneck and thus the processing power decreases. If the above queueing models are extended to include non-uniform sharing of total load among processors, the solution becomes complex since one must separate customers into classes. Despite the growing number of multi-bus architectures, very few analytic studies have been done on this.

2.2.4. MULTI-STAGE INTERCONNECTION NETWORKS

A multi-stage interconnection network is a network in which the nodes can be arranged in stages, with all the source nodes at stage 0, and all the outputs at stage i connected to inputs at stage $i+1$. These networks can provide an effective interconnection scheme for parallel computations of such problems as the Batcher sorting method ([Ston71]), polynomial evaluation, and fast Fourier transforms ([Peas77]).

An example of a multi-stage network is the shuffle-exchange (or Omega) network ([Lawr75]). The standard $N \times N$ shuffle-exchange network ([Lawr75]) consists of $k = \log_2 N$ columns of routing switches connecting N processors to N memories. An example of a 3-column 8×8 shuffle-exchange network is given in Figure 2.2.4.1.

Each column consists of $N/2$ two-input, 2-output routing switches. A perfect shuffle connection is used between each pair of adjacent columns. The switches forming the path from the source processor S to the destination memory D are set according to the binary value of the address of D. A switch in column j will interpret the j-th bit of D to set its internal path. If the j-th bit is 0, the upper output of request will be used, if it is 1 the lower output is used.

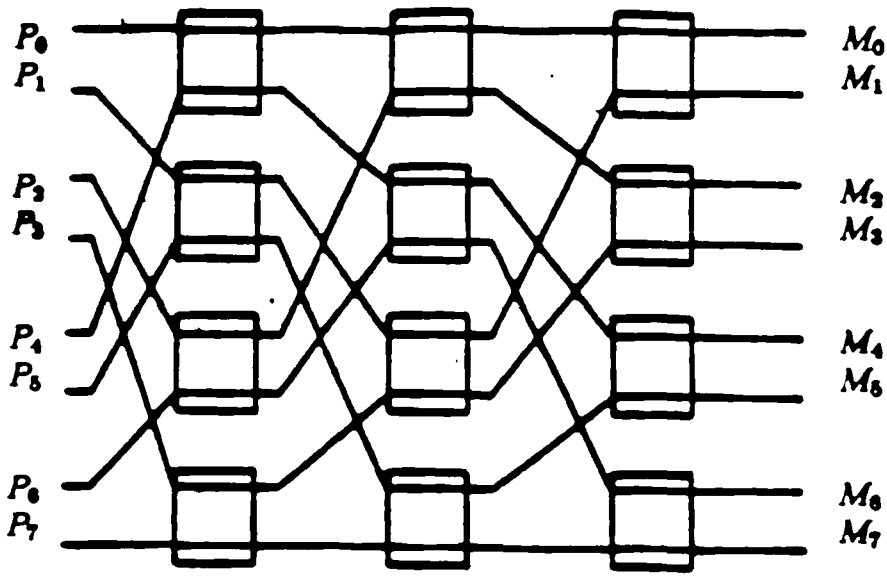

Figure 2.2.4.1: Standard 3-Column 8 × 8 Shuffle-Exchange Network

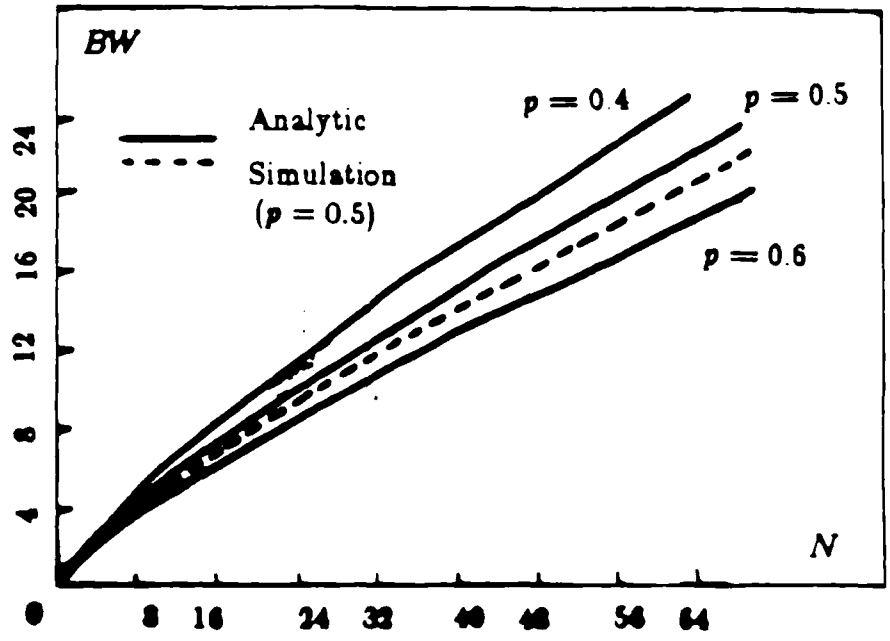

Figure 2.2.4.2: Memory Bandwidth for Shuffle-Exchange Network

Shuffle-exchange networks have many topologically equivalent forms, depending on the interconnection pattern. These include delta networks ([Pate79]), indirect binary cube ([Peas77]) and others. Most of the analytic results for multistage interconnection networks are in fact concerned with the shuffle-exchange class of networks.

In a shuffle-exchange network, if each input of a switch at column j has an active request, then whether or not there is interference at that switch will depend on the i bits of two destination tags. If they are the same, interference exists and only one request can be established. Because the j-th column examines only the j-th bit of a destination tag, it is reasonable to assume that if requests generated by processors are random, the binary distribution of destination bits is random as well. This assumption of "interference independence" between stages is the central idea of the analysis of all of these networks.

The first analytic results for shuffle-exchange networks were reported by Nelson and Thanawastien ([Than81]). They have analyzed these networks in a circuit switched mode using the discrete Markov chain approach. To calculate the bandwidth, one determines the probability of having n_i active outputs in column i, given that n_0 requests are originated by N processors. This probability is calculated by considering a 1-column $2^N \times 2^N$ network, n_0 active inputs, and calculating n_1. These n_1 active outputs become inputs to the second column of the network. The process is repeated k times resulting in n_k requests at the output of the k-th stage. The analysis proceeds under the assumption of equally likely access patterns and synchronous operation. The memory bandwidth is a polynomial in p, which is the probability of having a conflict at a switch given two active inputs. From the expressions of memory bandwidth, several other important parameters are derived, including the non-blocking probability $P = BW/N$ and the average delay $D = t_N/P$. (Here t_N denotes the time to complete a basic network cycle, that is, the time for a memory request to travel to and from shared memory) The figure 2.2.4.2. shows the memory bandwidth BW for some p. It is interesting to note that for a

fixed p, the bandwidth increases almost linearly with N.

Even though one can argue that unbiased resolution of the conflict should imply that p=1/2, the possibility of favorite memory requests ("locality of interference") implies that the value of $p < 1/2$ or $p > 1/2$ is quite reasonable.

Since no restriction is made on the connection pattern between columns of 2-input, 2-output switches, the above results are applicable to all of the shuffle-exchange networks (indirect binary n-cube, Delta, Baseline, and Reverse-Exchange).

In the above analysis it was assumed that the processor always generates the request at the beginning of a memory cycle. In real systems, however, this may not be the case. It is more realistic to assume that at the beginning of a cycle, a processor generates a request with some probability. Patel ([Pate81]) has analyzed such a model of $2^N \times 2^N$ shuffle-exchange networks, assuming synchronous operation, uniform distribution of requests, and a fixed probability m_0 that a processor generates a request during a cycle. Under these assumptions, one can derive a relation between the request rate m_i on an output line of stage i and the initial request rate m_0. Using this, recursive formulas for the bandwidth BW and the non-blocking probability P are derived:

$$m_i = 1 - \left(1 - \frac{m_{i-1}}{2}\right)^2 \qquad BW = 2^N m_N \qquad P = \frac{m_N}{m_0} \qquad (2.2.4.1)$$

The above equations can be solved only numerically. The comparison of P_A and BW with those of crossbar networks is illustrated in Figures 2.2.4.3. and 2.2.4.4.

The non-blocking probability for the crossbar approaches a constant value, whereas it continues to fall for shuffle-exchange networks as N grows. The bandwidth provided by the crossbar is higher than that of delta networks but at an enormous increase in the number of switching elements (N^2 vs. $N \times \log_2 N$). Note that the bandwidth is linear for larger N. This is consistent with the previous analysis of Thanawastien and Nelson ([Than81]) described above.

Bhuyan and Lee ([Bhuy83]) have extended the above approach to show the

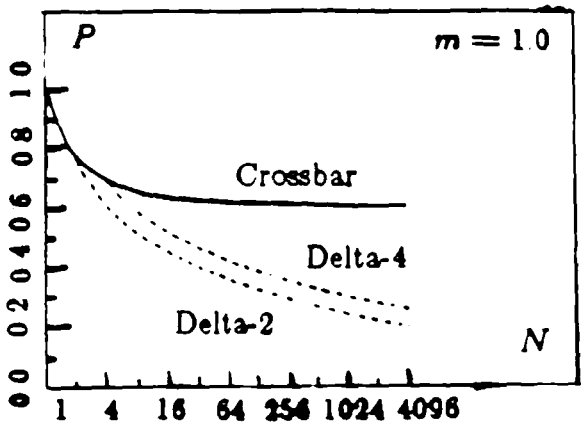

Figure 2.2.4.3: Non-Blocking Probability

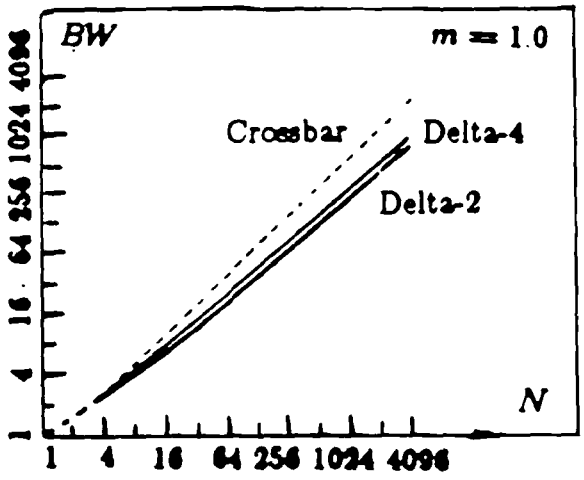

Figure 2.2.4.4: Bandwidth for Crossbar and Delta Networks.

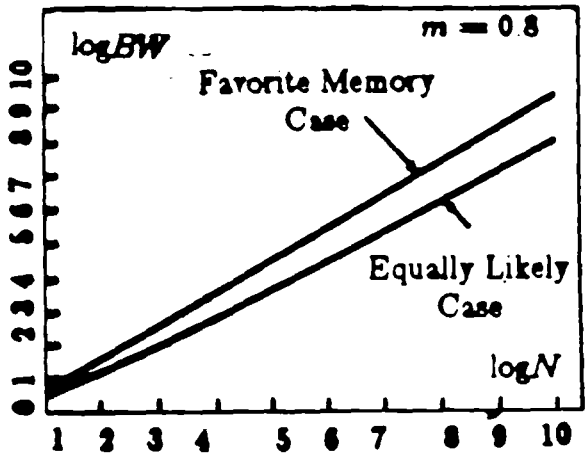

Figure 2.2.4.5: Comparison of Favorite and Equally Likely Access.

effect of "favorite memory case": a processor P_i requests a particular memory module M_i most of the time. The favorite memory connection for standard 3-column 8×8 shuffle-exchange network is shown in Figure 2.2.4.1. From the probabilities of favorite memory request m_0, a recurrence relation for the bandwidth (BW) is derived by quite complicated arguments. An example of comparison of favorite and equally likely access is given in Figure 2.2.4.5. For favorite memory case ("interference locality") the bandwidth is higher than for uniform access. This can be easily explained by the fact that processors try to access their favorite memory modules most of the time. This leads to a decrease in interference and thus to a higher bandwidth.

The above systems were analyzed in circuit-switching mode. Diaz ([Diaz81]) analyzed the delta networks in packet-switching mode. Under the packet-switching mode, the processor does not need to establish a continuous connection, but sends data in packets from one stage of the network to the next. Diaz assumed that the networks operate asynchronously with packet lengths distributed independently and equiprobably among all possible destinations. In addition, the analysis is done under the assumption of "maximal loading" - a buffer at the input link always has a packet to send. If interference occurs when two packets try to use the same link, one of the packets is equiprobably selected while the other is rejected. No "interference locality" is assumed.

Under these assumptions, the dependence of the rate of increase in throughput (average number of packets processed by the network per unit of time) on the number of buffers is largest for a change from one to two buffers. This rate increase falls off sharply as the buffer size increases. It seems, therefore, that for most applications, the number of buffers between stages should be limited to one or two. The approximate analysis and simulation results suggest no significant difference when the rejected packets are assumed to be lost. Thus, the simple model which assumes that interfering packets are lost can be used to obtain a fairly good estimate of interference for more realistic models and environments. Again, since no assumption

was made on a particular interconnection pattern, the above results are applicable to all multi-stage permutation networks.

The multi-stage interconnection networks considered above have a unique path from each processor to each destination (memory module or another processor). Any interconnection network with a unique path from "source" to "destination" is called a *banyan* network ([Goke73]). The term "banyan network" therefore encompasses a fairly large variety of switching networks whose interconnection geometries may be either highly regular or almost random. A multi-stage banyan network is therefore any multistage network with a unique path between any processor-processor or processor-memory pair. For example, the 8 × 8 shuffle exchange network given in Figure 2.2.4.1. is an example of a 3-stage banyan network built of 2 × 2 switches.

Unbuffered multi-stage banyan networks built of $k \times k$ switches were analyzed by Kruskal and Snir ([Krus83]). The unbuffered network is assumed to operate in synchronous packet switching mode, where packets are generated by independent random processes (no "interference locality") and there is a fixed probability for each processor to generate a packet. The relevant performance measure is the probability m_i that there is a packet on any particular input at the i-th stage of the network. This probability is shown to be

$$m_i = \frac{2k}{(k-1)i}\left[1 - \frac{(k+1)}{3(k-1)}\frac{\log_e i}{i} + O\left(\frac{1}{i}\right)\right] \qquad (2.2.4.2)$$

Therefore, the non-blocking probability is inversely proportional to the number of stages in the network. In particular, for the network built of 2 × 2 switches, the bandwidth

$$BW = N m_{\log N} = \frac{4N}{\log_2 N}\left[1 - \frac{\log_e \log_2 N}{\log_2 N} + O\left(\frac{1}{\log_2 N}\right)\right] \qquad (2.2.4.3)$$

Thus, interference grows geometrically with the number of stages, resulting in a poor bandwidth if many stages are used. The bandwidth of packet-switching networks

can be improved by using buffers to queue interfering packets. An accurate analysis of interference of buffered square banyan networks does not seem tractable.

One of the assumptions in the above analysis of multi-stage interconnection networks is the assumption of uniform distribution of traffic. This may not always be true in practice. It often happens that there are one or more *hot spots* - locations to which a specified fraction of the total references is directed. The quantitative investigation of the performance impact of such contention was considered by G.P. Pfister et al. ([Pfis85]). They analyzed an omega network ($4 \leq N \leq 64$), where the fraction of references directed to a "hot spot" was varied from 0.5% to 32%. If N is the number of processors, r ($0 \leq r \leq 1$) is the number of packets emitted per processor per unit time and h is the fraction of memory references directed at the hot spot, then there are $r(1-h) + rhN$ packets to the "hot" memory during one cycle. The hot spot occurs when this value equals to the capacity of the weakest link in the path between a processor and a "hot" memory. Saturating the capacity of the link, which is connected to hot memory, causes queues in the switch closest to that memory to fill. The same happens to the switches in the previous stage and so on.

Therefore, hot spot non-uniformity in the traffic can produce global degradation. The limitation from this effect can be significant. As shown in [Pfis85], for large systems with 1000 processors, the hot spot traffic of 0.125% limits the potential bandwidth of the system to 500, that is 50% efficiency. This effect of "hot spot" degradation is independent of network topology or switching mode (packet or circuit). It occurs in any multi-stage network with distributed routing. One of the possible solutions, suggested and implemented as part of the RP3 project ([Pfis85]), is to detect the occurence of memory requests directed at identical memory locations as they pass through each switch node. Such messages are combined into a single message. The preliminary results indicate ([Pfis85]) that this provides an adequate solution. The expense is, of course, the additional cost of a combining network. The development and analysis of strategies to solve the "hot-spot" degradation is

an important area of research.

A general queueing model for multi-stage networks was proposed recently by Mudge and Makruchi ([Mudg82]). Under the assumption of Poisson arrivals, uniform distribution of requests, and exponential service times, the first stage can be modeled as an $M/M/1/L$ queue. However, the output of that stage to the second stage fails to be Poisson. For large switches the distribution is assumed to be approximately Poisson. With such an approximation the network can be represented by a sequence of $M/M/1/L$ queues. The obtained results are within 20% of the simulation results. Deriving an exact tractable queueing model for multi-stage banyan networks appears to be impossible.

2.2.5. CONCLUDING REMARKS ON INTERCONNECTION NETWORKS

Exact models of "realistic" interconnection systems typically do not satisfy the BMCP theorem and are thus computationally intractable. For example, consider a shuffle-exchange system operating in the asynchronous mode and assume that memory requests are queued at inputs to stage 0 of the network. Some of the requests queued at different inputs can be active, and some cannot be active, depending on the address of the destination nodes. To describe a queueing model for such a system, one would need to take into account not only the size of the queue at each input, but also the contents of each queue. Such a queueing model is not in a product form.

Despite the growing number of multiprocessor systems, relatively few analytic studies of performance of interconnection networks have been done.

2.3. MULTIPLE ACCESS PROTOCOLS

2.3.1. GENERAL OVERVIEW AND PERFORMANCE ISSUES

The next class of models considered is packet broadcast networks. Packet radio networks consist of geographically distributed computers broadcasting data packets over a shared communication medium. It is often the case that two units cannot have a direct connection. Thus intermediate users must act as relays, creating a multihop communication network. Such a network can be represented by a graph where nodes are transmission sources and edges connect nodes that can directly communicate.

Since one is dealing with a network operating in a packet-switching mode, conflicts necessarily arise whenever transmissions attempt to acquire the same channel simultaneously. The problem is to minimize this conflict by providing an adequate access control. This "interference resolution" problem is non-trivial since the users are usually geographically distributed. To solve this problem, a set of rules which define under what conditions a node is allowed to transmit must be defined. Such a set of rules is called a multiple access protocol.

What are the key performance measures in analyzing the impact of interference in packet radio networks ?

- The most important performance measure is the throughput S - the average number of successful concurrent transmissions processed per unit of time and the dependence of the throughput on the system parameters (number of nodes N, offered traffic G, etc.).

- Capacity - the maximum throughput that a system can achieve.

- The relationship between the throughput S and the delay D (the average time from the point when the packet is generated until it is successfully received).

- Channel utilization - the percentage of channel capacity used by successful transmissions.

- Optimal degree - for a given protocol and the network size N, the average degree d (how many other nodes can a node "hear" on the average) that maximizes throughput. The larger the degree d (the more powerful the transmitter) the fewer "hops" will be required for a packet to reach the destination and thus increase the throughput. On the other hand, increasing d will introduce more interference and thus tend to degrade the throughput.

It is recognized that certain random access channels are unstable in the absence of certain channel control procedures ([Fayo77, Carl75, Lam75, Medi83, Toba77]). In the case of the so-called absolute instability, the channel throughput becomes zero even under very light traffic conditions. In the case of the so-called oscillatory instability, the channel throughput jumps at random times from one locally stable point to another. The detailed discussion of the question of stability is, however, beyond the scope of this paper.

Many multiple access protocols for interference resolution have been proposed and studied by numerous researchers ([Abra70, Boor80, Braz85, Klei80, Lam75, Rubi83, Silv83, Taka83, Toba75] to name just a few). An excellent survey is the paper by Kurose, Yemini and Schwartz ([Kuro84]). For the purpose of the discussion of interference, these protocols can be classified into two different classes: controlled-access and contention protocols.

Under controlled-access protocols, the distributed stations are coordinated in such a way that there is no interference - two or more stations never try to transmit at the same time. Such a coordination is achieved by some ordering of access rights. Under contention-based protocols, each user monitors the broadcast channel and tries to transmit his data the best he can without incurring interference. Interfering packets are retransmitted by the users according to control algorithms. The reader is again referred to an excellent comparative survey of these protocols

in [Kuro84].

In this paper only contention-based protocols are considered. We will not discuss the tree access protocols † ([Cape79, Kapl85]). The contention-based protocols discussed in this paper can be divided into two main groups: ALOHA-type and Carrier-Sense type schemes.

2.3.2. ALOHA-TYPE SCHEMES: SINGLE-HOP CASE

The ALOHA scheme is perhaps the first example of a multiple access protocol. Under the Aloha-type schemes ([Abra70]) any node wishing to transmit over the shared broadcast channel, does so independently of the others. Under light traffic, because of low interference, the transmission will succeed with a high probability, and thus with a low delay. As the load increases, so does the probability that a node that wants to use the channel will encounter interference when another node also wants to transmit. As a result of interference a "collision" occurs, destroying the packets of both nodes. These nodes retransmit the messages after some random time to avoid future interference. In the pure-ALOHA, each node transmits over a shared channel in completely unsynchronized manner. If within some appropriate time-out period they receive the acknowledgment from the destination, then they know that no interference was present. Otherwise, they assume that interference has occurred. Clearly, a given packet of time duration T interferes with any other packet broadcasted within T seconds before or after the start of the given packet. Thus, the interference can occur over a "vulnerable" period of length $2T$. To avoid continuously repeated conflicts, a random retransmission delay must be introduced, spreading the interfering packets over time. If one assumes that the generation of traffic is Poisson, it can be shown ([Abra70]) that for the offered channel traffic load

† In a tree access protocol, there is a sequential decision process to isolate a single busy node which is then allowed to transmit its data packet.

G, the throughput S is given by

$$S = Ge^{-2G} \tag{2.3.2.1}$$

This is shown in Figure 2.3.2.1. The maximum throughput is $S_{max} = \frac{1}{2e} = 0.184$ at $G = \frac{1}{2}$.

The effect on performance when packet lengths are selected from the random distribution was analyzed by Ferguson ([Ferg77]). The main result of his study was that if the mean of the packet length is the same as the constant packet length considered above, the optimum performance is achieved for constant packet lengths.

If the packets are of the same length, then the obvious way to minimize interference would be to minimize the "vulnerable" length. This is the idea of the slotted-ALOHA ([Robe74]). The completely unsynchronized channel is modified by "slotting" time into segments corresponding to a time to transmit a packet. Nodes can transmit only at the beginning of a time slot. The "vulnerable" period is reduced twofold to T. The throughput equation becomes

$$S = Ge^{-G} \tag{2.3.2.2}$$

The maximum throughput is $S_{max} = \frac{1}{e} = 0.368$ at $G = 1$. As one would expect, the maximal throughput is twice that of the pure-ALOHA. The corresponding throughput curves of slotted and unslotted ALOHA are given in Figure 2.3.2.1.

The above systems are analyzed under the pessimistic assumptions that if two packets interfere, both packets are lost. In practice, such an assumption provides a lower bound on system performance, since the stronger of two interfering packets may capture the receiver and be received without error. This fact can be used to show that by dividing users into two groups - one transmitting at higher power and one transmitting at lower power, the maximum throughput can be increased by about 50% ([Metz76]). The result may be of importance to packet radio networks with a mixture of data and packetized speech traffic.

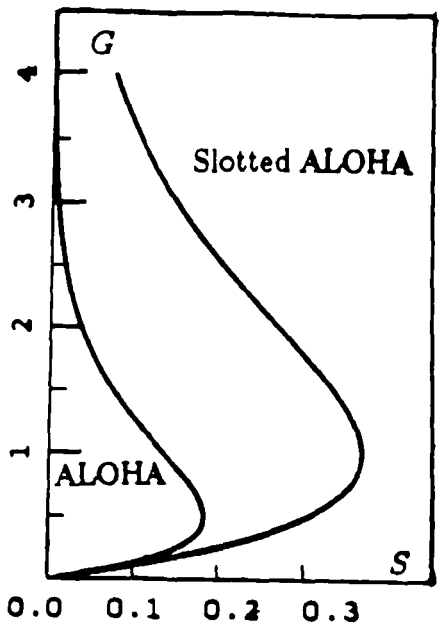

Figure 2.3.2.1: Throughput in ALOHA Networks

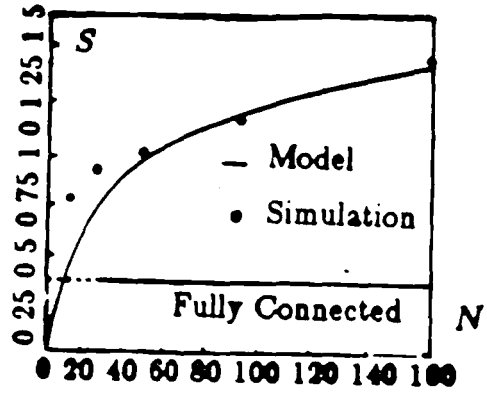

Figure 2.3.2.2: Throughput for One-Dimensional Random Networks

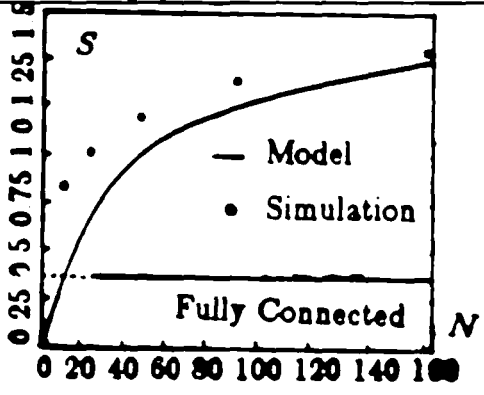

Figure 2.3.2.3: Throughput for Two-Dimensional Random Networks

The single-hop slotted ALOHA networks have been analyzed for various traffic matrices and transmission strategies by Silvester and Kleinrock ([Silv83a]). The network environment is a set of randomly located nodes that are able to communicate in one hop, described by the traffic matrix. In this model, if a node i has a packet to transmit, it does so in any time slot with probability p_i. This corresponds to the offered traffic randomized so that slotted ALOHA will operate correctly and resolve previous conflicts due to interference of simultaneous transmissions. If each node can reach any other node (completely connected topology) and carries identical traffic, then $p_i = p$. In such a case the throughput S is given by

$$S = Np(1-p)^{N-1} \mapsto N\frac{1}{Ne} = \frac{1}{e} \quad \text{as} \quad N \mapsto \infty \tag{2.3.2.3}$$

which is of course the familiar result for the capacity of the slotted ALOHA.

We can consider restricting the transmission range for a uniform traffic matrix. The transmission probability p_k that a node "hits" k nodes (including itself) is assumed to be $p_k = 1/k$. The throughput S_k for a node that hits k nodes is assumed to be the same for all of the N nodes and is $S_k = 1/N - 1$. The throughput of the network is then shown to be

$$S = \frac{N}{N-1}\left[\prod_{k=3}^{N}\left(1 - \frac{k-2}{N-2}\frac{1}{k}\frac{1}{N-1}\right)^{N-2}\right]\left[\sum_{k=2}^{N}\frac{1}{k} - \frac{1}{k^2}\right] \approx \frac{\log N}{e} \tag{2.3.2.4}$$

In other words, the throughput can be made proportional to the logarithm of the number of nodes (equation 2.3.2.4). This is verified by the simulations: random networks are generated with points uniformly distributed within a unit circle. Pairs are randomly assigned and transmission radii determined. The transmission probability for a node is taken to be the reciprocal of the number of nodes within the range of that node. From this, one can compute the success probabilities for each node and hence the throughput. The results of the simulation are given for

1-dimensional networks (Figure 2.3.2.2) and for 2-dimensional networks (Figure 2.3.2.3).

An excellent agreement is seen between the model and the simulation. For small networks, a large proportion of the nodes are close to the edge of the networks, and thus one would expect that the agreement is not as good for small networks. Nodes near the edge suffer less interference. As the number of nodes increases, these edge effects become proportionately less important. In two dimensions, there is a higher proportion of nodes on the edge of the network and thus the model requires larger networks in order for the agreement to be good.

The next problem is finding the best traffic matrix to maximize throughput: what traffic matrix allows the highest traffic levels to be supported for a random network ? In theory, it is possible to find a traffic matrix such that the throughput S is proportional to the number of nodes. Because of the "interference locality" of traffic requirements, it is reasonable to expect that the performance of single-hop slotted ALOHA is between the logarithmic and the linear performance.

2.3.3. ALOHA-TYPE SCHEMES: MULTIPLE-HOP CASE

In the previous section, it was assumed that the packet can reach its destination in one hop. It is more realistic to consider the multiple-hop case: the packets have to travel through the intermediate node(s) to reach the destination.

The simplest case is, of course, the two-hop ALOHA network. The two-hop slotted ALOHA has been analyzed by Tobagi ([Toba80]). The studied configurations are the star configuration and the fully connected configuration shown in Figure 2.3.3.1. In these systems, the traffic originates at terminals and is destined to a central station. This traffic may require that packets are relayed by store-and-forward receivers.

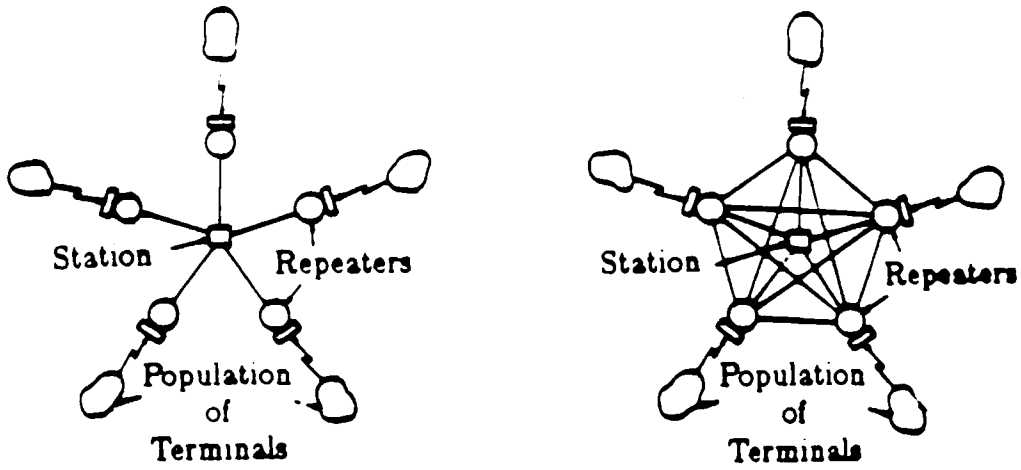

Figure 2.3.3.1: Star (left) and Fully Connected (right) Configuration

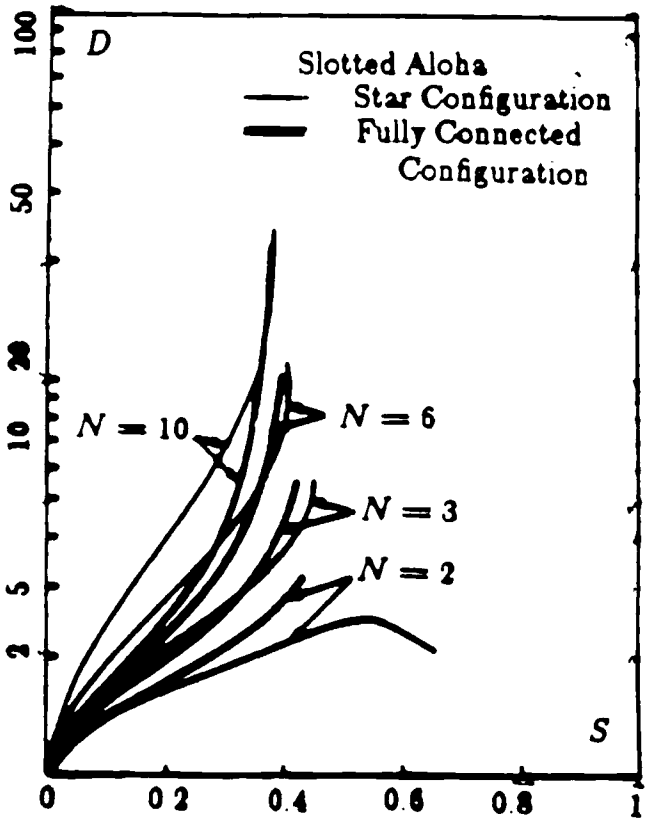

Figure 2.3.3.2: Delay vs Throughput

For small number of users N, the star configuration offers a higher system capacity. As N increases there is more interference at the inner hop, which becomes the critical hop, and both configurations are equivalent in capacity. The fully connected configuration provides smaller packet delays for low and moderate values of throughput. This becomes more noticeable for the larger values of N, where network delay becomes the important component of the total packet delay. This is shown in Figure 2.3.3.2.

Under the assumption of the negligible processing time at the devices, packet radio systems are channel bound: a slight improvement is gained by increasing the buffer size at the nodes from 1 to 2, but no significant improvement is obtained beyond that.

The above studies of the ALOHA were concerned with centralized networks - all the traffic was destined to the central node. For multihop packet radio networks one no longer has a centralized node. This greatly complicates the interference analysis and, in fact, relatively little work has been done in this area.

Silvester and Kleinrock ([Silv83b]) have analyzed multihop slotted ALOHA networks with regular structure - the nodes are placed on a regular graph such as a square grid or loop (Figure 2.3.3.4). It is assumed that the traffic matrix is uniform, that is, each node splits its traffic equally between all possible destinations. With such a traffic matrix and a regular topology, it is assumed that the traffic load on all the links is homogeneous. This seems to be a valid assumption, since either there are no edge effects to consider (e.g. loop) or they can be ignored since they are of minor importance to the rest of the network. Under the above assumptions, we find that the throughput is maximized if the probability of transmission during a slot $p = \frac{1}{d}$, where d is the average degree of a node plus 1 (for the node itself).

To calculate the throughput, one proceeds as follows. First, if $p = 1/d$ one can easily calculate the expected number of successful transmissions per slot for the whole network to be $S_{net} = N/d(1 - 1/d)^{d-1}$. The expected path length in hops is

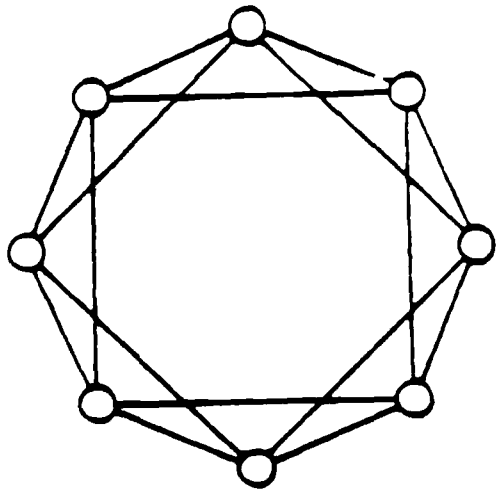

Figure 2.3.3.3: Regular Loop Network

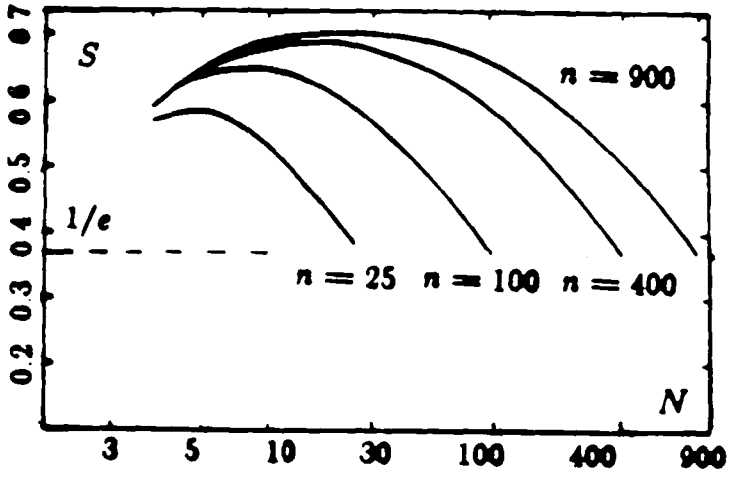

Figure 2.3.3.4: Throughput vs. Average Degree

shown to be $L = (N + d - 2)/(2d - 2)$. The throughput is then

$$S = \frac{S_{net}}{L} = \frac{2N}{N + d - 2}\left(1 - \frac{1}{d}\right)^d \tag{2.3.3.1}$$

Let us first examine the question of the optimal degree d for the loop networks. One can consider two extreme cases. If the network is fully connected ($d = N$), then from (2.3.3.1) we obtain

$$S = \left(1 - \frac{1}{d}\right)^{d-1} \mapsto \frac{1}{e} \tag{2.3.3.2}$$

for large d. This is what one would expect - the throughput corresponds to the usual infinite ALOHA population.

The other case is when each node is only connected to its two neighbors ($d = 3$). For such a case the throughput is

$$S = \frac{2N}{N + 1}\left(\frac{2}{3}\right)^3 \mapsto \frac{16}{27} \tag{2.3.3.3}$$

Since $16/27 > 1/e$, the maximum throughput is achieved for some intermediate value of d. From (2.3.3.1) we can find that the optimal degree turns out to be

$$d_{opt} \approx \sqrt{\frac{N}{2}} \tag{2.3.3.4}$$

The optimal throughput is $S_{max} = 2/e$. In fact, for large N the exact value of d is not very critical as long as it is greater than 3 and grows more slowly than N. Thus, the optimal throughput of $2/e$ will be achieved for any moderate value of d. This is illustrated below in Figure 2.3.3.5.

For the line networks ([Silv83b]), the throughput is

$$S = c\left(1 - \frac{1}{d}\right)^{d-1} \mapsto \frac{c}{e} \tag{2.3.3.5}$$

(The constant c depends on the average path length) For large d the throughput is almost independent of the degree and a capacity of c/e can be achieved.

For two-dimensional networks ([Silv83b]), the average path L for a uniform traffic matrix is found to be proportional to $\sqrt{N}/\sqrt{d}$. For small d the throughput is

$$S = c\sqrt{\frac{N}{d}}\left(1 - \frac{1}{d}\right)^{d-1} \qquad (2.3.3.6)$$

Here c is some constant, depending upon the topology. The above expression (equation 2.3.3.6) means that one should set d as small as possible, since both $\sqrt{\frac{N}{d}}$ and $(1 - \frac{1}{d})^{d-1}$ increase as d decreases. The optimal value turns out to be 4 (hexagonal tesselation).

2.3.4. CARRIER-SENSE SCHEMES

To reduce the level of interference caused by overlapping packets, one can allow the nodes to sense the channel. Based on the information gained in this way about the state of the channel (idle,busy), the node takes a particular action. This is the idea of Carrier Sense Multiple Access (CSMA) protocols. The various protocols presented below differ by the action (pertaining to packet transmission) that a node takes after sensing the channel.

The simplest example is the *nonpersistent* CSMA ([Klei75a]). The idea is to limit interference by always transmitting a packet (at some later time according to the retransmission delay distribution) by a node once that node sensed the channel to be busy. The other "extreme" is to never let the channel go idle. If a node senses the channel to be busy, it waits until the channel becomes idle ("persists") and then transmits. This is the idea of a *1 – persistent* protocol.

The above protocols differ by the probability of not rescheduling a packet which finds the channel busy upon arrival. If two or more nodes sense the channel

busy under a 1-persistent protocol, as soon as the channel becomes idle, they all try to transmit and thus interfere. To limit this interference, one would like to randomize the starting time of these transmissions. This can be achieved by letting a node "persist" (transmit as soon as it senses the channel idle) with probability p. This is the idea of a $p-persistent$ protocol ([Klei75a]), which is the generalization of a 1-persistent protocol described above. The parameter p is chosen in order to reduce the interference while keeping the idle periods between any two consecutive non-interfering transmissions as small as possible. If the packets are of the same length, a slotted version of these protocols can be considered in which the time axis is divided into slots, and packets can only be transmitted at the beginning of a slot.

Kleinrock and Tobagi ([Klei75a]) analyzed the above protocols under the assumption of an infinite number of users who collectively form an independent Poisson source with a mean packet generation rate of λ packets per unit of time. This is an approximation to a large population in which each user generates packets infrequently and each packet can be successfully transmitted in a time interval much less than the average interarrival time. Clearly, if T is the time to transmit a packet then the throughput per node is $S = \lambda T$ and is less than 1 because of interference.

The basic equations for the throughput S are expressed in terms of a (the ratio of propagation delay to packet transmission time) and G. The Figures 2.3.4.1 and 2.3.4.2 illustrate some comparisons between different protocols.

While the capacity of the ALOHA channels does not depend on the propagation delay, the capacity of a CSMA channel does. An increase in a increases the "vulnerable" period of a packet - the period over which interference can occur (Figure 2.3.4.1). It is not surprising, therefore, that capacities of non-persistent and p-persistent schemes are more sensitive to increases in a than the 1-persistent scheme. For larger a, non-persistent CSMA drops below 1-persistent. For large a, ALOHA becomes superior to any CSMA schemes. This is because any decisions that a node wishes to make under CSMA are based on obsolete data.

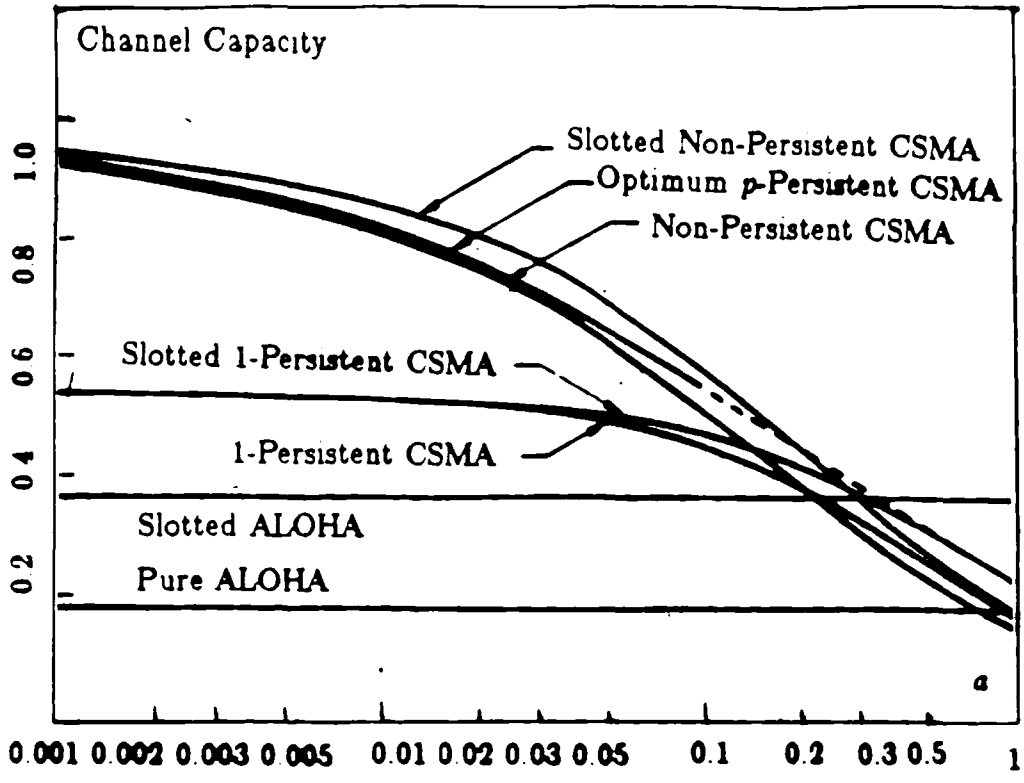

Figure 2.3.4.1: Effect of Propagation Delay on Capacity

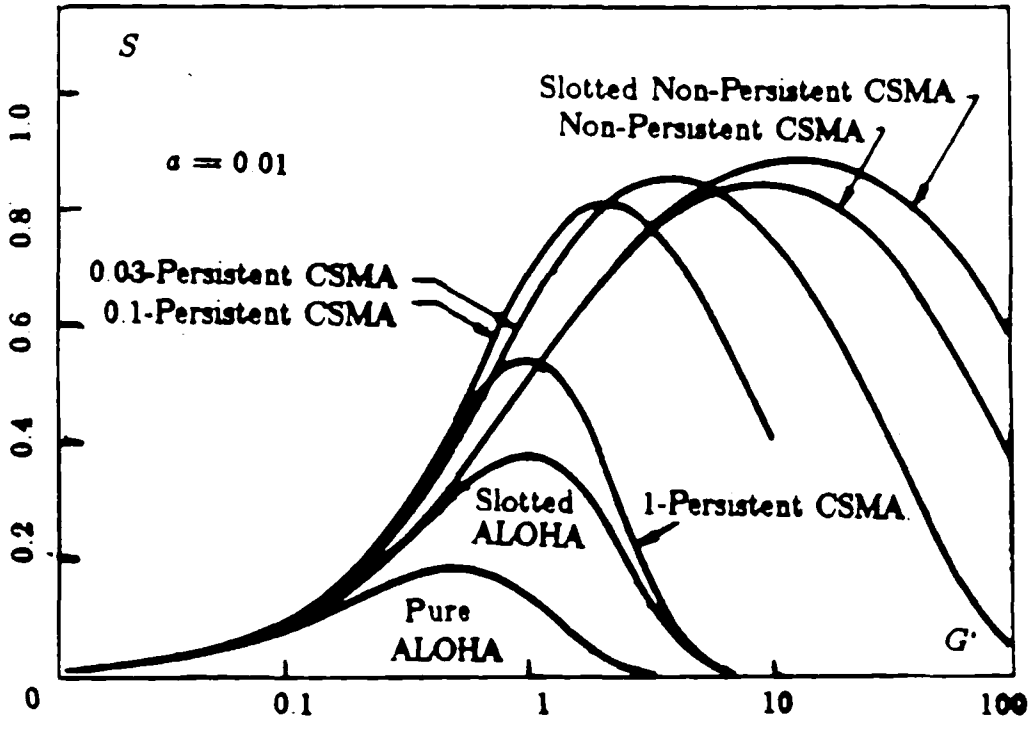

Figure 2.3.4.2: Throughput for Various Schemes (a=0.01)

The above modes deal with the case when all the nodes are within range and in line-of-sight of each other. However, this may not always be the case. It is possible that a node may not be able to "hear" all the other nodes' traffic. Not surprisingly, in such a multihop case, the analysis becomes much more complex. The reason is that the interference in a multihop environment comes not only from the transmission overlap but also from the so-called ***hidden*** terminals ([Toba75]). An idle channel around the transmitter does not necessarily imply that the channel around the intended receiver is also idle. This is illustrated for a simple multihop network in Figure 2.3.4.3. If node A senses an idle channel and sends a packet to node B, the packet will suffer a collision if node D is transmitting. Node D is *hidden* from node A because it is outside of the hearing range of A. The hidden terminal problem causes additional interference and thus decreases the throughput achievable using CSMA.

Tobagi and Kleinrock ([Toba75]) analyzed the performance of the above CSMA protocols in such an environment. A network configuration is specified by a "hearing" matrix: if there are N nodes, this is an $N \times N$ matrix A such that the element $a_{ij} = 1$ if and only if i hears j and is 0 otherwise. To simplify the description, one can partition the nodes into groups: all terminals within the same group hear exactly the same subset of terminals in the population. The input processes for each group are assumed to be Poisson. The solution for a general case is intractable. A simple case when the population can be partitioned into M independent groups of equal size can be analyzed. For each node, a fraction of population is hidden, namely $(M-1)/M$. The channel capacity for various values of M is given in Figure 2.3.4.4.

As seen from that figure, channel capacity experiences a big decrease between the case of $M = 1$ (no "hidden" terminals) and $M = 2$ (additional interference from "hidden" terminals). The decrease is more critical for the non-persistent CSMA than for the 1-persistent. For $M \geq 2$, slotted ALOHA performs performs better than CSMA. For $M > 2$, the channel capacity is rather insensitive to the

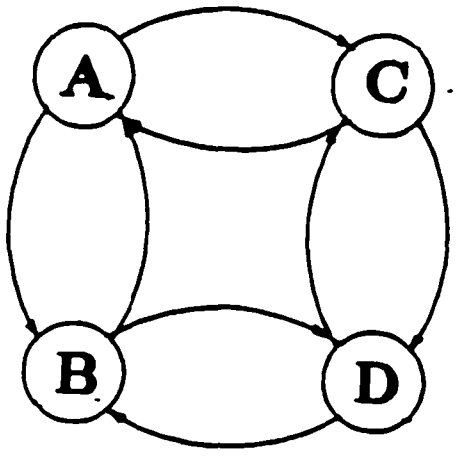

Figure 2.3.4.3: Simple Multihop Network

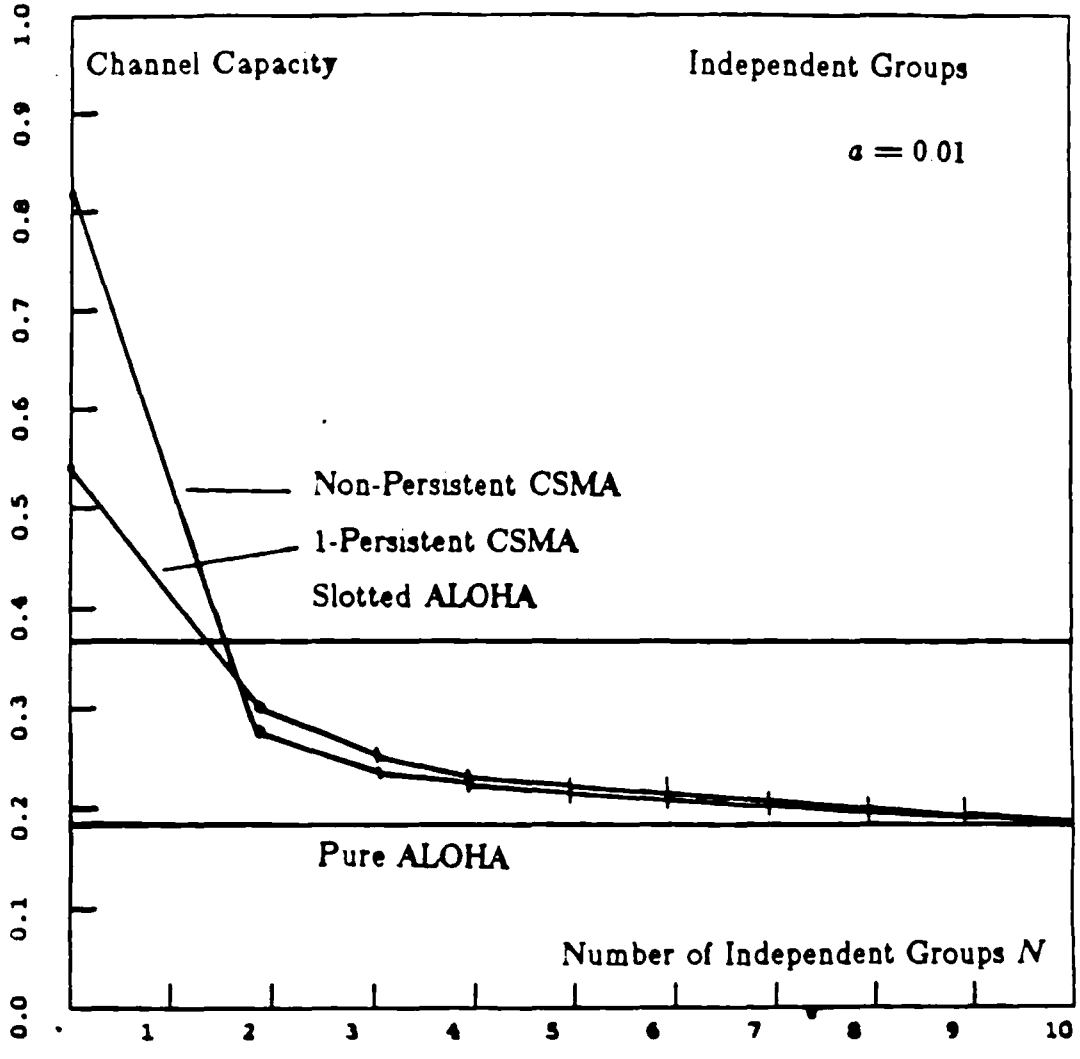

Figure 2.3.4.4: Channel Capacity vs the Number of Groups

number of groups and approaches Pure ALOHA, as one would expect.

The problem of interference from "hidden" nodes can be alleviated with the help of a busy tone transmitted on a separate frequency by the receiver. This is the idea of BTMA - Busy Tone Multiple Access protocols ([Toba75]). Under the assumptions of zero propagation delay, the busy tone guarantees correct reception of the original packet and prevents the additional interfering transmissions. Because of lower interference one can expect that the throughput of BTMA is higher than that of CSMA. Several BTMA protocols exist depending on who is transmitting the busy tone ([Braz83]). In the C-BTMA (conservative BTMA) any node that senses a carrier emits a busy tone. Thus, if node A transmits a packet to node B, then all neighbors of A transmit the busy tone, thus blocking all nodes in the region within twice the hearing radii from node A. In the I-BTMA (Ideal BTMA) the receiver node B transmits the busy tone, blocking only its neighbors. This scheme is not easily implementable since B must know a priori that it is the intended receiver. Some examples of BTMA will be considered below.

A general model to analyze various CSMA schemes in multihop environments has been developed recently by a number of researchers ([Boor80, Magl83, Braz83]). In the model, one assumes that message lengths are exponentially distributed, and redrawn independently from the corresponding distribution each time a message is transmitted. Receivers capture the first transmission that reaches them (perfect capture) and transmitters always get perfect acknowledgment. One assumes that interference is only due to the contention of the channel, thus each node is assumed to have an infinite buffer space, perfect acknowledgment, zero processing and propagation delays. If transmission fails because of the interference from another node, the packet must be scheduled for a later time. The interschedule times are assumed to be distributed Poisson and it is also assumed that there is always a packet available at each schedule time. Simulation studies ([Boor80]) for Single-hop ALOHA indicate that if rescheduling is delayed more than 10 times the packet length, the assumption is a valid one. Packet lengths are assumed to be

independently reassigned at each node in a path. This assumption is consistent with the rescheduling delay assumptions above. Note that under the above assumptions, the process of scheduling points for each node is Poisson. Let λ_i denote the rate of this Poisson process of scheduling points for node i and let $1/\mu_i$ be the mean length of packets transmitted by node i.

To use such a model to study different protocols, one needs to take into account the transmit status of each node. In the ALOHA schemes, the receive status of each node can be completely ignored, while in CSMA schemes, it is implied from the transmit status of the neighbors. The state-space $X(t)$ at time t is the set of nodes which are transmitting at that time. Since, depending on a particular protocol, not all the subsets of nodes can be transmitting, the state-space is a function of the protocol in use.

For example, for a 4-node chain of Figure 2.3.4.5. the state space S is

CSMA: $S = \{0, (1), (2), (3), (4), (1,3), (1,4), (2,4)\}$

C-BTMA: $S = \{0, (1), (2), (3), (4), (1,4)\}$

ALOHA: $S =$ Power Set of $\{1, 2, 3, 4\}$

Because of the assumption of exponential message lengths and the Poisson nature of the scheduling point process, it can be shown that $X(t)$ is a continuous reversible Markov chain, and thus the steady-state distribution has a simple product-form.

If $P(A)$ denotes the steady-state probability of state A, it is easy to show that

$$P(A) = \left(\prod_{i \in A} \frac{\lambda_i}{\mu_i} \right) P_0 \qquad (2.3.4.1)$$

where P_0 is the normalization constant and equals the probability of finding all the nodes idle.

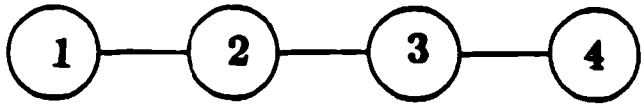

Figure 2.3.4.5: 4-Node Chain

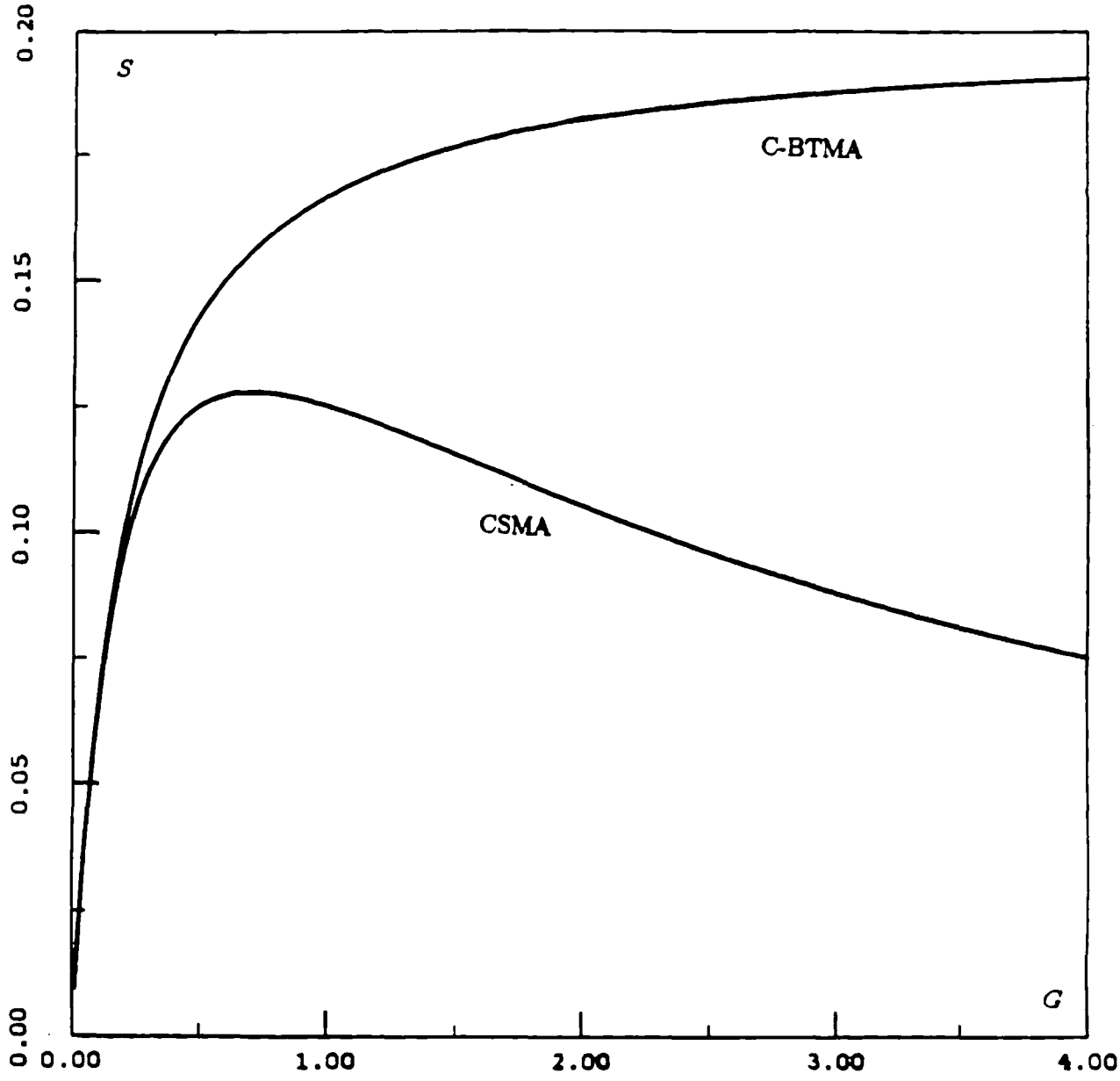

Figure 2.3.4.6: Throughput for CSMA and C-BTMA for 4-Node Chain

For example, for a 4-node chain above, if one assumes the link throughput pattern to be uniform, then the throughput equations become

For CSMA: $S = \frac{G}{1+5G+2G^2}$ with $S_{max} = 0.128$

For C-BTMA: $S = \frac{1}{5}\left[1 - \frac{1}{5G+1}\right]$ with $S_{max} = 0.2$.

The corresponding curves are shown in Figure 2.3.4.6. The tractability of these models for larger systems is tampered by the absence of an efficient computational algorithm to calculate the normalization constant P_0.

The analysis of zero capture for the above model has been presented by Brazio and Tobagi ([Braz84]). Under zero capture, any two interfering packets result in the failure of transmission. The analysis of zero capture is more difficult. Due to the dependency that exists between the message length and the success of its transmission, the average transmission time of a successful packet is not $1/\mu_i$. The calculation of the link throughput from i to j involves the construction of a Markov chain representing the states of the network under which the i-to-j transmission results in success. Although it can be done in principle, it seems that the exact solution based on such a construction is infeasible even for very simple networks.

For multihop systems, one can gain some throughput by transmitting even if a node senses the channel to be busy. This is the idea of Rude-CSMA ([Nels85]). The motivation for this is that a busy channel around the transmitter does not necessarily imply that the intended receiver also senses the channel to be busy. This can be explained with the simple example in Figure 2.3.4.4. Suppose node A has a packet for node B. If node A senses that the channel busy, the transmission could be from node C. Since node C lies outside of the hearing range of node B, transmission from A to B (despite the busy signal sensed by A) will be successful. To create the mathematical model, one takes the general model of Boorstyn ([Boor80]). And, in addition, one assumes that each node i knows the number N_1 of its neighbors who are transmitting and the number N_2 of its idle neighbors. If λ is the rate at which packets are generated at node i for transmission, then the rate λ_i at which node i

transmits is given by

$$\lambda_i = \lambda x^{N_1} y^{N_2} \qquad (2.3.4.2)$$

where x and y are static network parameters. If the state of the neighbors is of no relevance, then $x = y = 1$, and the resulting protocol is our old friend the ALOHA protocol. If only the transmitting neighbors are relevant ($x = 0, y = 1$), one obtains the CSMA protocol. Thus, Rude-CSMA can be thought of as a generalization of these two protocols.

The basic problem in the analysis of this protocol is to find the optimum values of x, y, λ to achieve maximum throughput. This is done numerically and only for very simple networks (a 6-node grid and a 7-node random network.) The optimal performance was obtained when the protocol was CSMA with optimized channel input rates (found numerically). Thus, for practical networks, the Rude-CSMA should not be "rude". We should add here that even if Rude-CSMA were to perform better for some x, y, it would be difficult to implement a system where a node knows the status of all of its neighbors.

Brazio and Tobagi ([Braz85]) have used the general model of Boorstyn ([Boor80]) to analyze the throughput of spread spectrum multihop packet radio networks. The bits 0 and 1 in the data are encoded with specified sequences of N binary digits (chips). The receiver is equipped with matching filters that allow the detection of the presence of the code waveforms of the data bits. Each packet transmission has a preamble which the receiver uses to acquire bit and packet synchronization ([Kahn78]). Note that the conditions of success depend on the ways codes are selected. If all the nodes use the same pattern, then any two overlapping transmissions interfere. On the other hand, if they are all different, the interference will be confined to preambles only. With the spread spectrum operation, one can introduce the Destination Code Sensing Multiple Access (DCSMA). Under DCSMA, the source node of a link monitors the channel for the existence of transmissions using the code assigned to the destination of the packet, prior to transmitting it. The link is blocked if, at a scheduling point, any such transmission is detected. Note

that the reception of the packet may have to be aborted if a scheduled transmission is to take place. The Markovian model is then formulated, from which one can obtain the throughput equations. This produces some numeric results are presented for very small systems. Analysis of the spread spectrum multihop networks is in its very early stages.

2.3.5. CONCLUDING REMARKS ON MULTIPLE ACCESS SCHEMES

For many models, a number of protocols for multi-hop networks do not lend themselves to a simple product form solution, as for example in ALOHA schemes, where a node is inhibited from starting a transmission if it is receiving a packet or the I-BTMA. The question of the existence of a product form solution for a variety of CSMA protocols for the above model of multihop networks was addressed by Brazio and Tobagi ([Braz84]). The neccessary and sufficient condition for a protocol to possess a product form solution turns out to be the "symmetry" of interference: for all pairs of used links i and j, link j blocks link i whenever link i blocks link j.

The schemes that do admit a product-form solution can be numerically analyzed in practice only for very small and simple networks. Many of these schemes were originally designed for simple hop systems and it is not yet clear how to compare them for large multihop packet radio networks. New approximation techniques to analyze such schemes are clearly required.

2.4. LOCKING SCHEMES FOR DATABASE SYSTEMS

2.4.1.GENERAL OVERVIEW AND PERFORMANCE ISSUES

The last class of systems we will consider here is database systems. A

database consists of a collection of items shared by many users. These users update the database with transactions. Concurrent transaction processing is a means of improving system performance and resource utilization. To process transactions concurrently, the database management system must control the read and write transactions so that they do not produce incorrect results (deadlock or inconsistent state). Thus, a concurrency control mechanism for locking must be used. It must guarantee the user that transactions will perform the same computation as they would in a serial environment (serializability). Clearly, the concurrent processing of transactions creates a problem of interference. Locking, while essential in concurrent transaction processing, creates interference among transactions accessing shared item(s). This clearly impairs concurrency. In addition, an implementation of a concurrency control mechanism, essential as it is, creates a noticeable overhead which makes it difficult to analyze the impact of interference on concurrency.

What are the key performance measures in analyzing the impact of interference in database systems ?

- The most important performance measure is the average number of concurrent transactions. Another related measure is the multiprogramming level - the maximum number of concurrent transactions.

- The blocking probability that an arriving transaction can get the required data items.

- The optimal unit of locking (granularity of locking).

The impact of granularity on interference and concurrency for a particular concurrency control mechanism is a very important question. Fine granularity decreases the interference and thus allows a higher degree of concurrency. In the extreme case, if a granule corresponds to a record (the smallest addressable data), then interference is minimal, but the database system must be prepared to handle a lock table with the same number of entries as records in the database. On the other hand, course granularity decreases both concurrency and the overhead of managing

the locks. Again, in the extreme case, if a granule corresponds to an entire database, then only one transaction can be active at a time and the database system will have to keep track of only one lock.

Another issue related to granularity is the distribution of lock requests to the lockable units of the database, and its effect on interference. Few experimental studies have been conducted to determine the spatial reference pattern according to which transactions reference the items in a database ([Smit78]). It seems that items referenced by a transaction are usually clustered in several localities and sequentiality of access is an inherent characteristics of most systems. Because of the lack of agreement upon reference patterns and the mathematical tractability requirements for analytical studies, rather simple reference patterns are usually considered. One example is where access requests are equally likely distributed over entire database ([Lang82]). This question of "interference locality" seems to have received far less attention here than in the case of multiprocessor interconnection networks discussed above.

To analyze database performance, one must analyze interference in the context of a particular concurrency control mechanism. There has been considerable research on concurrency control schemes and there are a number of ways to classify them ([Bada80, Bern81, Fran84]). Although numerous concurrency control mechanisms have been implemented (for a good survey see [Bern81]), there is an obvious lack of analytical studies. Work in this area is in its preliminary stages.

2.4.2. ANALYTIC MODELS

Perhaps the first analytic models to investigate the trade-off between locking granularity and increased concurrency were proposed by Irani and Lin ([Iran79]). The database is assumed to consist of D units of physical blocks, a number of granules H, a multiprogramming limit N, and mean transaction time T. With such

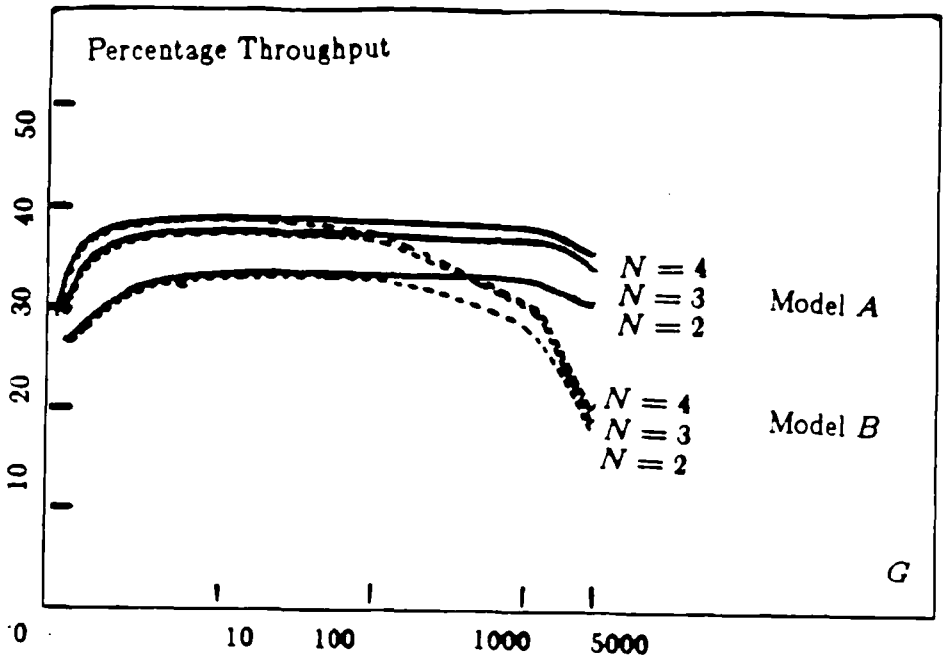

Figure 2.4.2.1: Percentage Throughput vs Locking Granularity

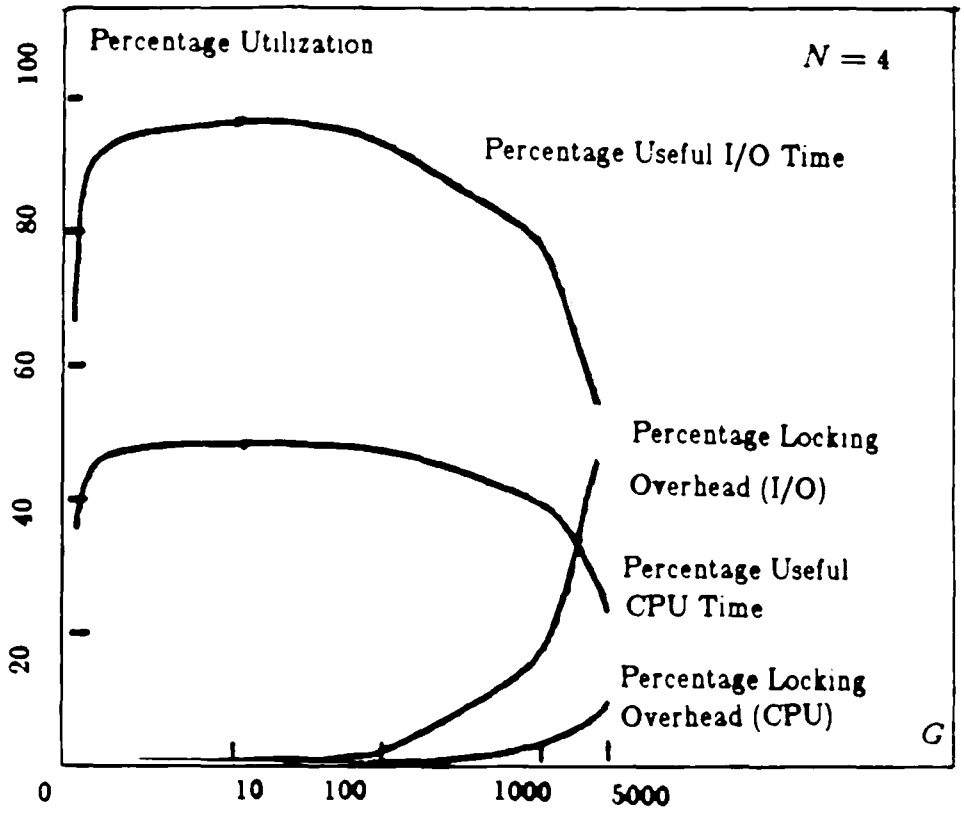

Figure 2.4.2.2: Percentage Utilization vs Locking Granularity

definitions, the locking granularity $G = D/H$. The mean number of lock requests issued by a transaction is $L = T/G$. The transaction processing is modeled using the class exchange mechanism of the BCMP networks. The authors consider three types of service centers: the CPU, I/O units, and pseudo service center. The pseudo service center models the waiting time for a blocked transaction until a lock is released. Two different models are then considered. In the first one, the granularity is assumed to be so coarse that the lock table is small enough to be kept in core. The probability that a locked request is blocked is

$$P = \frac{L(N-1)}{2H} \qquad (2.4.2.1)$$

The second model considers finer granularity and assumes that the lock table is stored on disk. To discuss the trade-offs between granularity and performance and to be able to carry out numerical calculations two assumptions are made. First of all, the mean time until lock release for a blocked transaction is assumed to be a constant - it is independent of the level of multiprogramming level N and granularity G. Although this time should clearly depend on the concurrency in the system and the granularity of locking, the authors justify the "constant" time assumption by simulation and empirical evidence. Secondly, the probability that a lock request is blocked is assumed to be inversely proportional to the number of granules. Clearly, this makes a big difference in the overhead associated with each transaction. Secondly, the models ignore the overhead for accessing the list of waiting transactions for queueing and dequeuing. The results are what one would intuitively expect.

For large transactions that access too many items, fine granularity is too expensive since it requires higher locking overhead. On the other hand, coarse granularity results in a loss of concurrency since many small transactions would require very few items. This is shown in Figures 2.4.2.1 and 2.4.2.2. If the number of items required by transactions varies in size and some transactions require many items, coarse granularity is appropriate. A number of interesting conclusions can be drawn from the results of this study. To maximize the throughput, a rel-

atively coarse granularity should be chosen. Locking overhead is very costly for fine granularity. If the locking granularity is fixed, increasing the degree of multiprogramming can always improve the system throughput. It is interesting to note that the results of this study agree well with the simulation results of Ries and Stonebraker ([Ries77, Ries79]), who assumed that the amount of service required by a transaction is proportional to the number of locks.

Potier and Leblanc ([Poti80]) have given a quantitative analysis of two classes of locking policies. The first class is the static lock policy, under which all the locks necessary for transaction are obtained in one atomic action, and are all released upon completion. The second class is the Dynamic Lock policy, where the locks are acquired and released as they are required. Note that under dynamic acquisition, deadlocks and inconsistent states are possible. On the other hand, the interference and the locks' mean times are clearly shorter than under the static rule. If the part of the transaction during which locks are obtained is short, then the static lock policy is a close approximation to the dynamic one. The above conclusions are based on the assumption that each transaction makes the same number of accesses n, which are also assumed to be independent and uniformly distributed. The model is solved by finding the probability of being blocked. The results for the throughput for different n and granularity G under static locking policy are shown in Figure 2.4.2.3.

The throughput exhibits a minimum for each n, since in this model the mean number of locks $\nu(G)$ tested on each access is proportional to granularity G. One can argue, therefore, that these kinks in the throughput curves indicate the importance of the no-locality assumption that is indicated by the linear function $\nu(G)$. Transactions with a large number of accesses very rapidly tend to lock all the files, thus increasing interference and preventing activation of the new transactions. This effect is partly alleviated by using finer granularity, although, as it has been pointed out, this leads to an increase in locking overhead. Since two large transactions are more likely to interfere than two smaller ones, it is not surprising that

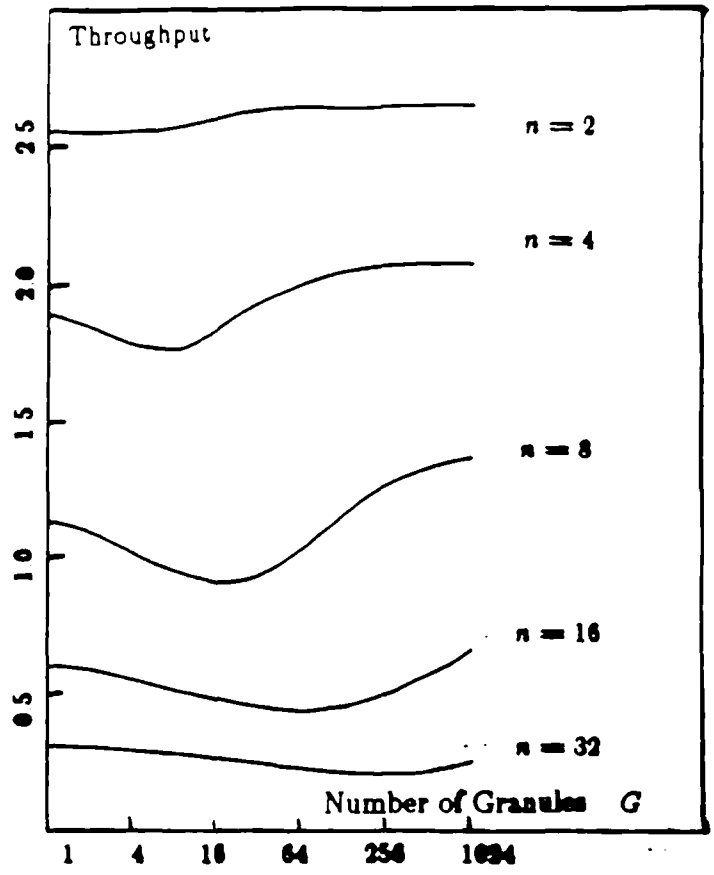

Figure 2.4.2.3: Throughput vs Number of Granules

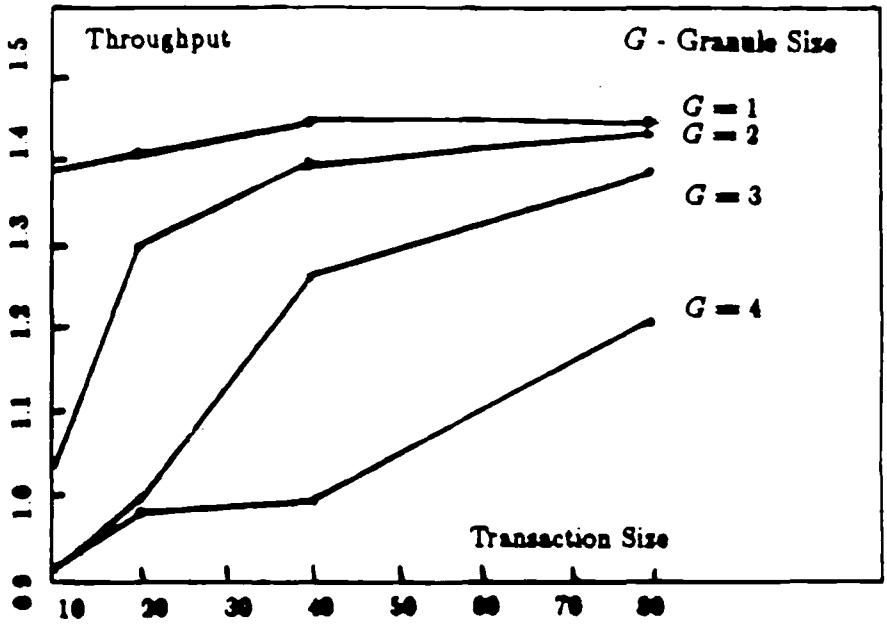

Figure 2.4.2.4: Throughput vs. Transaction Size

increasing the granularity tends to increase the throughput more for the database with larger number of accesses.

A different analysis for static locking was considered by Galler ([Gall82]). He analyzed both centralized and distributed database systems. His model is a single class queueing model consisting of N terminals generating access requests to a database of size D and of granularity G. The total number of concurrent accesses is limited by some multiprogramming level (certainly no more than D/G). The model is described by several equations of the form $W = f(W, D, G)$ where W is the expected waiting time. The equations can be solved by successive substitution and can be proven to converge when modelling many real systems. The accuracy can be expected to be higher than that from the previous approach of Potier and Leblanc ([Poti80]). In that model, the control parameter used in the calculation is the maximum number of blocked transactions resubmitted after a transaction leaves the system. That parameter is important to calculating the throughput. Unfortunately, this number is difficult to measure and a bad estimate could lead to erroneous results.

The throughput and the response time can be calculated from the expected waiting times using simple formulae. An example of throughput comparison for different transaction sizes and granularity is given in Figure 2.4.2.4. For finer granularity (smaller G), the probability of interference conflict is small. Not surprisingly, the throughput is higher the smaller the granularity G. For a fixed value of G, the increase in the throughput for larger transaction sizes is noticeable for coarser granularity (larger G). The results are similar to those obtained by Potier and Leblanc ([Poti80]). The analytic results best agree with the simulation when the waiting time for a lock is deterministic and the transactions require only one lock. Therefore, the applicability of the model seems limited.

The above studies considered mostly static locking policy. The analytic study of Shum and Spirakis ([Shum81]) describes some forms of the dynamic 2-phase locking. It investigated three basic problems: First, the non-trivial lower

bounds for the steady-state probability P that a transaction will complete without restarts (deadlocks); second, an upper bound on the average number n of restarts per transaction, and third, the mean response time R of a transaction (including restarts). The model assumes a database of M items (granules). The arrival time distribution of transactions is assumed to be Poisson with rate λ. Transactions are characterized by "hopping behavior" - each transaction moves from one item to the next item with probability $(1-\theta)/M$. On each visit to the granules, transactions obtain service for a time period distributed exponentially with parameter μ. Worst case lower bounds on the performance are obtained by assuming a simple deadlock prevention 2-phase locking algorithm.

A number of analytic bounds for P, n, R are obtained, given $\lambda/\mu, M, \theta$. For example, the probability of completion with no restarts is

$$P = 1 - \frac{1-\theta}{\theta^2 M}\frac{\lambda}{\mu} \text{ for } \frac{\lambda}{\mu} < \frac{\theta^2 M}{1-\theta}, \quad M \geq 100 \tag{2.4.2.2}$$

The average number of restarts is

$$n = \frac{1}{P} - 1$$

The response time for $M > 100$ is approximated by

$$R = \frac{1}{P\theta\mu}$$

For $M < 100$, the computational effort is acceptable and can be used to obtain an estimate of P. For the more general case of 2-phase locking where deadlocks are allowed, the rate of deadlocks is proportional to the average number of concurrent transactions.

Another dynamic locking scheme was considered by Chesnais, Gelenbe and Mitrani ([Ches83]). In their model, a transaction requires items which are locked until the transaction either completes or aborts. If the required item is locked by another transaction, it waits for a random period of time before trying again. A transaction that is still unsuccessful on the $L+1$-st attempt is aborted

immediately and restarted from the beginning. The time required to process an item and time to process a transaction is assumed to be distributed exponentially. Finally, the multiprogramming level is assumed to be constant: if one transaction is completed, a new one enters the system. Since the behavior of a transaction is influenced by that of all other transactions operating on a database, the number of states to describe the system will be so large that exact numerical computation is prohibitively expensive for any problem of reasonable size. Therefore, the following approximation is suggested. The approach is to treat an individual transaction in isolation and then analyze it in steady state. The influence of other parameters is reflected in the analysis by the probability F that an item requested by a transaction is unavailable. Assuming that all transactions are statistically identical, one can derive the equation for F in the form $F = \phi(F)$. The authors then show that an admissible solution exists. The question of the uniqueness of the solution is very difficult and is treated only for some special cases. For example, if the time necessary to commit or abort a transaction is negligible, the solution is unique and corresponds to the case $L = 0$. This means that a transaction should restart whenever interference is encountered. It is encouraging to note that the results of simulations are within 10% of estimated values.

Tay ([Tay84]) has analyzed both static and dynamic locking policies for a centralized database model using mean value analysis. To apply this method, a number of simplifying assumptions are made: The total number of concurrent transactions is assumed to be fixed. When a transaction finishes, it is replaced by another one. If the next cannot proceed because some of the locks are not available, it is held back for a period long enough to let the conflicting transactions leave the system. While the restarted transaction is kept out of the system, a different transaction begins processing. Note that under mean-value analysis, the number of transactions is kept constant by assuming that there is always a transaction which can proceed. This is equivalent to saying that a restarting transaction requires different items from the previous time. The access pattern is assumed to be uniform.

This means that a transaction requests any item with the same probability. This assumption can be relaxed by dividing the database into two parts with different access probabilities for each part of the database. This allows the modelling of situations where some items of the database are accessed almost all the time.

The author considers two cases: the no-waiting case where the conflicts are resolved by restarts, and the waiting case where conflicts are resolved by blocking. The analysis proceeds by writing the functional equations involving the number of transactions N, a database of size D, the transactions length k, and "load" $\lambda = N/D$. The objective is to determine the throughput for a given N.

An example of the throughput comparison for waiting and non-waiting cases is presented in Figure 2.4.2.5. As long as the multiprogramming level N is small, the throughputs are almost identical - interference is not very likely. Increasing N increases the potential number of conflicts (increase in interference) and this will tend to decrease the throughput for the waiting case. This is precisely what is described by the curves referred to above.

Morris and Wong [Morr84] presented some models to compare locking and optimistic concurrency control for the cases of both exclusive and nonexclusive access. Here, a database consists of N items. An arriving transaction requires exclusive locks on w items (the "write" set of data items) and non-exclusive (the "read" set of data items) locks on r items. The more restrictive assumption of only exclusive access can be analyzed by taking $r = 0$. Each transaction has exponentially distributed service time requirements with mean $1/\mu$. The authors consider two classes of concurrency control schemes: locking schemes and optimistic schemes. In a locking scheme, after admission to the system, a transaction declares all its read and write sets. If it does not interfere with any runnable transaction, it starts the execution. If there is interference, it is blocked. When a transaction finishes, all blocked transactions are checked in FCFS order to see if they can proceed. In an optimistic scheme, an admitted transaction begins execution immediately, making all of its writes to temporary copies of the data items. When the transaction

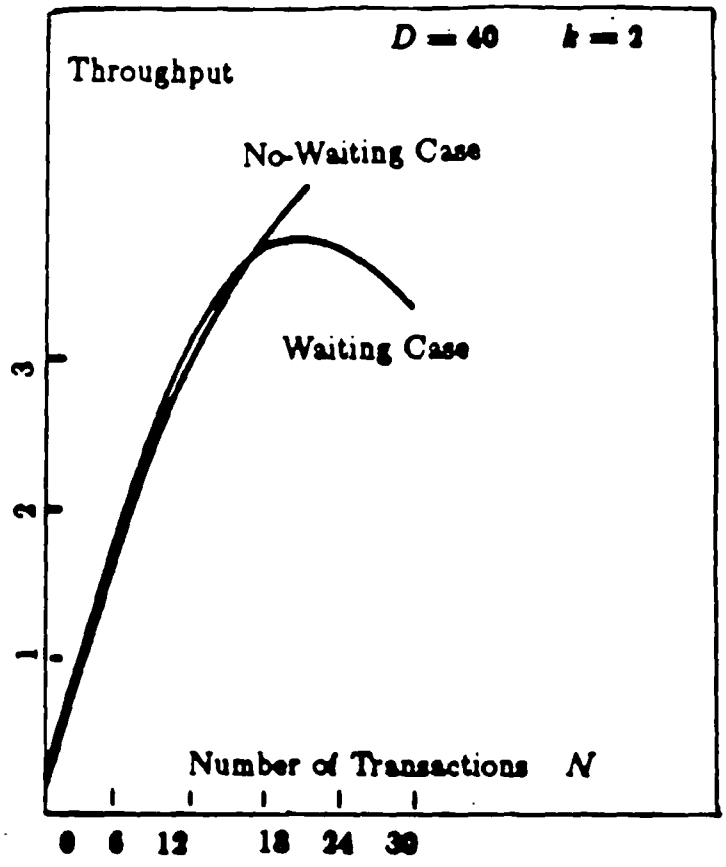

Figure 2.4.2.5: Throughput for Waiting and No-Waiting Case

finishes, a check is made to see if any of these items have been modified by any departed transaction. If not, the completing transaction makes its writes/outputs permanent (i.e. *commits*) and departs from the system. Otherwise, it aborts and must try reaquiring the same data items at a later point in time. The primary performance measure for both schemes is the maximal throughput $T(M)$ when the multiprogramming level (total number of running and blocked transactions) is a constant M.

An exact analysis of these schemes is intractable since one must keep continuous record of the exact access sets of every runnable and blocked transaction. However, if a blocked transaction is allowed to independently redraw its access sets, the model becomes tractable while retaining a high level of accuracy. With such an assumption for a locking scheme, one can recursively compute the probability $p(i)$ that there are i customers running in the system (multiprogramming level M). If $\mu(i)$ denotes the corresponding service rate function, the maximal throughput can be evaluated as follows

$$T(M) = \sum_{i=1}^{M} p(i)\mu(i)$$

A rough upper bound for $T(M)$ can be computed in $O(M^3 + Mw^2)$ operations. On the other hand, for an optimistic concurrency control scheme, $T(M)$ is computed to be of the form

$$T(M) = \max\left(\frac{1 + 2(M-1)\psi - \sqrt{1 + 4(M-1)\psi}}{2(M-1)^2\psi^2}, \frac{1}{M}\right)\mu(M)$$

Here ψ is the probability that a departing transaction caused another transaction to abort.

The above two approaches to concurrency control are compared by simulation for two choices of $\mu(i)$: (1) $\mu(i) = i$ and (2) $\mu(i) = i/(i+5)$. An example of

the first model is a balanced central-server model with the number of disk drives much greater than the maximum multiprogramming level. An example of a second system is a balanced central server model with 5 disk drives. The simulation results track the analytic results very closely. For the above models, the results show that the locking scheme consistently outperforms the optimistic scheme over a wide range of parameters. The advantage of the locking scheme is more evident in higher interference cases. Not surprisingly, the results also indicate that the use of a non-exclusive locking policy offers considerable improvement over the exclusive locking policy. Both schemes were analyzed under the assumption of fixed size of access sets. A more complete characterization of models in which the locking scheme outperforms the optimistic scheme is an important topic for further research.

The analysis of the impact of interference on concurrency for various general classes of scheduling policies independent of any transaction processing system was considered by P. Franaszek and J.T. Robinson ([Fran85]). The primary performance measure here is the effective level of concurrency $E(N,p)$, defined as the expected number of the N transactions that can run concurrently and do useful work, where p is the probability of conflict among transactions. In contrast, most of the previous analytic work discussed above have concentrated mainly on analysis and simulation of relatively detailed models of systems. It can be difficult, therefore, to analyze the effect of interference independent of other system characteristics.

The authors suggest the following model. The multiprogramming level is set to be N - there are always N transactions in the system. If a transaction does work leading to its completion, it is said to be *active* ([Fran85]). If a transaction is doing work, which will be aborted due to interference from any other transaction, it is said to be *inactive*. An interference conflict occurs between any two concurrent transactions with probability p. At the end of each unit of time, all active transactions finish and are replaced by the new ones. In the new time unit, if two transactions were also in the previous unit, then there is interference conflict if and only if there was a conflict in the previous time cycle. The set of interfering trans-

actions can be modeled by a random graph, whose vertices represent transactions, and an edge between two transactions indicates an interference conflict between the two.

Using this model, the authors consider three general classes of concurrency control policies:

• *Priority-less.* In this type of policy, any transaction can be blocked by any other transaction. An example is the standard locking policy, where locks are acquired incrementally ([Gray78]). For this policy, they show that

$$E(N,p) \leq N\left(1-\frac{p}{2}\right)^{N-1} \qquad (2.4.2.3)$$

This implies that for the standard locking methods, $E(N,p) \mapsto 0$ for fixed p and $N \mapsto \infty$.

• *Strict Priority.* In this type of policy, given an interference conflict between two transactions, the higher priority transaction becomes active and blocks the other one. An example of such a policy is the wound-wait method ([Rose78]) - a transaction with lower priority is made to wait (if possible) or aborted. For this policy, the authors show that

$$E(N,p) \leq \frac{1-(1-p)^N}{p} \qquad (2.4.2.4)$$

• *Essential Blocking with Priority.* In this type of policy, a transaction is blocked if and only if it interferes with a transaction of higher priority that is active (doing useful work). An example is the optimistic concurrency control methods. For this policy

$$1+\frac{(1-p)}{p}\log(p(N-1)+1) \leq E(N,p) \leq 1+\frac{1}{p}\log(p(N-1)+1) \qquad (2.4.2.5)$$

If p is fixed and N is increased, the expected number of active transactions is unbounded but still of $O(\log N)$. This is in contrast with the strict priority scheduling in which the expected number of active transactions is bounded by $1/p$. On the

other hand, it results in very high costs in terms of wasted work if it is obtained using the optimistic methods.

The predictions of the random graph model and the above expressions for $E(N,p)$ are confirmed by simulations of an abstract transaction system. Even if one can achieve an increase in $E(N,p)$ using conflict-dependent scheduling, no dramatic increase can be expected if the interference conflicts occur independently. All of the above results are of practical importance only for fairly large N or p. Even for the essential blocking with priority policy, to achieve a linear increase in $E(N,p)$ by increasing the number of processors, one needs an exponential increase in the number of concurrent transactions. This means that distributed database systems with a large number of processors must be designed such that p decreases with N or that conflicts do not occur independently. This can be used to an advantage. Simulation results show that a number of new scheduling policies perform better than known methods. A precise analysis of these methods is a subject for future work.

2.4.3. HIERARCHICAL DECOMPOSITION APPROACH

One of the main difficulties in the performance analysis of database systems is relating the available solution techniques for queueing networks with different strategies for database design at both the logical and physical levels. An overall hierarchical analytic framework of assessing and predicting performance measures of a variety of physical and logical database decisions has been suggested by Sevcik ([Sevc81]). At the lowest level, one has a queueing network model. From the input parameters (transaction sizes, granularity, statistics on the arrival rates, and service time of transactions) one obtains the estimates for total storage requirements, the necessary file structures, and so on. From these estimates, one can obtain the next level of this analytic framework, namely the specification of the necessary physical

database. And so on. By analytical techniques, the workload description at one level and the set of design choices is transferred into the workload description of the more fully specified next level. The highest level is, of course, the logical database design.

Such a decomposition solution method to evaluating a centralized database with static locking was considered by Thomasian and Ryu ([Thom83]). They analyze the database system using the hierarchical decomposition method, where the highest level model yields the mean user response time. From the viewpoint of the multilevel modeling approach proposed in [Sevc81], mentioned above, this work presents a breakdown of the highest modeling level for computing user-oriented performance measures such as mean response time. (Because the arrival process is Markovian, the highest modeling level can be represented by a one-dimensional birth-death process and analyzed very easily.) It extended the work of Potier and Leblanc [Poti80] by allowing more than one transaction to be considered for activation. This permits the incorporation of the scheduling discipline into the model. The authors analyzed two variations of FCFS - FCFS with skip (the scheduler concludes its scan after the first interference is detected) and FCFS without skip (the scheduler scans all of the transactions in the pending queue). The queueing network model is assumed to have a product form solution with a single job class. The authors consider two cases: resampling of the locks - the locks requested by pending transactions are resampled each time the transaction is checked for activation and no resampling of the locks. The obtained analytic expressions indicate that the differences in throughput are insignificant for both FCFS with skip and FCFS without skip. This is shown in Figure 2.4.3.1. For small transaction sizes n, there are no significant differences in throughput with resampling and without resampling.

This can be explained by the fact that for small transaction sizes, the probability of interference conflict is very small. As one increases the transaction size, the resampling case should give better throughput since the transaction does

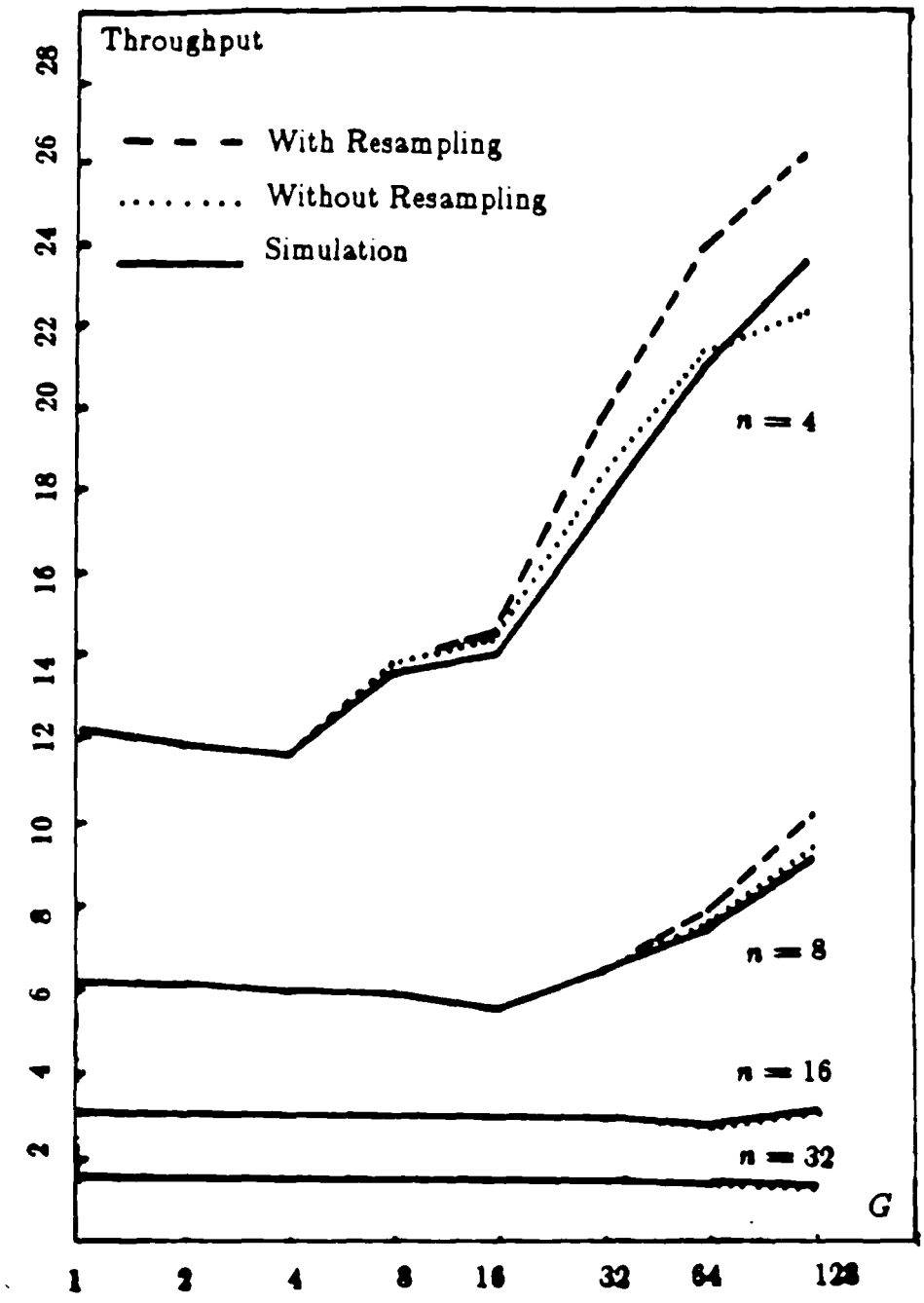

Figure 2.4.3.1: Throughput vs. Granularity of Locking (FCFS with skip)

not persist on the same specified locks but rather requires a different set of locks each time it is checked for activation. Not surprisingly, a finer granularity allows higher concurrency in the system.

2.4.4. EXACT MARKOV MODELS OF STATIC LOCK POLICY

As one can see from the previous discussion, analytic models of locking behavior adopt to some extent the approach of using the steady state average values. One exception to this is the exact model of a static lock policy, based on a Markov process. Such a model was analyzed by Mitra and Weinberger ([Mitr84]) and by Lavenberg ([Lave84]). In this model, one considers a database system with N items and p classes of transactions. Transactions in the same class σ require the same number j_σ of items. A transaction of class σ arrives according to the Poisson distribution with rate λ_σ, requires exclusive locks on j_σ items, and holds them for a time distributed exponentially with parameter μ_σ. Locks are obtained in one atomic action. A transaction interferes with many other transactions which contain items in common with it. In real systems, an interfering transaction must wait until the locks are available. To capture this interference exactly, it is necessary to take into account not only the number and type of transactions in process and in the queue, but also their context. The simplifying feature in the model is that blocked transactions are cleared.

The probabilistic model obtained with the above assumptions is a Markov process. Because of the simplifying assumption that blocked transactions are cleared, the model can be described by a time-reversible Markov process and the steady state probability distribution $P(i_1, ...i_p)$ that there are i_σ transactions of class σ is in product form

$$P(i_1, ..., i_p) = \frac{\rho_1^{i_1} ... \rho_p^{i_p}}{Z_N}$$

The global performance averages can be calculated once one evaluates the normalization constant Z_N. Because Z_N has $O(N^p)$ terms, direct evaluation of the normalization constant is out of the question even for small values of N and p. Instead, it is evaluated recursively in $O(Np)$ multiplications. Thus, the analysis is tampered by the difficulty of estimating Z_N in closed form.

In the asymptotic analysis, the following result for the blocking probability P_σ of transactions of class σ is obtained

$$P_\sigma = \left[\sum_{c=1}^{p} \frac{j_c \binom{N}{j_c} \lambda_c}{\mu_c}\right] j_\sigma / N + o(1/N) \qquad (2.4.4.1)$$

It states that the blocking probability of a particular class σ is proportional to the number of items locked by all concurrent transactions, and by the weighted fraction j_σ/N of the database locked by a transactions of class σ. Simulations were done for cases where the analytical models can be solved. Numerical results demonstrate that the above relation remains adequately valid provided P_σ is sufficiently small, e.g. $P_\sigma \leq 0.05$.

The authors consider some extensions where a transaction of class σ requires j_σ exclusive locks and k_σ non exclusive locks. The following asymptotic expression for the blocking probability P_σ can be obtained

$$P_\sigma = \left[\sum_{c=1}^{p} \frac{(j_c + k_c)\binom{N}{j_c+k_c} \lambda_c}{\mu_c}\right] \frac{j_\sigma + k_\sigma}{N} - \left[\sum_{c=1}^{p} \frac{k_c \binom{N}{j_c+k_c} \lambda_c}{\mu_c}\right] \frac{k_\sigma}{N} + o(1/N) \quad (2.4.4.2)$$

The above formula (2.4.4.2) is amenable to an interpretation which is intuitively appealing: Blocking probability for class σ transactions is the product of the number of items locked by individual class σ transactions and the weighted fraction of the database touched by individual transactions, minus the product of

the number of items non-exclusively locked by individual class σ transactions and the weighted fraction of the database touched by non-exclusively held items.

For this model with both read/write transactions, we do not even know whether there is a recurrence formula for the partition function. The system performance measures need to be computed here from the solution of the basic Markov process. Thus asymptotic analysis and simulation appear to be the only tools to analyze the performance of these systems. Exact results do not yet exist in the general case of both exclusive and non-exclusive locks.

Lavenberg ([Lave84]) considered the above model allowing both exclusive and shared locks. A transaction of class σ requires σ_1 exclusive locks and σ_2 shared locks. If n_1 and n_2 are random variables denoting the stationary number of exclusive and shared locks respectively, then to the order of $O(\frac{1}{N^2})$ the blocking probability P_σ for class σ transactions is given by

$$P_\sigma = \sigma_1 E[n_1 + n_2]/N + \sigma_2 E[n_1]/N + O(1/N^2) \qquad (2.4.4.3)$$

The first term, which approximates the probability of blocking due to exclusive lock requests, is the product of the number of exclusive locks requested and the probability that a particular lock is already held. The second term, which approximates the probability of blocking due to shared lock requests, is the product of the number of shared locks requested and the probability that a particular lock is held in the exclusive mode.

If all locks are exclusive, the probability of blocking reduces to

$$P_\sigma = j_\sigma \left[\sum_{c=1}^{p} \frac{j_c \binom{N}{j_c} \lambda_c}{\mu_c} \right] /N + O(\frac{1}{N^2}) \qquad (2.4.4.4)$$

Note that the above equation (2.4.4.4) is identical to the equation (2.4.4.1) obtained by Mitra and Weinberger ([Mitr84]). It is somewhat stronger since it is $O(\frac{1}{N^2})$ instead of $o(\frac{1}{N})$.

One of the shortcoming of both models is the assumption that blocked transactions are lost. In real database systems, the blocked transactions are resubmitted later. Such a case is approximately analyzed as follows ([Mitra84]). One assumes that a blocked transaction is resubmitted after a random delay having mean d. The total offered traffic (new transactions and retransmissions) of class-σ transactions is assumed to be Poisson with rate τ_σ^*. The offered traffic of "new" class-σ transactions is Poisson with rate τ_σ. Under these assumptions, the mean total delay D_σ for class-σ transactions is ([Mitra84]):

$$D_\sigma = \frac{1}{\mu_\sigma} + d\left(\frac{\tau_\sigma^*}{\tau_\sigma}\right)$$

2.4.5. CONCLUDING REMARKS ON DATABASE LOCKING SCHEMES

The limited number of analytic studies have led to different conclusions about the system performance, mostly due to the differences in modeling assumptions and the values of parameters. Even if there existed an agreement on modeling assumptions, the inherent analytic difficulties remain. Allowing transactions to use more than one resource (e.g. item) simultaneously produces models that are difficult to solve using the traditional queueing network approach. In general, these models do not obey the local balance equations and thus the solution is not in the product form. For example, locks may be held for a period of time which may depend on the amount of time spent accessing other resources (e.g. disks). In addition, while some transactions hold locks, others requesting the same locks cannot start processing. Because of these two facts, local balance assumptions are violated and thus global balance techniques or heuristic approximation must be used. Therefore, the models are not amenable to efficient exact solution techniques for product form queueing networks or mean-value analysis.

The analysis of the impact of interference is central to the understanding of the performance of concurrency control mechanisms. Such analysis allows one to select the database parameters (load, size, granularity) to achieve the maximum possible concurrency. While interference is the determining factor in the analytic tractability and the understanding of these systems, the analysis is often difficult and very little work has been done.

2.5. CONCLUSION

The key to analyzing these systems is the ability to capture interference. Relatively few analytic results are known and the techniques one uses today may not be adequate. The complexity of the systems and the nature of interaction and interference make them very difficult to analyze, even with many simplifying assumptions. Where analytical methods succeed, the solutions are often obtained only in the asymptotic cases and, except in some simple cases, are very difficult to interpret.

Clearly, new methods and approximation techniques to analyze interference and its effect on the performance of computer systems are needed.

CHAPTER 3

A MODEL OF INTERFERENCE AND THE METHOD OF CANONICAL APPROXIMATION

This chapter presents a model of interference for a wide variety of distributed systems. The key difficulty in the analysis of the model is the computation of the system partition function. To solve this problem, we introduce the method of canonical approximation to compute the partition function and a number of performance measures in closed form. Exact error bounds are derived. The method is illustrated via a solution of the classical machine interference model.

3.1. A MODEL OF INTERFERENCE

The model of interference presented here is the same as the one given in [Yemi83]. It is described here for the sake of completeness. Consider a set of N distributed agents competing for a shared resource. N will be used to indicate both the set of agents and its cardinality as long as no confusion arises. The terms *agent* and *activity* are used interchangeably. Assume that there are p classes of agents. An agent of class i becomes active according to the exponential interarrival law with rate λ_i, and once active (once it has acquired part of the resource) it uses the resource for a period distributed exponentially with rate μ_i. Assume further that agents may interfere with each other and that two interfering agents cannot be active at the same time. For example, in database systems, transactions (agents) constitute a class if they require the same number of items; they interfere with each other by locking the subsets of the items. In multiprocessor interconnection networks, an established connection (agent) may block (and thus interfere with)

with other connections. In packet radio networks, transmissions (agents) share the broadcast medium and thus may interfere with one another if in progress at the same time. Assume further that if an arriving agent cannot get access to the resource, then it departs without further affecting the system. In the language of queueing theory this means that blocked customers are cleared.

We call such a system an *Interference System*. An interference system can be represented by an interference graph $G =< V, E >$ where the set of nodes V represents possible agents, and an edge between two nodes represents a mutually exclusive interference between them ([Yemi83]). An interference graph of a 2×2 crossbar is given in Figure 3.1.1. An agent is a processor-memory pair. For example, node 12 of the interference graph corresponds to a connection request from processor 1 to memory 2. Agents interfere if they try to access the same memory module or come from the same processor. For example, nodes 12 and 22 are connected by an edge in the interference graph. They correspond to connection requests that interfere over the access the same memory 2.

With the above assumptions, the evolution of the system is that of a spatial birth-death process over the interference graph. Let $\pi(A)$ represent the equilibrium probability of the set $A \in V$ being active while $\overline{A} \in V$ is idle. It is clear that A can be represented by the p-tuple $A = (n_1, n_2, \ldots, n_p)$ where n_i is the number of activities of type i. It is easy to show that the above birth-death process is time-reversible and that the equilibrium probability distribution $\pi(.)$ satisfies the following detailed balance equation:

$$\pi(A)\left(\lambda_i|\overline{A^c}| + \mu_i|A|\right) = \sum_{x \in \overline{A^c}} \mu_i \pi(A \cup \{x\}) + \sum_{x \in A} \lambda_i \pi(A \backslash \{x\}) \qquad (3.1.1)$$

where x corresponds to an activity of type i. Here A^c denotes the closure of A, that is, the set of vertices in A and those neighboring vertices in A.

To solve equation (3.1.1), define a set of nodes in the interference graph to be *independent* if no two nodes in such a set are neighbors. There is a one-to-one

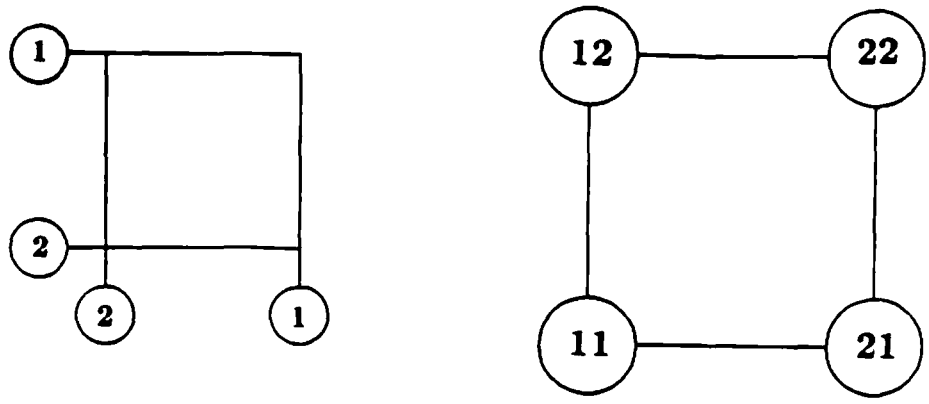

Figure 3.1.1: Interference Graph of 2 × 2 Crossbar

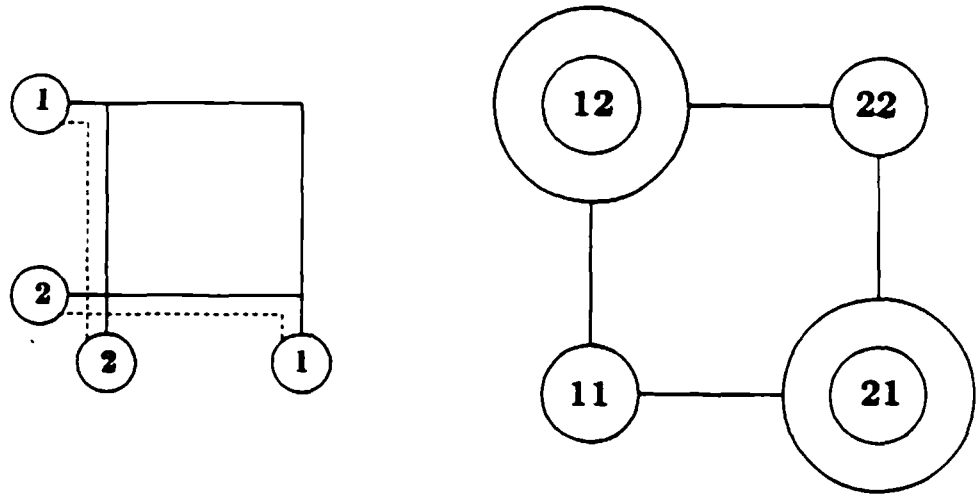

Figure 3.1.2.a: Possible Transmission Configuration (Independent Set)

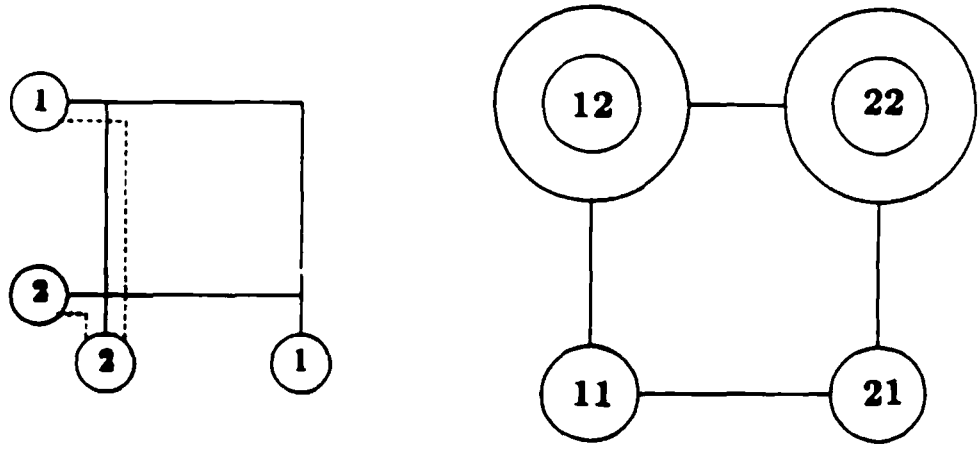

Figure 3.1.2.b: Impossible Transmission Configuration (Not-Independent Set)

correspondence between possible concurrent activities and "independent " nodes. This is illustrated for a 2×2 crossbar in Figure 3.1.2.a. and in Figure 3.1.2.b

Let J denote the set of independent subsets of G. Define $\alpha_N^{n_1,\ldots,n_p}$ to be the number of distinct independent subsets of G having n_i nodes of type i. It is easy to verify the following:

Theorem 3.1. The equilibrium probability that solves equation (3.1.1) is given by

$$\pi(A) = \begin{cases} \rho_1^{n_1} \cdots \rho_p^{n_p} / Z_N & A \in J \\ 0 & \text{otherwise} \end{cases}$$

where $\rho_i = \lambda_i/\mu_i$ and Z_N, the "partition" function of the system is given by:

$$Z_N = \sum_{(n_1,\ldots,n_p)\in J} \rho_1^{n_1} \cdots \rho_p^{n_p} = \sum \alpha_N^{n_1,\ldots,n_p} \rho_1^{n_1} \cdots \rho_p^{n_p} \qquad \blacksquare$$

The partition function is thus the generating function, with one term for each possible concurrency level, and each term given a weight, related to the combinatorics of concurrency of the corresponding configurations. In other words, the partition function specifies how the possible system configurations are partitioned among different concurrency levels. This is analogous to the concept of a partition function in statistical mechanics (see Appendix 1 at the end of this thesis)

A number of important performance averages can be computed from Z_N. The average concurrency of class σ can be found in terms of the "logarithmic derivative" of Z_N:

$$E_\sigma = \sum_k k\pi(n_\sigma = k) = \rho_\sigma \frac{\partial \log Z_N}{\partial \rho_\sigma} \tag{3.1.2}$$

The total number of concurrent activities is obviously

$$E_{total} = E_1 + E_2 + \cdots + E_p \tag{3.1.3}$$

The throughput T_σ is defined as the average number of class σ activities processed per unit of time.

$$T_\sigma = \mu_\sigma E_\sigma = \lambda_\sigma \frac{\partial \log Z_N}{\partial \rho_\sigma} \tag{3.1.4}$$

If there are N_σ nodes in the interference graph corresponding to activities of type σ, then knowing the throughput T_σ, we can calculate the non-blocking probability B_σ that an arriving activity of class σ is not blocked.

$$B_\sigma = \frac{T_\sigma}{\lambda_\sigma N_\sigma} \tag{3.1.5}$$

The utilization of the system (the probability of at least one activity) is:

$$U = 1 - \frac{1}{Z_N} \tag{3.1.8}$$

Therefore, a number of global performance averages can be computed once we know Z_N. For most models, it is very difficult to derive Z_N in closed form. The key difficulty in the performance analysis of such models can be viewed as trying to derive a good approximation to Z_N or to calculate it numerically.

We should note some critical points about the interference model considered in this work. Some assumptions could be relaxed without incurring any significant change. The assumption that each agent is either idle or busy does not properly account for blocked activities. The easy solution is to assume that blocked agents are regenerated from the same statistics as new arrivals. This is usually a good approximation used routinely in the analysis of many systems ([Coop81, Mitr84, Rior62]).

Finally, in this work we will also analyze some product-form solution models (e.g. queueing networks) which are not described by the above interference model.

3.2. THE METHOD OF CANONICAL APPROXIMATION

In this section we introduce the method of canonical approximation to approximate Z_N in closed form. It computes the closed form approximation to Z_N from its generating function $Z_G(t)$. The method is similar to that developed in statistical mechanics to establish the equivalence of the canonical and grand canonical ensemble (see Appendix 2 at the end of this thesis). This ensemble equivalence allows us to use the method to compute average performance measures using the obtained approximation of the partition function. By analogy with physics (see Appendix 2 at the end of this thesis), we will call $Z_G(t)$ the *Grand Partition Function.* It is assumed that $Z_G(t)$ can be computed in closed form. This is true for many models, including a variety of product form solution models which are not described by an interference system (e.g. closed queueing networks described in Chapter 4). The canonical approximation method is based on the following theorem:

Theorem 3.2.1. (Main theorem) Let $Z_G(t)$ be a function analytic around the origin and having the power series

$$Z_G(t) = \sum_{N=0}^{\infty} Z_N t^N \tag{3.2.1}$$

with $Z_N > 0$. Let the function $f_N(t)$ be defined in the domain of convergence of $Z_G(t)$ by

$$f_N(t) = \log Z_G(t) - (N+1)\log t \tag{3.2.2}$$

Then

(1) There exists R, $0 < R \leq \infty$ such that the function $Z_G(t)$ is analytic in the domain $D = \{t: \quad 0 \leq |t| < R\}$.

(2) There exists a unique positive real solution $t_N \in D$ of the equation $f'_N(t_N) = 0$.

(3) The partition function Z_N can be represented

$$Z_N = \frac{Z_G(t_N)}{t_N^{N+1}\sqrt{2\pi f''_N(t_N)}}(1+\epsilon_N) \tag{3.2.3}$$

where

$$\epsilon_N = \frac{1}{24}(3\nu_4 - 5\nu_3^2) + \frac{1}{1152}(168\nu_3\nu_5 + 385\nu_3^4 - 630\nu_3^2\nu_4 - 24\nu_6 + 105\nu_4^2) + \cdots$$

and ν_s is defined by

$$\nu_s = f_N^{(s)}(t_N) \Big/ \left(\sqrt{f_N^{(2)}(t_N)}\right)^s \tag{3.2.4}$$

Proof. By classical Abel's theorem of complex analysis (Theorem 2.4. in [Alfh66]), for every power series with positive coefficients which is analytic around the origin†, there exists a number R, $0 < R \leq \infty$, called the radius of the convergence such that the series converges to an analytic function in the domain $D = \{t: \quad 0 \leq |t| < R\}$.

By Cauchy's theorem ([Alfh66])

$$Z_N = \frac{1}{2\pi i}\oint \frac{Z_G(t)}{t^{N+1}}\,dt \tag{3.2.5}$$

This can be rewritten in terms of $f_N(t)$ (equation 3.2.2) as follows

$$Z_N = \frac{1}{2\pi i}\oint \exp(f_N(t))\,dt \tag{3.2.6}$$

---

† It is not true, in general,that $Z_N > 0$ implies that $Z_G(t)$ is analytic with a positive radius of convergence. For example, $Z_N = N!$ is an obvious counterexample. Thus, the assumption of analyticity around the origin is very important.

Let us show that there is a unique positive $t_N \in D$ at which $f'_N(t_N) = 0$. To do so, consider the function

$$h_N(t) = tZ_G(t)f'_N(t) = tZ'_G(t) - (N+1)Z_G(t) = \sum_{n=0}^{\infty}[n-(N+1)]Z_n t^n$$

In this discussion, we will consider $h_N(t)$ as a function of a real variable. Clearly, $t_N > 0$ is a root of $f'_N(t) = 0$ if and only if $h_N(t_N) = 0$. There are two cases to consider

Case 1: $R < \infty$. We can rewrite $h_N(t)$ as follows

$$h_N(t) = \sum_{n=0}^{N}[n-(N+1)]Z_n t^n + \sum_{n=N+1}^{\infty}[n-(N+1)]Z_n t^n$$

Let us examine the function $h_N(t)$ as $t \mapsto R$. The first part of $h_N(t)$ decreases but is bounded below by some finite M

$$-\infty < M = \sum_{n=0}^{N}[n-(N+1)]Z_n R^n < 0$$

On the other hand, the second part of $h_N(t)$ tends to $+\infty$ as $t \mapsto R$. Therefore, $h_N(t)$ tends to $+\infty$ as $t \mapsto R$. Since $h_N(0) = -(N+1)Z_0 < 0$ and $h_N(t)$ is continuous, it follows that there is a $t_N \in (0, R)$ such that $h_N(t_N) = 0$.

Case 2: $R = \infty$. Set $s = n - (N+1)$. Then $h_N(t)$ can be rewritten as follows

$$h_N(t) = \sum_{n=0}^{N}[n-(N+1)]Z_n t^n + \sum_{s=0}^{\infty} sZ_{s+N+1}t^{N+1+s}$$

$$= \sum_{n=0}^{N}\left[(n-N-1)Z_n + nZ_{N+n+1}t^{N+1}\right]t^n + \sum_{s=N+1}^{\infty} sZ_{s+N+1}t^{N+1+s}$$

Let t^* be defined by

$$t^* = \max\left(\frac{NZ_n}{nZ_{n+N+1}}\right)^{\frac{1}{N+1}} \qquad n = 1,\ldots,N$$

Since $Z_n > 0$ for all n, it follows that t^* is finite. For $t > t^*$, all of the coefficients of $h_N(t)$ become positive. Therefore, $h_N(t) > 0$ for $t > t^*$. Again, since $h(0) = -(N+1)Z_0 < 0$ and $h_N(t)$ is continuous, it follows that there is $t_N \in (0, t^*)$ such that $h_N(t_N) = 0$.

To prove the uniqueness, let t_N be any of the positive roots of $h_N(t) = 0$ and let us examine $h'_N(t_N)$.

$$h'_N(t_N) = \frac{1}{t_N}\sum_{n=0}^{N} n[n-(N+1)]Z_n t_N^n + \frac{1}{t_N}\sum_{n=N+1}^{\infty} n[n-(N+1)]Z_n t_N^n$$

$$> \frac{N+1}{t_N}\sum_{n=0}^{N}[n-(N+1)]Z_n t_N^n + \frac{N+1}{t_N}\sum_{n=N+1}^{\infty}[n-(N+1)]Z_n t_N^n$$

$$= \frac{N+1}{t_N}h_N(t_N) = 0$$

Hence, $h'_N(t_N) > 0$. This means that each root t_N is a simple root. Moreover, it means that the function $h_N(t)$ is always increasing in the neighborhood

of a root. But then, if $t_N^{(1)}$ and $t_N^{(2)}$ are two adjecent real roots (i.e. no real root in $(t_N^{(1)}, t_N^{(2)})$) within the radius of convergence of $h_N(t)$, then for sufficiently small $\epsilon > 0$, we have $h_N(t_N^{(1)} + \epsilon) > 0$ and $h_N(t_N^{(2)} - \epsilon) < 0$. Since $h_N(t)$ is analytic within the radius of convergence, there exists a root of $h_N(t) = 0$ in $(t_N^{(1)} + \epsilon, t_N^{(2)} - \epsilon)$. This contradiction proves the uniqueness of t_N. Therefore, there is a unique positive $t_N \in D$ such that $f_N'(t_N) = 0$. It is easy to show that at $t = t_N$, the second derivative of $\exp(f_N(t))$ is positive. Therefore, the integrand $\exp(f_N(t))$ (equation 3.2.6) exhibits a minimum at t_N, when treated as a function of a real variable.

Since $f(t)$ is analytic in D, it must possess a *unique* derivative everywhere in D. This means that $f'(t_N) = 0$ irrespective of the direction in which we pass through the point t_N. From the Cauchy-Riemann conditions ([Alfh66]),

$$f_N'(t) = \frac{\partial f_N}{\partial x} - i\frac{\partial f_N}{\partial y}$$

the second derivative of the function with respect to y must be equal and opposite to the second derivative with respect to x. It follows then that as we proceed through the point t_N in a direction orthogonal to the real axis, the integrand (equation 3.2.5) exhibits a maximum. Thus, t_N is a saddle point. This is illustrated in Figure 3.2.1.

Intuitively, for large N, the saddle-point is very "steep". One can expect that the most dominant contribution to the integral comes only from the immediate neighborhood of the point t_N. This is, in fact, the basic idea of the approximation.

Formally, let us choose as a path of integration the straight line passing through t_N and parallel to the imaginary axis. It can be shown ([Wint47]) that on any straight line parallel to the imaginary axis, the integrand attains its maximum modulus only where the line crosses the real axis. We can thus try to approximate Z_N by the expansion of $f_N(t)$ in the neighborhood of t_N.

Formally, on this contour we have

$$f_N(t) = f_N(t_N) - \frac{1}{2!}f_N^{(2)}(t_N)t^2 - \frac{1}{3!}f_N^{(3)}(t_N)it^3 + \frac{1}{4!}f_N^{(3)}(t_N)t^4 + \cdots \qquad (3.2.7)$$

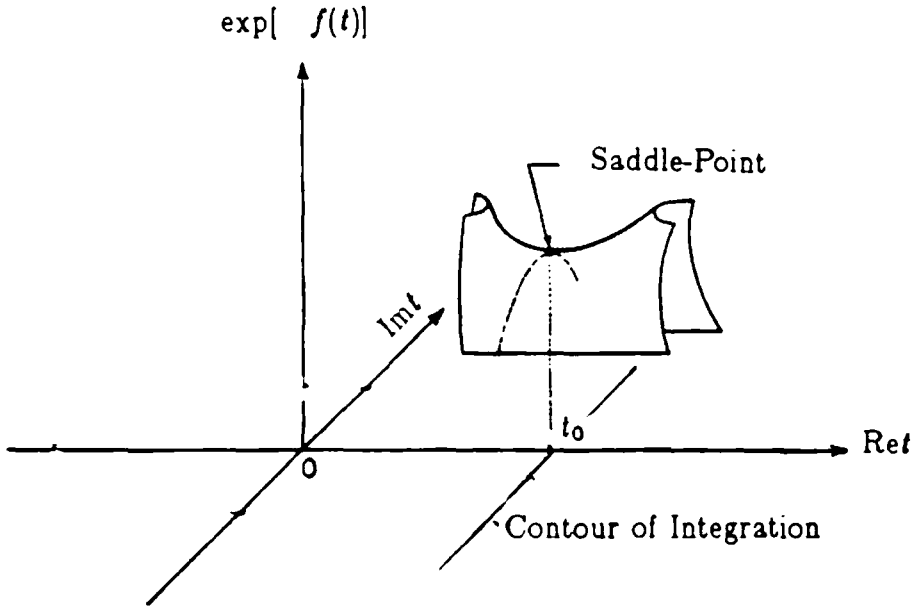

Figure 3.2.1: The Saddle-Point

The equation 3.2.5 can now be rewritten as follows

$$Z_N = \frac{\exp(f_N(t_N))}{2\pi} \int_{-\infty}^{+\infty} \exp\left(-\frac{f_N^{(2)}(t_N)t^2}{2}\right) \exp\left(-\frac{f_N^{(3)}}{3!}it^3 + \frac{f_N^{(3)}(t_N)}{4!}t^4 + \cdots\right) dt \tag{3.2.8}$$

Set $w = t\sqrt{f_N^{(2)}(t_N)}$. Using the expansion $\exp(x) = 1 + x + x^2/2! + \cdots$, and equation 3.2.4, we can rewrite 3.2.8 as follows

$$Z_N = \frac{\exp(f_N(t_N))}{2\pi} \int_{-\infty}^{+\infty} [\exp(-w^2/2)] \left\{1 - \frac{\nu_3}{3!}iw^3 + \frac{\nu_4}{4!}w^4 - \frac{1}{2}\left(\frac{\nu_3}{3!}w^3\right)^2 + \cdots\right\} dw$$

where ν_s are defined by 3.2.4. The odd powers of w vanish on integration. For the remaining terms we use the identity ([Abra82])

$$\int_{-\infty}^{+\infty} e^{-\frac{1}{2}x^2}x^{2n} = 1 \cdot 3 \cdot 5 \cdots (2n-1)\sqrt{2\pi}$$

to obtain the following expansion of Z_N:

$$Z_N = \frac{Z_G(t_N)}{t_N^{N+1}\sqrt{2\pi f_N''(t_N)}}[1 + \frac{1}{24}(3\nu_4 - 5\nu_3^2) +$$

$$\frac{1}{1152}(168\nu_3\nu_5 + 385\nu_3^4 - 630\nu_3^2\nu_4 - 24\nu_6 + 105\nu_4^2) + \cdots]$$

Let us rewrite this as follows

$$Z_N = \frac{Z_G(t_N)}{t_N^{N+1}\sqrt{2\pi f_N''(t_N)}}[1 + \epsilon_N] \tag{3.2.9}$$

where

$$\epsilon_N = \frac{1}{24}(3\nu_4 - 5\nu_3^2) + \frac{1}{1152}(168\nu_3\nu_5 + 385\nu_3^4 - 630\nu_3^2\nu_4 - 24\nu_6 + 105\nu_4^2) + \cdots \quad (3.2.10)$$

It is not apparent from the above formal development that this is an asymptotic expansion of Z_N, in which the first (most dominant) and second term of the error are given by the corresponding terms in equation 3.2.10, and in which the remainder in this equation is of the same order as the last term neglected. This can be established by the method of steepest descent - the path of integration being the curve of the steepest descent through t_N upon which the modulus of the integrand decreases most rapidly. Details can be found in [Dani54]. Thus, to establish the relative error of the approximation, we need to compute the first error term(s) as given by equation 3.2.10. The estimates on these terms give a good bound on the relative error. For most of the functions $Z_G(t)$ considered in our examples, the relative error is typically of order $O(1/N)$. Hence, for large N, we have the following approximation for Z_N:

$$Z_N \approx \frac{Z_G(t_N)}{t_N^{N+1}\sqrt{2\pi f_N''(t_N)}}$$

∎

The above theorem shows how to compute the partition function Z_N from its generating function $Z_G(t)$. If we consider $Z_G(t)$ to be the exponential generating function, the representation for Z_N (equation 3.2.3) will have a factor of $N!$ in front. To apply the method, we need to compute $Z_G(t)$. For a wide variety of problems, it is easier to compute $Z_G(t)$ in closed form than Z_N.

It is often the case that $R < \infty$, that is, the grand partition function $Z_G(t)$ is not an entire function - it is analytic everywhere except at isolated points (singularities). It is interesting to note the following:

Theorem 3.2.2. Let t_N be the saddle point used in evaluating Z_N and let R be the radius of convergence of the grand partition function. If $R < \infty$ then $t_N \mapsto R$.

Proof. First we show that $\{t_N\}$ is a monotonically increasing sequence. Recall that

$$f'_N(t) = \frac{Z'_G(t)}{Z_G(t)} - \frac{N+1}{t}$$

Hence, t_N is the solution of

$$t_N Z'_G(t_N) - (N+1)Z_G(t_N) = 0 \tag{3.2.11}$$

Define the following functions $w_N(t)$ by

$$w_N(t) = tZ'_G(t) - (N+1)Z_G(t)$$

Clearly, t_N is a root of $w_N(t) = 0$. By the previous theorem, there is a unique positive $t_{N+1} < R$ at which the function $f_{N+1}(t) = \log Z_G/t^{N+2}$ attains a minimum (when treated as a function of a real variable), that is, $w_{N+1}(t_{N+1}) = 0$. Since $f''_{N+1}(t_{N+1}) > 0$ (see theorem 3.2.1), it follows that $f_{N+1}(t)$ is a convex function. It follows then that for $t > t_{N+1}$ we have $w_{N+1}(t) > 0$. Since $Z_G(t) > 0$ for $t > 0$, it follows then that

$$w_N(t) = w_{N+1}(t) + Z_G(t) > w_{N+1}(t)$$

Therefore, for $t > t_{N+1}$ we have $w_N(t) > 0$. This implies that if $w_N(t_N) = 0$ then $t_N < t_{N+1}$. Hence, $\{t_N\}$ is monotonically increasing. Since $t_N < R$ this sequence has a limit $t^* \leq R$. Clearly, $t_N < t^*$ for all N. We claim that $t^* = R$

Assume otherwise, that is $t^* < R$. We can derive a contradiction by showing that there exists N_0 such that for $N > N_0$ we have $t_N > t^*$. First, since $t^* < R$ it follows that $Z_G(t^*) < \infty$. In particular, $t^* Z'_G(t^*) < \infty$. Now, the sequence $\{t_N\}$ is monotonically increasing with N. Moreover, since $Z_N > 0$ for all N, the grand partition function $Z_G(t)$ is monotonically increasing with $t > 0$. Clearly then, $(N+1)Z_G(t_N) \mapsto \infty$ with N. In particular, there is N_0 such that for $N > N_0$ we have

$$(N+1)Z_G(t_N) > t^* Z'_G(t^*) \tag{3.2.12}$$

From equations 3.2.11 and 3.2.12, it follows that for such N

$$t_N Z'_G(t_N) - t^* Z'_G(t^*) > 0 \tag{3.2.13}$$

Again, since $tZ'_G(t)$ is monotonically increasing as a function of t, it follows from equation 3.2.13 that $t_N > t^*$. The above contradiction shows that $t^* = R$ and hence, $t_N \mapsto R$. ∎

3.3. THE CASE OF POLES

The main theorem of canonical approximation is applicable for any $Z_G(t)$ satisfying the conditions of theorem 3.2.1. However, when $Z_G(t)$ is meromorphic (analytic everywhere except at isolated poles), we can approximate Z_N in a different way using the calculus of residues. The main result of this section is the following theorem†.

Theorem 3.3.1. Let the grand partition function $Z_G(t)$ have poles at $t_0, t_1, \ldots$ with $0 < |t_0| < |t_1| < \cdots$. Let m_i be the degree of the pole at t_i and let $H_i(N, m_i, t_i)$ be defined by

$$H_i(N, m_i, t_i) = \sum_{s=0}^{m_i - 1} (-1)^s \binom{N+s}{s} \frac{Res[Z_G(t)(t - t_i)^s]_{t=t_i}}{t_i^{s-1}}$$

Then for large N we have

$$Z_N \approx -\frac{H_0(N, m_0, t_0)}{t_0^{N+1}}$$

† This is a generalization of the theorem given in [Henr77] for the case where all poles are simple.

The relative error of the approximation

$$\epsilon_N \leq \sum_{i=1} \frac{H_i(N, m_i, t_i)}{H_0(N, m_0, t_0)} \left(\frac{t_0}{t_i}\right)^{N+1} \mapsto 0$$

exponentially with N. In particular, if t_0 is a simple pole (order 1) then

$$Z_N \approx -\frac{Res[Z_G(t_0)]}{t_0^{N+1}}$$

The proof of theorem 3.3.1 can be found in the Appendix at the end of this chapter (section 3.6). It follows from the above theorem that the partition function is asymptotically determined by the smallest pole of its grand partition function. The precision of the approximation depends on $|t_0/t_i|$. If the second (by magnitude) pole is very close to t_0, then one can use the approximation by evaluating the residue at the first two poles and so on. In cases when the other poles are not close (in magnitude) to t_0, taking the residue at the smallest pole provides a very good approximation. More examples will be considered in Chapter 7.

If the smallest pole is simple, then the evaluation of the partition function (and its derivatives, to obtain the performance measures) is straightforward. If the smallest pole is not simple, then the computation of Z_N and of its derivatives will involve the terms of the order of $N!$. In such cases, we should consider the approximation using the saddle-point (Theorem 3.2).

A Very Simple Example. Let us consider a simple example of a $M/M/1/N$ queue. Assuming that the grand partition function

$$Z_G(t) = \sum_{N=0}^{\infty} Z_N t^N = \sum_{N=0}^{\infty} \frac{1-\rho^{N+1}}{1-\rho} t^N = \frac{1}{(1-t)(1-\rho t)}$$

is given, we would like to obtain an estimate of Z_N. Assume $\rho < 1$. The smallest pole $t_1 = 1$ is of order 1, the corresponding residue is $Res[Z_G(t_1)] = \frac{-1}{1-\rho}$.

Canonical approximation gives the following

$$Z_N \approx \frac{-Res[Z_G(t_1)]}{t_1^{N+1}} = \frac{1}{1-\rho}$$

The relative error is

$$\epsilon_N = \rho^N \mapsto 0$$

For larger N our approximation is more exact. In fact, the approximate partition function corresponds exactly to the partition function of the $M/M/1$ queue (any number of customers can be queued), which is the limiting case of $M/M/1/N$ system for increasing N.

3.4. EXAMPLE: MACHINE INTERFERENCE MODEL

Let us now consider an example of a machine repairman model. It is variously called the machine repair model, the machine interference model, or the cyclic queue model ([Alle78, Ferd71]). This model consists of N machines and a single repairman. A machine breaks down according to the exponential interarrival law with rate λ. If a repairman is available, the broken machine requests the service for a period distributed exponentially with rate μ. It is then back in operation. If the repairman is busy, the broken machine has to wait, thus forming a queue in front of the repairman.

The above system is thus an $M/M/1/N/N$ queueing model. It can be used to represent an interactive computer system ([Munt75]). For such systems, the repairman consists of one or more CPUs plus the associated queues (Figure 3.4.1). The customers (users) are interacting with the system through N terminals. Each customer is assumed to be in exactly one of three states at any time:

(1) "thinking" at the terminal. The think time is exponentially distributed with average value $1/\lambda$.

(2) queueing for CPU service (First-Come First-Serve Discipline).

(3) receiving the CPU service which is distributed exponentially with the average of $1/\mu$ time units.

Thus, a user at a terminal cannot submit a new request for CPU service until the previous request has been satisfied. To calculate the partition function Z_N, we need to to calculate α_N^i - the number of configurations corresponding to i broken machines in the queue. The number of ways to have i broken machines is, obviously, $\binom{N}{i}$. Since the order in which these i machines are arranged in the queue is important, we get $\alpha_N^i = \binom{N}{i} i!$. Therefore, the partition function for the model is

$$Z_N = \sum_{i=0}^{N} \binom{N}{i} i! \rho^i \tag{3.4.2}$$

We can rewrite this as follows:

$$Z_N = \sum_{i=0}^{N} \binom{N}{i} i! \rho^i = \sum_{i=0}^{N} \frac{N!}{(N-i)!} \rho^i = N! \rho^N \sum_{i=0}^{N} \frac{1}{i! \rho^i} \tag{3.4.2}$$

From equation 3.4.2 we obtains the following recursive expression for the partition function:

$$Z_{N+1} = (N+1)! \rho^{N+1} \sum_{i=0}^{N+1} \frac{1}{i! \rho^i} = (N+1) \rho Z_N + 1$$

Rewriting that as

$$\sum_{N=0}^{\infty} \frac{Z_{N+1}}{(N+1)!} t^{N+1} = \rho t \sum_{N=0}^{\infty} \frac{Z_N}{N!} t^N + \sum_{N=0}^{\infty} \frac{t^{N+1}}{(N+1)!}$$

we obtain the following expression for the (exponential) grand partition function

$$Z_G(t) = \sum_{N=0}^{\infty} \frac{Z_N}{N!} t^N = \frac{\exp(t)}{1-\rho t} \tag{3.4.3}$$

The grand partition function has only one (simple) pole at $t_0 = 1/\rho$. The residue at that pole is

$$Res[Z_G(1/\rho)] = -\frac{\exp(1/\rho)}{\rho}$$

Canonical approximation gives the following estimate to the partition function

$$Z_N \approx -N! \frac{Res[Z_G(1/\rho)]}{(1/\rho)^{N+1}} = N! \rho^N \exp(1/\rho) \tag{3.4.4}$$

The mean queue length of customers (including the one at the machine)

$$Q_N = \frac{\rho}{Z_N} \frac{\partial Z_N}{\partial \rho} \approx N - \frac{1}{\rho} \tag{3.4.5}$$

The utilization

$$U_N = 1 - \frac{1}{Z_N} \approx 1 - \frac{1}{N! \rho^N \exp(1/\rho)} \tag{3.4.6}$$

The throughput is μU_N. Therefore, by Little's formula, the response time (mean time required to put the machine back into operation)

$$W_N = \frac{Q_N}{\mu U_N} \approx \frac{N - \frac{1}{\rho}}{\mu \left(1 - \frac{1}{N! \rho^N \exp(1/\rho)}\right)}$$

For large N, we can use the Stirling approximation for $N!$ to show

$$\frac{1}{N! \rho^N \exp(1/\rho)} \approx \frac{1}{\sqrt{2\pi N} \left(\frac{N\rho}{e}\right)^N \exp(1/\rho)} \mapsto 0$$

And therefore,

$$W_N \mapsto \frac{N}{\mu} - \frac{1}{\lambda} \tag{3.4.7}$$

It is interesting to compare our approach here to that taken by earlier researchers. In the "traditional" analysis of the machine repairman model ([Alle78, Klei75]) one shows that mean response time is

$$W_N = \frac{N}{\mu(1-p_0)} - \frac{1}{\lambda}$$

where p_0 is the probability that the server is idle. For large N, one assumes that it is very unlikely that the server is idle and thus sets $p_0 = 0$. From this assumption, one obtains the expression for the response time identical to that in equation 3.4.7 and similarly for other performance measures.

The notable exception to these "traditional" analyses is the work by Ferdinand ([Ferd71]) who used a statistical mechanics approach to analyze this system. The partition function Z_N is evaluated by its asymptotic expansion in terms of the incomplete gamma function. Although this work used the analogies from statistical mechanics, the computational method to evaluate Z_N cannot be applied to other systems.

Let us examine the behavior of the response time of the system as one changes N but keeps λ and μ fixed. For $N = 1$ there is no queueing and therefore $W_N = 1/\mu$. For small values of N (corresponding to $\rho \ll 1/N$), the users interfere with each other very little: when one user wants a CPU interaction, the others are usually in the think mode and so very little queueing occurs. Thus, the curve of the mean response time is asymptotic at $N = 1$ to the curve $W_N = 1/\mu$. On the other hand, if $N \mapsto \infty$, then the response curve is asymptotic to $W_N = (N/\mu) - (1/\lambda)$. This is illustrated in Figure 3.4.2.

The two asymptotes intersect at the point $N^* = 1 + (1/\rho)$. This can be thought of as the system *saturation* point ([Klei76]). If each interaction required exactly $1/\mu$ units of CPU service time and exactly $1/\lambda$ units of think time, then N^* is the maximum number of terminals that could be scheduled in a way that causes no mutual interference. For $N \ll N^*$, there is almost no interference and the response time W_N is approximately $1/\mu$. On the other hand, for $N \gg N^*$, users

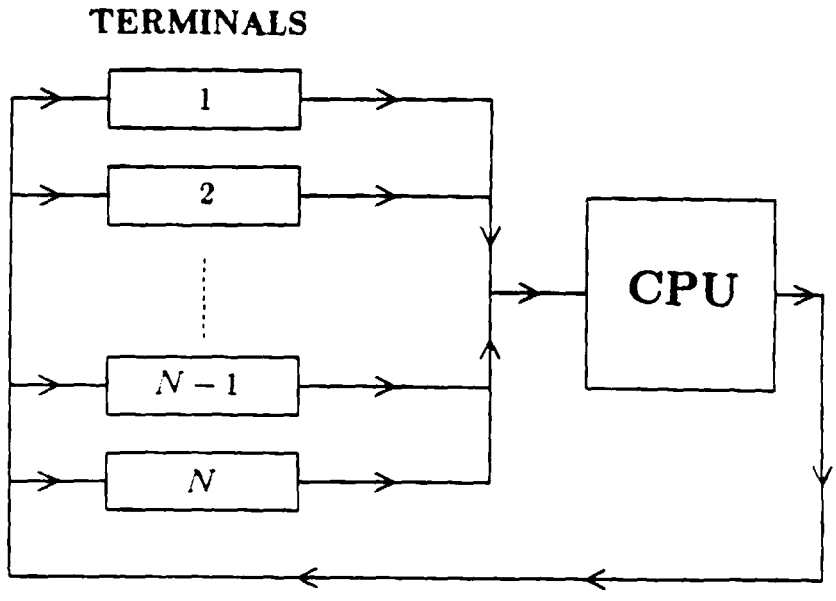

Figure 3.4.1: Interactive Computer System

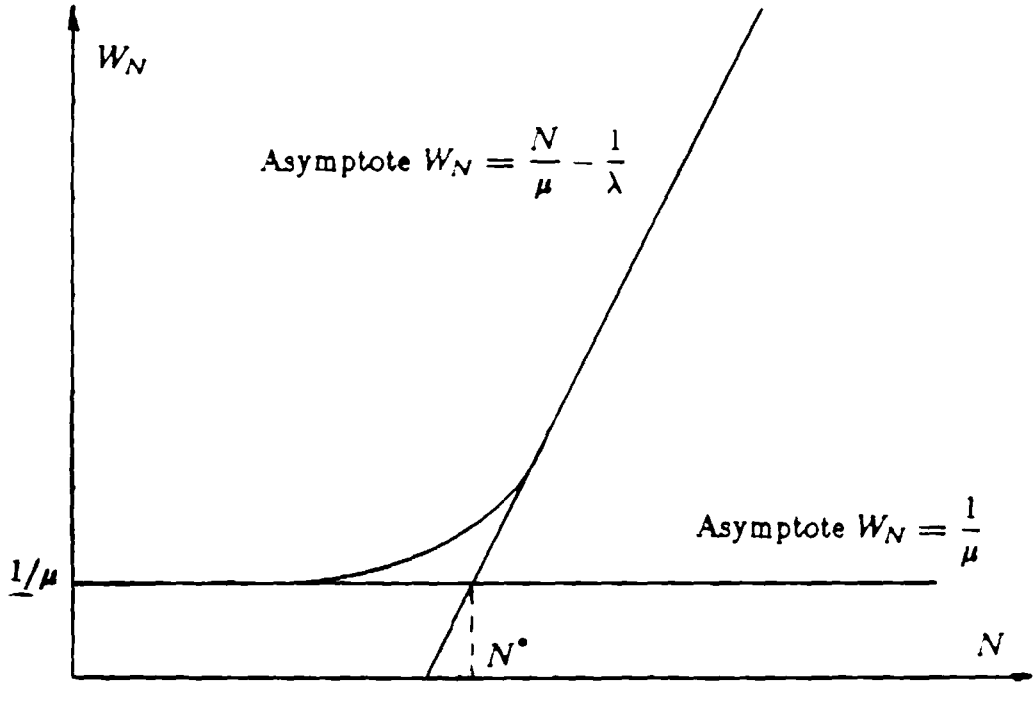

Figure 3.4.2: Response Time vs Number of Users

interfere totally with each other. This means that adding one more terminal raises everyone's response time by $1/\mu$.

3.5. CONCLUSION

This chapter presented a model of interference for a variety of distributed systems. It presented rigorous mathematical foundations for the canonical approximation method and for the analysis of its complexity and precision. We presented and analyzed a classical machine interference model using the new method.

3.6. APPENDIX (PROOF OF THEOREM 3.3.1)

In this appendix we present the proof of theorem 1. To that end, we need the following two lemmas:

Lemma 1. Suppose $f_N(t)$ has a pole of order m at $t = t_0$. Then the residue of $f_N(t)$ at point t_0 is given by (see [Alfh66])

$$Res[f_N(t_0)] = \frac{1}{(m-1)!} g^{(m-1)}(t_0)$$

where $g(t) = (t - t_0)^m f_N(t)$. ∎

Lemma 2.

$$Res\left[\frac{Z_G(t)}{t^{N+1}}\right]_{t=t_i} = \sum_{s=0}^{m_i-1} (-1)^s \binom{N+s}{s} \frac{Res[Z_G(t)(t-t_i)^s]_{t=t_i}}{t_i^{N+s}} \qquad (1)$$

Proof. Since the pole at $t = t_i$ is of order m_i we can write $Z_G(t) = F_N(t)/(t-a)^{m_i}$ where $F_N(t)$ is analytic at $t = t_i$. By the previous lemma

$$Res\left[\frac{Z_G(t)}{t^{N+1}}\right]_{t=t_i} = \frac{1}{(m_i-1)!}\frac{\partial^{m_i-1}}{\partial t^{m_i-1}}\left[\frac{F_N(t)}{t^{N+1}}\right]_{t=t_i} \tag{2}$$

By Leibnitz's rule

$$\frac{\partial^{m_i-1}}{\partial t_i^{m-1}}\left[\frac{F_N(t)}{t^{N+1}}\right] = \sum_{k=0}^{m_i-1}\binom{m_i-1}{k}\frac{\partial^k F_N(t)}{\partial t^k}\frac{\partial^{m_i-1-k}[t^{-(N+1)}]}{\partial t^{m_i-1-k}} \tag{3}$$

Clearly,

$$\begin{aligned}\frac{\partial^{m_i-1-k}t^{-(N+1)}}{\partial t^{m_i-1-k}} &= \frac{(N+1)(N+2)\cdots(N+m_i-1-k)}{t^{N+m_i-k}} \\ &= (-1)^{m_i-1-k}\frac{(m_i-1-k)!}{t_i^{N+m_i-k}}\binom{N+m_i-1-k}{m_i-1-k}\end{aligned} \tag{3}$$

On the other hand,

$$\frac{1}{k!}\left(\frac{\partial^k F_N(t)}{\partial t^k}\right)_{t=t_i} = Res\left[\frac{F_N(t)}{(t-t_i)^{k+1}}\right]_{t=t_i} = Res[Z_G(t)(t-t_i)^{m_i-k}]_{t=t_i} \tag{5}$$

Equations $2-5$ imply 1. ∎

If we define

$$H_i(N, m_i, t_i) = \sum_{s=0}^{m_i-1}(-1)^s\binom{N+s}{s}\frac{Res[Z_G(t)(t-t_i)^s]_{t=t_i}}{t_i^{s-1}}$$

then we can write

$$Res\left[\frac{Z_G(t)}{t_{N+1}}\right]_{t=t_i} = \frac{H_i(N, m_i, t_i)}{t_i^{N+1}}$$

Note that $Res[Z_G(t)(t-t_i)^s]_{t=t_i}$ is independent of N. It is obvious that $H_i(N, m_i, t_i)$ is polynomial in N of degree at most $m_i - 1$.

Now, we can prove the theorem 3.3.1. The statement of the theorem is similar to the method of subtracted singularities ([Henr77])†. Let us recall the statement of the theorem:

Theorem 3.3.1. Let the grand partition function $Z_G(t)$ have poles at $t_0, t_1, \ldots$ with $0 < |t_0| < |t_1| < \cdots$. Let m_i be the degree of the pole at t_i‡. Let $H_i(N, m_i, t_i)$ be defined by

$$H_i(N, m_i, t_i) = \sum_{s=0}^{m_i-1} (-1)^s \binom{N+s}{s} \frac{Res[Z_G(t)(t-t_i)^s]_{t=t_i}}{t_i^{s-1}}$$

Then for large N we have

$$Z_N \approx -\frac{H_0(N, m_0, t_0)}{t_0^{N+1}}$$

The relative error of the approximation

$$\epsilon_N \leq \sum_{i=1} \frac{H_i(N, m_i, t_i)}{H_0(N, m_0, t_0)} \left(\frac{t_0}{t_i}\right)^{N+1} \mapsto 0$$

---

† The proof in [Henr77] is given for the case when all of the poles are simple.

‡ In this theorem, we consider the case where there is a unique pole of the same magnitude. The proof for the more general case, where there are several poles of the same magnitude, is along the same lines.

exponentially with N. In particular, if t_0 is a simple pole (order 1) then

$$Z_N \approx -\frac{Res[Z_G(t_0)]}{t_0^{N+1}}$$

Proof. Let C be the circle around the origin excluding all the poles and let C' be the-circle around the origin surrounding all the poles. Then by the residue theorem ([Alfh66])

$$Z_N = \frac{1}{2\pi i}\oint_C \frac{Z_G(t)}{t^{N+1}}dt = -\sum_{i=0} Res\left[\frac{Z_G(t)}{t^{N+1}}\right]_{t=t_i} + \frac{1}{2\pi i}\oint_{C'} \frac{Z_G(t)}{t^{N+1}}dt =$$

$$= -\sum_{i=0} \frac{H_i(N, m_i, t_i)}{t_i^{N+1}} + \frac{1}{2\pi i}\oint_{C'} \frac{Z_G(t)}{t^{N+1}}dt =$$

$$= -\frac{H_0(N, m_0, t_0)}{t_0^{N+1}}\left[1 + \sum_{i=1} \frac{H_i(N, m_i, t_i)}{H_0(N, m_0, t_0)}\left(\frac{t_0}{t_i}\right)^{N+1}\right] + \frac{1}{2\pi i}\oint_{C'} \frac{Z_G(t)}{t^{N+1}}dt$$

Clearly,

$$\left|\frac{1}{2\pi i}\oint_{C'} \frac{Z_G(t)}{t^{N+1}}dt\right| \leq \frac{\max_{t\in C'}[Z_G(t)]}{[t]^N}$$

The term $H_i(N, m_i, t_i)/H_0(N, m_0, t_0)$ is a polynomial in N of degree at most m_i and thus increases polynomially with N. On the other hand, since $|t_0| < |t_i|$, the term $(t_0/t_i)^{N+1}$ is exponentially decreasing with N. Therefore,

$$\frac{H_i(N, m_i, t_i)}{H_0(N, m_0, t_0)}\left(\frac{t_0}{t_i}\right)^{N+1} \rightarrowtail 0$$

exponentially fast. For large N we can therefore write

$$Z_N = \frac{-H_0(N, m_0, t_0)}{t_0^{N+1}}[1 + \epsilon_N]$$

with the (relative) error of the approximation

$$\epsilon_N \leq \left[\sum_{i=1} \frac{H_i(N, m_i, t_i)}{H_0(N, m_0, t_0)} \left(\frac{t_0}{t_i}\right)^{N+1}\right] + \frac{\max_{t \in C'}[Z_G(t)]}{H_0(N, m_0, t_0)} \left(\frac{t_0}{|t|}\right)^{N+1}$$

$$\leq \sum_{i=1} \frac{H_i(N, m_i, t_i)}{H_0(N, m_0, t_0)} \left(\frac{t_0}{t_i}\right)^{N+1} \rightarrow 0$$

exponentially fast. Therefore, for large N

$$Z_N \approx -\frac{H_0(N, m_0, t_0)}{t_0^{N+1}}$$

If t_0 is a simple pole, then

$$Res\left[\frac{Z_G(t)}{t^{N+1}}\right]_{t=t_0} = \frac{Res[Z_G(t_0)]}{t_0^{N+1}}$$

We have therefore

$$Z_N \approx -\frac{Res[Z_G(t_0)]}{t_0^{N+1}}$$

CHAPTER 4

CANONICAL APPROXIMATION: PACKET RADIO NETWORKS

4.1. INTRODUCTION

In this chapter the method of canonical approximation is applied to the performance analysis of multihop packet radio networks (PRNETs). These networks consist of geographically distributed radio units broadcasting data packets over a limited range. A key design problem of a PRNET is to resolve the interference that occurs whenever two or more nodes try to transmit over a shared channel within the same neighborhood. This is accomplished by means of a multiple access protocol - a set of rules which defines the process by which a node proceeds to transmit. The primary performance measure is the nodal throughput - the average number of packets delivered by a node successfully per unit of time. Other measures of interest include the steady-state probability distribution of the number of transmissions, the average delay, the fraction of the channel capacity used for successful transmission, the probability that a scheduled transmission is successful, and the average number of packets in the system.

Most of the previous work on multiple access protocols has been confined to the single hop case: a transmission of each node may interfere with transmissions of all other nodes (see Chapter 2). Although a general markovian model of multihop packet radio networks is available ([Boor80, Toba82]), the computation of performance measures (primarily, the nodal throughput) has been done only for very small networks due to the complexity of the numerical algorithms involved.

In this chapter, we apply canonical approximation to calculate the nodal throughput for several networks operating under CSMA and C-BTMA. We obtain

simple explicit formulae for the nodal throughput for any network size N, as well as identify (analytically) the loads when a particular protocol gives a better performance. We introduce a method to compute the nodal throughput assuming the zero capture†. The analysis of the multihop networks under zero capture is typically done by simulation ([Toba85]) or by the explicit construction of an underlying Markov chain ([Braz85]), which is computationally intractable unless N is very small.

4.2. THE MODEL

The model of multihop packet radio networks used here is the one introduced by Boorstyn and Kerschenbaum ([Boor80]). In this model, the network consists of N nodes with a specified "hearing matrix". For any two nodes i and j the hearing matrix specifies whether or not i can hear j. The points in time when new and retransmitted packets are scheduled for transmission are called scheduling points. Packets are retransmitted either because at some scheduling point they were inhibited from transmission or because their transmission has been interfered with. The process of scheduling points from a node is assumed to be Poisson with parameter λ. In general, we need not assume the same rate of λ for all the nodes. The lengths of packets are assumed to be distributed exponentially with parameter μ. For notational convenience, assume $\mu = 1$. The model assumes negligible propagation delay.

We will consider two protocols - Carrier Sense Multiple Access (CSMA) with perfect capture, and Conservative Busy Tone Multiple Access (C-BTMA). Capture assumption is defined as the ability of the receiver to correctly receive a packet despite the presence of other time overlapping transmissions. Perfect capture is the ability of receiving correctly the first packet regardless of future overlapping

† This method was developed jointly with Professor Moshe Sidi of the Technion

packets. The zero capture assumption means the complete destruction of the first packet by any overlapping transmission. This is considered in section 4.4.

Under CSMA, a node wishing to transmit senses the channel. If it senses the channel idle, the node starts transmitting; otherwise it waits for the next scheduled point in time and repeats the above procedure. Under the perfect capture assumption, the transmission of a packet from i to j may not be successful only if any of the "hidden nodes" k (neighbors of j but not of i) are transmitting to j at the same time when i starts its transmission.

Under C-BTMA, any node that senses a carrier emits a busy tone. If a node i transmits a packet to node j, all the other neighbors of i transmit a busy tone, thus blocking all nodes in a region within twice the "hearing radius" of node i. Note that under C-BTMA, once a node starts transmitting, it is guaranteed of success.

For these protocols, with the assumptions of Poisson arrivals and exponential service time, it can be shown ([Boor80, Toba83]) that the equilibrium probability distribution $\pi(i)$ of having i simultaneous transmissions in the system is given by

$$\pi(i) = \frac{\rho^i}{Z_N}$$

where $\rho = \lambda/\mu$ and Z_N, the "partition" function of the system, is given by:

$$Z_N = \sum_{i=0}^{N} \alpha_N^i \rho^i$$

and where α_N^i denotes the number of ways to have i concurrent active nodes.

A number of important measures can be obtained once the partition function is computed. The most important performance measure in a packet radio network is the nodal throughput S_i which is defined as the average number of successful transmissions processed by node i per unit of time. Note that it is not just

the average number of concurrent transmissions, since some of these will not be received by their destinations. The maximum nodal throughput is called the nodal capacity.

To calculate the nodal throughput, let us first calculate the link throughput $S_{i,j}$, the average number of successful transmissions over the i-to-j link per unit of time. Let $A_{i,j}$ denote the set of nodes that must be silent at the initiation of the i-to-j transmission to guarantee its success. The probability of success of the i-to-j transmission is then

$$P_{i,j} = P(A_{i,j} \text{ idle}) = \sum_{S \subset N \setminus A_{i,j}} \pi(S) = \frac{\sum_{S \subset N \setminus A_{i,j}} \rho^{|S|}}{Z_N} = \frac{Z_{N \setminus A_{i,j}}}{Z_N} \qquad (4.2.1)$$

Note that if the set of nodes N can be represented as $N = N_1 \cup N_2$ where N_1 and N_2 do not interfere with each other, then

$$Z_N = Z_{N_1} Z_{N_2} \qquad (4.2.2)$$

If $\rho_{i,j}$ denotes the traffic intensity for the packets from node i to node j, then from 4.2.1 we obtain

$$S_{i,j} = \rho_{i,j} P_{i,j} = \rho_{i,j} \frac{Z_{N \setminus A_{i,j}}}{Z_N} \qquad (4.2.3)$$

And therefore, the nodal throughput is given by

$$S_i = \sum_j S_{i,j} \qquad (4.2.4)$$

Before considering the examples, let us note the following lemma which will be used extensively throughout this chapter. The proof of this lemma is given in the Appendix at the end of this chapter.

Lemma 4.2.1. The function $f(t) = 1 - t - \rho t^n = 0$ where $\rho > 0$ has n distinct roots. There is only one positive root t_0. This root is the smallest in magnitude among all other roots. ∎

In particular, if $Z_G(t)$ is of the form

$$Z_G(t) = \frac{F(t)}{1 - t - \rho t^{n+1}}$$

and t_0 is a simple pole of $Z_G(t)$ (a simple root of $1 - t - \rho t^{n+1} = 0$ and $F(t_0) \neq 0$) then

$$Res[Z_G(t_0)] = -\frac{F(t_0)}{(n+1)\rho t_0^n + 1}$$

4.3. APPLICATIONS

4.3.1. TANDEM NETWORKS: CSMA

Consider a tandem of N packet radios operating under the Carrier Sense Multiple Access (CSMA) scheme ([Boor 80]). In such a system, all nodes share the same bandwidth and each node (except for the end nodes) can communicate with two neighbors (see Figure 4.3.1). To calculate the partition function Z_N we apply the canonical approximation as follows.

Step 1. To calculate the grand partition function, derive a recursive relation for α_N^i, the number of configurations involving i transmissions. To that end, suppose one more node is added to a tandem of size N. Let us examine a configuration involving i transmissions. There are clearly two cases to consider:

Case 1: the $(N+1)$-st radio is not transmitting. There are α_N^i such configurations.

Case 2: the $(N+1)$-st radio is involved in a transmission. There are α_{N-1}^{i-1} such configurations.

Hence,

$$\alpha^i_{N+1} = \alpha^{i-1}_{N-1} + \alpha^i_N \quad \text{for} \quad N \geq 1, 1 \leq i \leq N \qquad \text{and} \qquad \alpha^0_0 = \alpha^0_N = 1$$

This implies the following recursive relation

$$Z_{N+1} = Z_N + \rho Z_{N-1} \qquad \text{with} \qquad Z_0 = 1, Z_1 = 1 + \rho \tag{4.3.1.1}$$

Therefore, the grand partition function is

$$Z_G(t) = \sum_{N=0}^{\infty} Z_N t^N = \frac{1 + t\rho}{1 - t - \rho t^2} \tag{4.3.1.2}$$

Step 2. Find the smallest positive pole of the grand partition function.

$$t_0 = \frac{2}{1 + \sqrt{1 + 4\rho}}$$

Step 3. Compute the residue of the grand partition function at t_0

$$Res[Z_G(t_0)] = -\frac{1 + \rho t_0}{2\rho t_0 + 1} = -\frac{1 + \sqrt{1 + 4\rho}}{2\sqrt{1 + 4\rho}}$$

Step 4. The partition function is given by

$$Z_N \approx -\frac{Res[Z_G(t_0)]}{t_0^{N+1}} = \frac{1}{\sqrt{1 + 4\rho}} \left(\frac{1 + \sqrt{1 + 4\rho}}{2} \right)^{N+2} \tag{4.3.1.3}$$

For this particular example, it is easy to calculate the relative error. The second pole of the grand partition function is

$$t_1 = \frac{2}{\sqrt{1 + 4\rho} - 1} \tag{4.3.1.4}$$

The relative error of the approximation is easily calculated to be

$$\epsilon_N = \frac{Res[Z_G(t_1)]}{Res[Z_G(t_0)]}\left(\frac{t_0}{t_1}\right)^{N+1} = \left(\frac{\sqrt{1+4\rho}-1}{\sqrt{1+4\rho}+1}\right)^{N+2} \rightarrow 0 \qquad (4.3.1.5)$$

Thus, the relative error ϵ_N decreases exponentially with N.

To calculate the link throughput $S_{i,i+1}$ under the perfect capture assumption, consider a typical node i. The transmission of i to $i+1$ will be successful, if at the start of the transmission, all of the nodes in $A_{i,i+1} = \{i-1, i, i+1, i+2\}$ remain silent. This is shown in below in Figure 4.3.1.

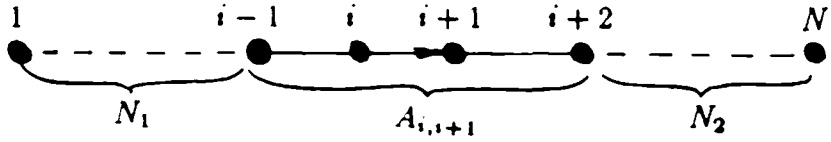

Figure 4.3.1: A Transmission Configuration (CSMA, $d = 1$)

The sets of activities generated by the subsets of nodes $N_1 = \{1, \ldots, i-2\}$ and $N_2 = \{i+3, \ldots, N\}$ are mutually non-interfering and correspond to two tandems of sizes $i-2$ and $N-i-2$ respectively. Therefore, using equations 4.2.1 and 4.2.2 one finds the probability of successful transmission from node i to node $i+1$ to be

$$P_{i,i+1} = P(A_{i,j} \text{ idle}) = \frac{Z_{i-2}Z_{N-i-2}}{Z_N} \approx \frac{1}{\sqrt{1+4\rho}}\left(\frac{2}{1+\sqrt{1+4\rho}}\right)^2 \qquad (4.3.1.6)$$

Assuming that node i is equally likely to transmit to node $i+1$ as to node $i-1$, (that is, $\rho_{i,i+1} = \rho/2$), the link throughput is given by

$$S_{i,i+1} = \frac{\rho}{2}P_{i,i+1} \approx \frac{\rho}{2\sqrt{1+4\rho}}\left(\frac{2}{1+\sqrt{1+4\rho}}\right)^2 \qquad (4.3.1.7)$$

If ϵ_N denotes the relative error of canonical approximation in calculating Z_N (equation 4.3.1.5), then from equations 4.3.1.6 and 4.3.1.7 we obtain the following expression for the relative error δ_N in calculating the nodal throughput (as

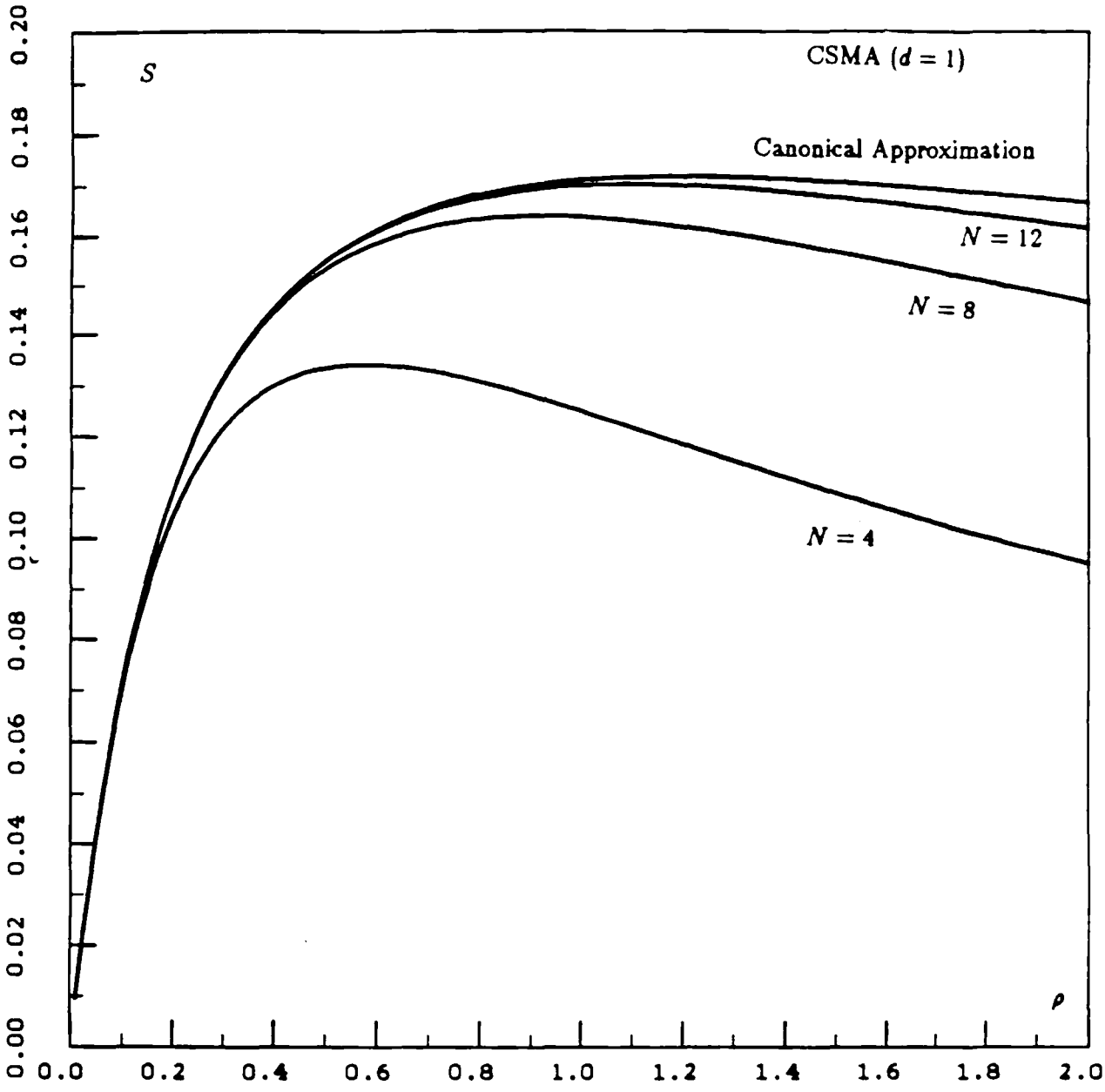

Figure 4.3.2: CSMA Nodal Throughput for Different N (d=1)

given by equation 4.3.1.7):

$$\delta_N = \frac{(1+\epsilon_{i-2})(1+\epsilon_{N-i-2})}{1+\epsilon_N} - 1 = \frac{\epsilon_{i-2}+\epsilon_{N-i-2}+\epsilon_{i-2}\epsilon_{N-i-2}-\epsilon_N}{1+\epsilon_N}$$

The relative error δ_N for the nodal throughput decreases exponentially with N.

The nodal throughput is obviously $S_i = 2S_{i,i+1}$. Figure 4.3.2 gives the nodal throughput as a function of the load for different values for N. For small N, the edge effects (the rightmost and leftmost node can only broadcast to one neighbor) are significant, reducing the average nodal throughput. As one increases N, these edge effects become less and less noticeable. Even for relatively small $N = 12$, the nodal throughput is very close to the one calculated by the canonical approximation.

Let us note the shape of the curves. As ρ increases from 0, so does the throughput. For light ρ, there is little interference. Therefore, the throughput increases with an increase in ρ. For large ρ, there is more interference and, in fact, a transmission will find it less likely to succeed. Therefore, after some point, the throughput decreases with an increase in ρ. The nodal capacity of $S_i = 0.174$ is achieved at $\rho = 1.2$. For $\rho = 1$ the throughput is 0.17 as has been shown in [Boor80, Toba83] by using numerical methods and simulation. However, one gains a slight improvement in the throughput if packet retransmission attempts are generated faster than the average transmission duration time ($\rho = 1.2$).

4.3.2. TANDEM NETWORKS: C-BTMA

Let us consider the tandem of N packet radios as before, but assume that it operates under the C-BTMA protocol. This means that a node can transmit only if its immediate neighbors as well as the neighbors of the immediate neighbors are silent. As before, we can derive a recurrence relation for the partition function by adding one more node to the tandem and examining the number of configurations

having i transmissions. One would get that for $N \geq 2$

$$\alpha_{N+1}^{i} = \alpha_{N}^{i} + \alpha_{N-2}^{i-1} \qquad \text{and} \qquad \alpha_{0}^{0} = \alpha_{1}^{0} = \alpha_{N}^{0} = 1$$

The partition function then satisfies

$$Z_{N+1} = Z_N + \rho Z_{N-2}, \quad N \geq 2 \quad \text{with} \quad Z_0 = 1, Z_1 = 1+\rho, Z_2 = 1+2\rho \quad (4.3.2.1)$$

From the above, the grand partition function satisfies:

$$Z_G(t) = \frac{1+\rho t+\rho t^2}{1-t-\rho t^3} \qquad (4.3.2.2)$$

Let t_0 be the smallest (positive) pole of $Z_G(t)$. The existence and uniqueness of this pole follows from lemma 4.2.1. By the same lemma, the residue of $Z_G(t)$ at t_0 is given by

$$Res[Z_G(t_0)] = -\frac{1+\rho t_0+\rho t_0^2}{3\rho t_0^2+1}$$

Applying the canonical approximation, one gets the following expression for the partition function

$$Z_N \approx -\frac{Res[Z_G(t_0)]}{t_0^{N+1}} = \frac{1+\rho t_0+\rho t_0^2}{(3\rho t_0^2+1)t_0^{N+1}} \qquad (4.3.2.3)$$

To calculate the throughput, consider a typical node i in a tandem. This node will be successful in a transmission to node $i+1$ if the set of nodes one and two hops away $A_{i,i+1} = \{i-2, i-1, i, i+1, i+2\}$ are silent. The sets of activities generated by the subsets of nodes $N_1 = \{1, 2, \ldots, i-3\}$ and $N_2 = \{i+3, \ldots, N\}$ are mutually independent and correspond to two tandems of sizes $i-3$ and $N-i-2$, respectively. This is illustrated below in Figure 4.3.3.

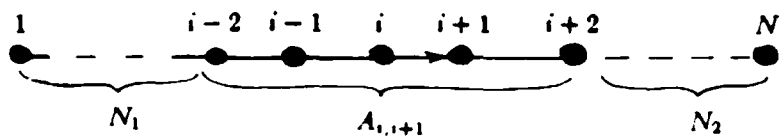

Figure 4.3.3: A Transmission Configuration (C-BTMA, $d = 1$)

Therefore, the probability of success

$$P_{i,i+1} = \frac{Z_{i-3}Z_{N-i-2}}{Z_N} \approx \frac{1+\rho t_0+\rho t_0^2}{(3\rho t_0^2+1)} t_0^4$$

Assuming that a node is equally likely to talk to any two of its neighbors, the link throughput $S_{i,i+1}$ is given by

$$S_{i,i+1} = \frac{\rho}{2} P_{i,i+1} \approx \frac{\rho}{2} \frac{1+\rho t_0+\rho t_0^2}{(3\rho t_0^2+1)} t_0^4$$

The nodal throughput is

$$S_i = 2S_{i,i+1} = \rho \frac{1+\rho t_0+\rho t_0^2}{(3\rho t_0^2+1)} t_0^4 \qquad (4.3.2.4)$$

To calculate the capacity, note that once a node is permitted to start a transmission, the success of transmission is guaranteed. Thus, the capacity is achieved at $\rho \mapsto \infty$. For large ρ the pole can be approximated by $t_0 \approx \rho^{-\frac{1}{3}}$. With this approximation

$$S_i = \frac{\rho(1+\rho^{\frac{2}{3}}+\rho^{\frac{1}{3}})}{3\rho^{\frac{1}{3}}+1} \rho^{-\frac{4}{3}} \mapsto \frac{1}{3}$$

The throughput per node is $\frac{1}{3}$. This is what one expects: as $\rho \mapsto \infty$ the tandem will be densely packed with transmissions. One would expect every 3-rd node to be active.

It is interesting to compare the above results to the simulation studies on ring networks reported in [Toba83]. The link throughput for a ring under C-BTMA exhibits a quasi-periodicity of period 3: all rings with a number of nodes which is a multiple of 3 have a little higher throughput than those which are not multiples of 3. The difference decreases as the number of nodes increases. This can be explained

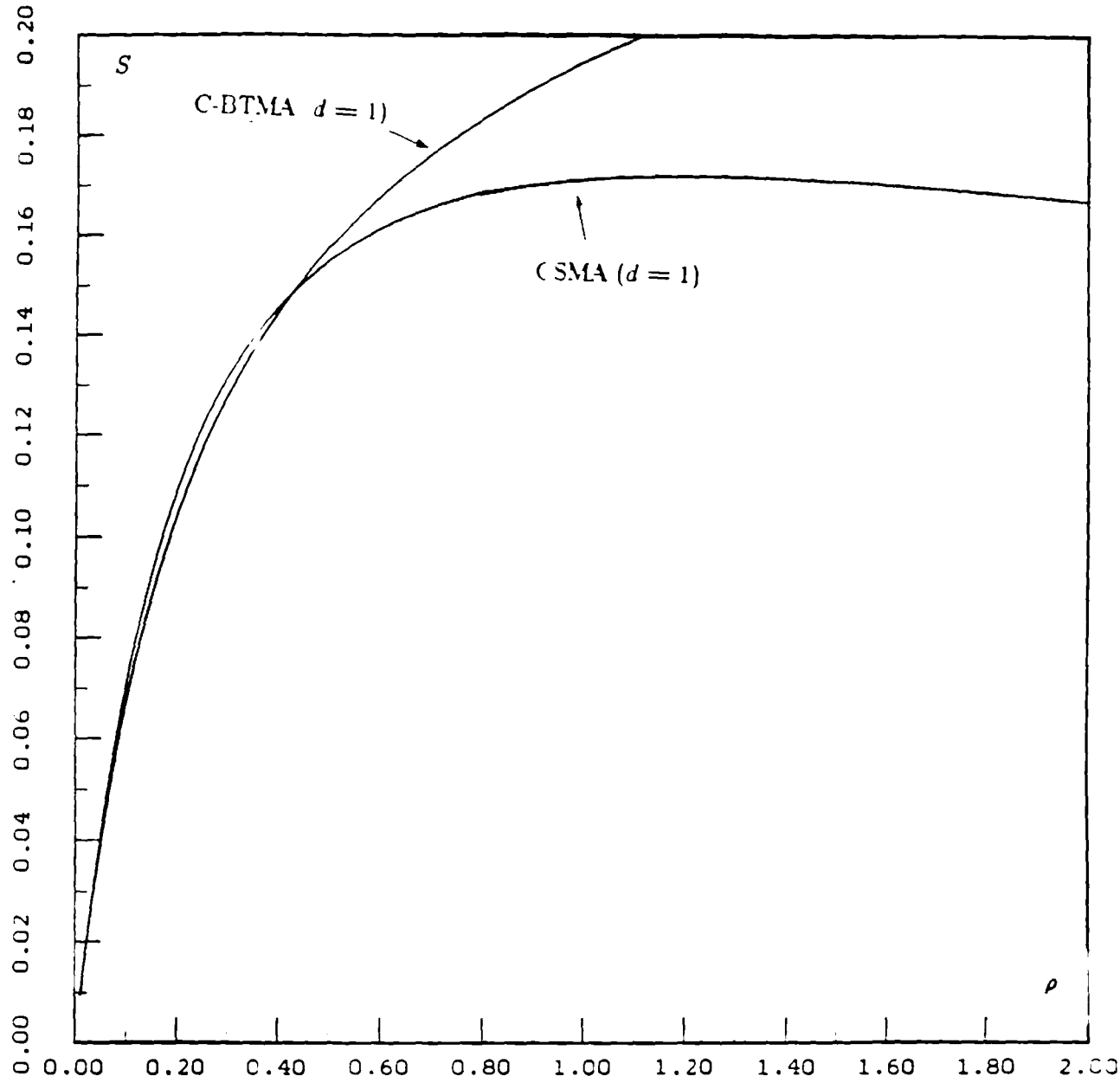

Figure 4.3.4: Nodal Throughput for CSMA and C-BTMA (d=1)

using canonical approximation. First, note that the tandem and ring exhibit similar behavior, especially for large N. The grand partition function (equation 4.3.2.2) has three poles - one is t_0 (real, positive, and smallest in magnitude) and two are complex conjugates. If one considers the exact expression of the partition function, it is dominated by the smallest root. However, for small N, the contributions from the complex roots are not negligible. These contributions are the largest when N is a multiple of 3. As N increases, the contributions from these complex roots become smaller as the partition function is increasingly dominated by t_0. For large N, the partition function is insensitive to the divisibility of N by 3. The simulation results were reported for relatively small size ($N < 20$) rings.

Figure 4.3.4 gives the the nodal throughput and compares it to the one using CSMA with perfect capture. For $\rho < 0.43$, CSMA outperforms C-BTMA. This says that for lighter loads, there is no need to be overly restrictive ("conservative"). For heavier loads, when there is a lot of interference, greater restriction helps.

4.3.3. LINEAR ARRAY: CSMA

Consider a packet radio network of N nodes placed on a linear array of degree $2d$, that is nodes can transmit up to d nodes in either direction. A typical transmission configuration for a linear array of degree 2 is illustrated in the Figure 4.3.5.

To calculate the partition function Z_N one can derive a recursive relation for α_N^i as follows. Suppose one more node is added to the network. Let us examine a configuration involving i transmissions. There are clearly two cases to consider:

Case 1: the $N+1$-st radio is not transmitting. There are α_N^i such configurations.

Case 2: the $N+1$-st radio is involved in a transmission. There are α_{N-d}^{i-1} such configurations.

Hence, $\alpha^i_{N+1} = \alpha^{i-1}_{N-d} + \alpha^i_N$ and with initial conditions $\alpha^i_N = 1$ for $N < d$.

The above relation implies the following

$$\begin{cases} Z_{N+1} = Z_N + \rho Z_{N-d} & \text{for } N \geq d \\ Z_N = 1 + \rho N & \text{for } 0 \leq N < d \end{cases} \tag{4.3.3.1}$$

It follows then

$$Z_G(t) - \sum_{k=0}^{d} Z_k t^k = t\left[Z_G(t) - \sum_{i=0}^{d-1} Z_i t^i\right] + \rho t^{d+1} Z_G(t)$$

which after some algebraic manipulations reduces to

$$Z_G(t) = \frac{t - 1 + \rho t^{d+1} - \rho t}{(t-1)(1 - t - \rho t^{d+1})} \tag{4.3.3.2}$$

Let t_0 be the smallest positive pole of $Z_G(t)$. The existence and uniqueness of this pole follows from lemma 4.2.1. By the same lemma, the residue

$$Res[Z_G(t_0)] = -\frac{t_0 - 1 + \rho t_0^{d+1} - \rho t_0}{(t_0 - 1)((d+1)\rho t_0^d + 1)} = \frac{-\rho t_0^2}{(1 - t_0)[1 + d(1 - t_0)]}$$

Applying the canonical approximation, one obtains the following expression for the partition function:

$$Z_N \approx -\frac{Res[Z_G(t_0)]}{t_0^{N+1}} = \frac{\rho t_0^2}{(1 - t_0)[1 + d(1 - t_0)]t_0^{N+1}} \tag{4.3.3.3}$$

To calculate the throughput, consider a typical node i and let $S_{i,i+k}$ $(0 < k \leq d)$ be the throughput of a link connecting nodes i and $i + k$ which are k hops apart. Since node i can communicate with up to d successive nodes in either direction, under the perfect capture assumption, the transmission to node $i + k$ $(0 < k \leq d)$ will be successful if the set of nodes $A_{i,i+k} = \{i - d, i - d + 1, \ldots, i -$

$1, i, i+1, \ldots, i+k+d\}$ are silent at the initiation of that transmission. This is shown in Figure 4.3.5.

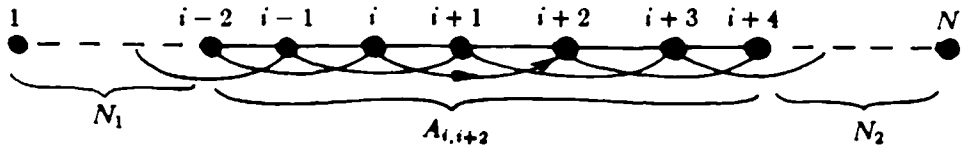

Figure 4.3.5: A Transmission Configuration (CSMA, $d = 2$)

The probability of success is therefore

$$P_{i,i+k} = P(A_{i,i+k} \text{ idle}) = \frac{Z_{N \setminus A_{i,i+k}}}{Z_N} = \frac{Z_{i-d-1} Z_{N-i-k-d}}{Z_N} \approx -Res[Z_G(t_0)] t_0^{2d+k}$$

Let us consider the case where the traffic is equally distributed among $2d$ outgoing links from node i, that is, $\rho_{i,i+k} = \rho/2d$. In such a case, the link throughput

$$S_{i,i+k} = \frac{\rho}{2d} P_{i,i+k} \approx \frac{\rho^2 t_0^{2d+k+2}}{2d(1-t_0)[1+d(1-t_0)]}$$

The nodal throughput of node i,

$$S_i = 2\sum_{k=1}^{d} S_{i,i+k} \approx \frac{\rho^2 t_0^{2d+3}}{d(1-t_0)[1+d(1-t_0)]} \frac{1-t_0^d}{1-t_0} = \frac{t_0(1-t_0^d)}{d[1+d(1-t_0)]} \qquad (4.3.3.4)$$

Figure 4.3.6. gives the curves of the nodal throughput as a function of load for $d = 3$, $d = 5$ and $d = 10$. The corresponding capacities are $S_i = 0.0826$ at $\rho = 0.73$, $S_i = 0.0544$ at $\rho = 0.53$ and $S_i = 0.0293$ at $\rho = 0.31$.

Let us calculate the capacity for large d. From equation 4.3.3.4 one differentiates the nodal throughput S_i with respect to t_0 and solves $S'(t_0) = 0$. We obtain

$$d^2 t_0^{d+1} - (d+1)^2 t^d + d + 1 = 0$$

For large d, the pole t_0 is very close to 1. Therefore, writing $t_0 = 1-\alpha$ and using the approximation $(1-\alpha)^d \approx 1 - d\alpha$, we find that the capacity is achieved at

$$t_0 \approx 1 - \frac{1}{d+1}$$

From equation 4.3.3.4, we arrive at a capacity of

$$S_i \approx \frac{\frac{d}{d+1}\left[1 - \left(1 - \frac{1}{d+1}\right)^d\right]}{d\left(1 + \frac{d}{d+1}\right)} \approx \frac{1 - \frac{1}{e}}{2d} = \frac{0.318}{d}$$

The corresponding load

$$\rho = \frac{1 - t_0}{t_0^{d+1}} \approx \frac{e}{d}$$

4.3.4. LINEAR ARRAY: C-BTMA

Consider the same linear array of packet radio nodes but assume they are now operating under the C-BTMA scheme. As before, we can derive a recurrence relation for the partition function by adding one more node and considering the corresponding two cases. Because of C-BTMA, the recurrence relation becomes $\alpha_{N+1}^i = \alpha_N^i + \alpha_{N-2d}^{i-1}$ for $N \geq 2d$ with initial conditions $\alpha_N^1 = N$ for $N < 2d$

This gives

$$\begin{cases} Z_{N+1} = Z_N + \rho Z_{N-2d} & \text{for } N \geq 2d \\ Z_N = 1 + \rho N & 0 \leq N < 2d \end{cases} \qquad (4.3.4.1)$$

The grand partition function $Z_G(t)$ can be shown to satisfy

$$Z_G(t) = \frac{t - 1 + \rho t^{2d+1} - \rho t}{(t-1)(1 - t - \rho t^{2d+1})} \qquad (4.3.4.2)$$

Let t_0 be the smallest positive pole. The existence and uniqueness of this pole follows from lemma 4.2.1. By the same lemma, the residue

$$Res[Z_G(t_0)] = -\frac{t_0 - 1 + \rho t_0^{2d+1} - \rho t_0}{(t_0 - 1)(1 + \rho(2d+1)t_0^{2d})} = -\frac{\rho t_0^2}{(1 - t_0)\,[1 + 2d(1 - t_0)]}$$

The partition function is given by

$$Z_N \approx -\frac{Res[Z_G(t_0)]}{t_0^{N+1}} = \frac{\rho t_0^2}{(1 - t_0)\,[1 + 2d(1 - t_0)]\,t_0^{N+1}} \qquad (4.3.4.3)$$

As in the previous case, define $S_{i,i+k}$ $(0 < k \leq d)$ to be the link throughput between nodes i and $i+k$ that are k hops apart. Because of the C-BTMA, the node i will be successful at every transmission to $i + k$. It can initiate a transmission if the set of nodes $A = \{i - 2d, i - 2d + 1, \ldots, i + 2d\}$ is silent. The probability of this is

$$P_{i,i+k} = \frac{Z_{i-2d-1} Z_{N-i-2d}}{Z_N} \approx Res[Z_G(t_0)]t_0^{4d} = \frac{\rho t_0^{4d+2}}{(1 - t_0)\,[1 + 2d(1 - t_0)]}$$

Assuming that node i is equally likely to talk to any of its $2d$ neighbors, the link throughput is given by

$$S_{i,i+k} = \frac{\rho}{2d} P_{i,i+k} \approx \frac{\rho^2 t_0^{4d+2}}{2d(1 - t_0)\,[1 + 2d(1 - t_0)]}$$

The nodal throughput

$$S_i = 2\sum_{i=1}^{d} S_{i,i+k} \approx \frac{\rho^2 t_0^{4d+2}}{(1 - t_0)\,[1 + 2d(1 - t_0)]} \qquad (4.3.4.4)$$

Since every transmission results in success, the capacity is achieved at $\rho \mapsto \infty$. For large ρ the pole can be approximated as follows

$$t_0 \approx \rho^{-\frac{1}{2d+1}}$$

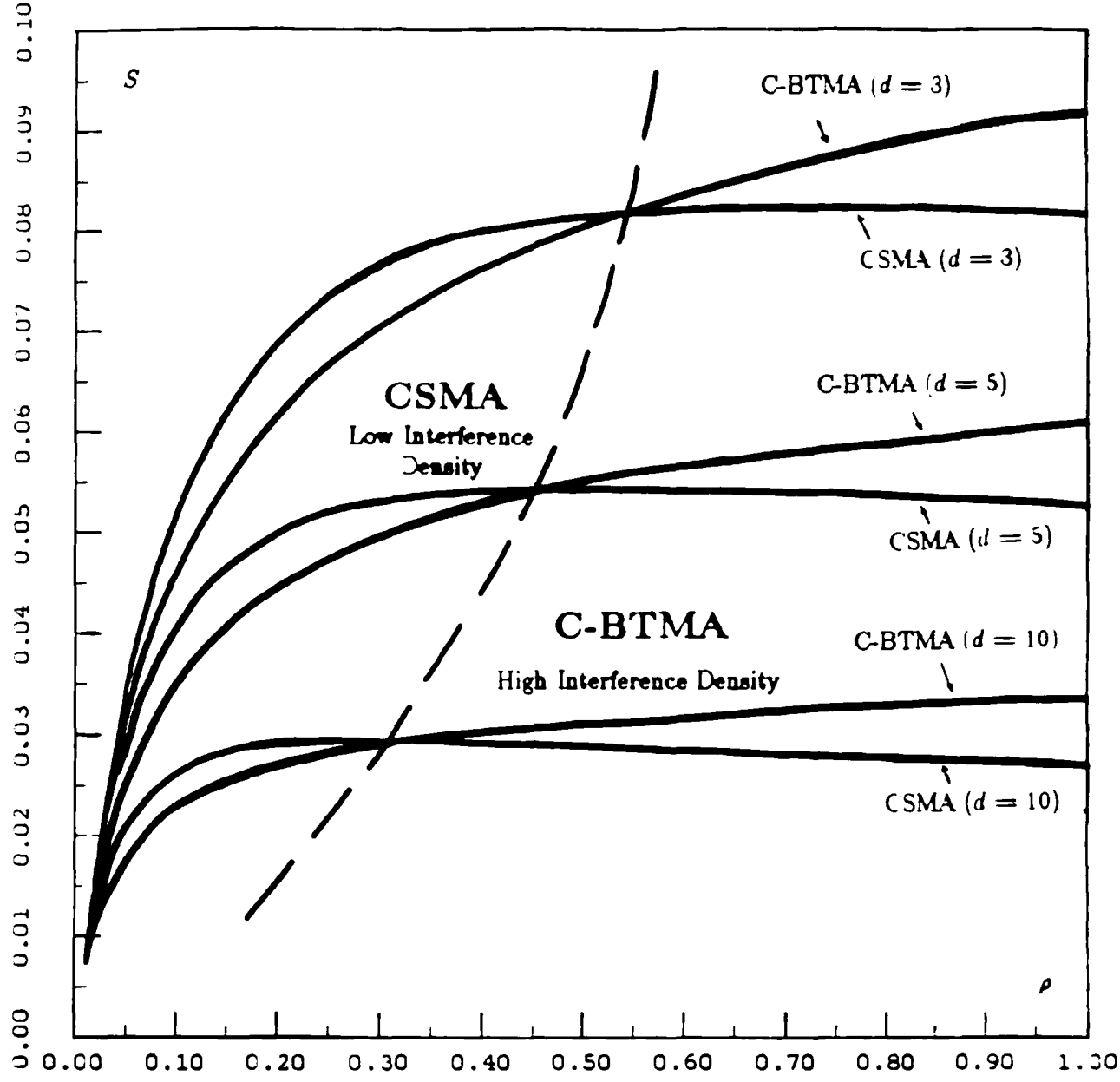

Figure 4.3.6: Nodal Throughput: CSMA and C-BTMA (d=3,5,10)

The asymptotic nodal throughput is easily seen to be $S_i = 1/(2d+1) \approx 0.5/d$. This is what one would intuitively expect: for very large ρ the linear array will be packed with transmissions. Since every i-to-j transmission blocks $2d+1$ nodes, and the system is densely packed, one would expect that the throughput per node is $1/(2d+1)$.

It is interesting to compare the capacity S_i^{max} for CSMA with perfect capture and C-BTMA for large d.

$$S_i^{max} \approx \frac{0.318}{d} \qquad \text{for CSMA and} \qquad S_i^{max} \approx \frac{0.5}{d} \text{ for C-BTMA}$$

This says that for the same (large) d, the capacity under CSMA is 36% worse than under C-BTMA. This is what one can intuitively expect. By being more restrictive, C-BTMA avoids the situations where a transmission cannot be guaranteed of success.

Figure 4.3.6 gives the nodal throughput and compares it with CSMA with perfect capture for $d = 3$,$d = 5$ and $d = 10$. Just as in the case of $d = 1$, for lighter loads CSMA outperforms C-BTMA. For each d, canonical approximation allows one to identify (analytically) the regions when CSMA outperforms C-BTMA. These regions are indicated in Figure 4.3.6. Note that as the level of interference increases (higher load ρ, or denser topology - larger d), the range of the loads for which CSMA is better decreases. This says that one should be more "restrictive" as the interference increases.

4.4. ANALYSIS OF ZERO CAPTURE

The analysis of zero capture is very difficult, because the probability of successful transmission depends upon the duration of the transmission. The analysis can be done by simulation ([Toba85]) or by the explicit construction of a Markov

chain describing the activity of a packet radio network ([Braz85]). In the latter case, only very small ($N = 4$) networks can be analyzed. Our approach here is to derive both a lower bound and an easy approximation for the link throughput under zero capture†.

Let $A_{i,j}$ be defined as before and let $B_{i,j}$ be the set of nodes that can transmit but must remain silent during the i-to-j transmission. Obviously, $B_{i,j} \subseteq A_{i,j}$. To guarantee the success of the i-to-j transmission, every node in $B_{i,j}$ must remain silent throughout this transmission. These nodes may be silent if they are blocked by other active transmissions or if they have no data to transmit when they are not blocked. This is shown in Figure 4.4.1.

To obtain a lower bound on $S_{i,j}$, one assumes that nodes in $B_{i,j}$ are silent during the i-to-j transmission because they have no data. To calculate the probability that nodes in $B_{i,j}$ are silent during the i-to-j transmission, let the random variable $\tau_{i,j}$ denotes the time duration of i-to-j transmission (recall that it is distributed exponentially with parameter μ). We obtain

$$P(B_{i,j} \text{ idle}) = \int_0^\infty P(B_{i,j} \text{ idle} | \tau_{i,j} = x) P(\tau_{i,j} = x) dx = \int_0^\infty e^{-\lambda |B_{i,j}| x} \mu e^{-\mu x} dx = \frac{1}{1 + \rho |B_{i,j}|} \tag{4.4.1}$$

The lower bound for probability that the i-to-j transmission is successful is then

$$P_{i,j}^{L.B.} = \frac{Z_{N \setminus A_{i,j}}}{Z_N} \frac{1}{1 + \rho |B_{i,j}|} \tag{4.4.2}$$

The lower bound for the throughput under zero capture is therefore

$$S_{i,j}^{L.B.} = \rho_{i,j} P_{i,j} = \rho_{i,j} \frac{Z_{N \setminus A_{i,j}}}{Z_N} \frac{1}{1 + \rho |B_{i,j}|} \tag{4.4.3}$$

† This approximation was developed jointly with Professor Moshe Sidi of the Technion.

Now, let us consider the following approximation. During the time $\tau_{i,j}$, a node $k \in B_{i,j}$ is silent because it is blocked by other nodes or it has no data to send. It is easy to calculate the probability α that all nodes in $B_{i,j}$ are silent

$$\alpha = \frac{Z_{N \setminus B_{i,j}}}{Z_N} \tag{4.4.4}$$

Then $\alpha\tau_{i,j}$ is the "vulnerable" period in which node $k \in B_{ij}$ can start transmission but should remain silent. This suggests the following expression

$$P(B_{i,j} \text{ idle}) = \int_0^\infty e^{-\lambda|B_{i,j}|\alpha x}\mu e^{-\mu x}dx = \frac{1}{1+\alpha\rho|B_{i,j}|} \tag{4.4.5}$$

This yields the following approximation for the link throughput for CSMA with zero capture:

$$S_{i,j} = \rho_{i,j}\frac{Z_{N \setminus A_{i,j}}}{Z_N}\frac{1}{1+\alpha\rho|B_{i,j}|} \tag{4.4.6}$$

The nodal throughput S_i for node i is obviously

$$S_i = \sum_j S_{ij} \tag{4.4.7}$$

We apply these ideas to analyze the tandem ($d = 1$). First, we establish the lower bound. The set of nodes that must be silent at the initiation of an i-to-$i+1$ transmission is $A_{i,i+1} = \{i-1, i, i+1, i+2\}$. To guarantee the success of this transmission, node $i+2$ must remain silent. Hence, $B_{i,i+1} = \{i+2\}$. The lower bound for the nodal throughput is then

$$S_i^{L.B.} = \rho\frac{Z_{N \setminus A_{i,i+1}}}{Z_N}\frac{1}{1+\rho|B_{i,i+1}|} \approx \rho\frac{Z_{i-2}Z_{N-i-2}}{Z_N(1+\rho)}$$

$$\approx \frac{\rho}{\sqrt{1+4\rho}(1+\rho)}\left(\frac{2}{1+\sqrt{1+4\rho}}\right)^2$$

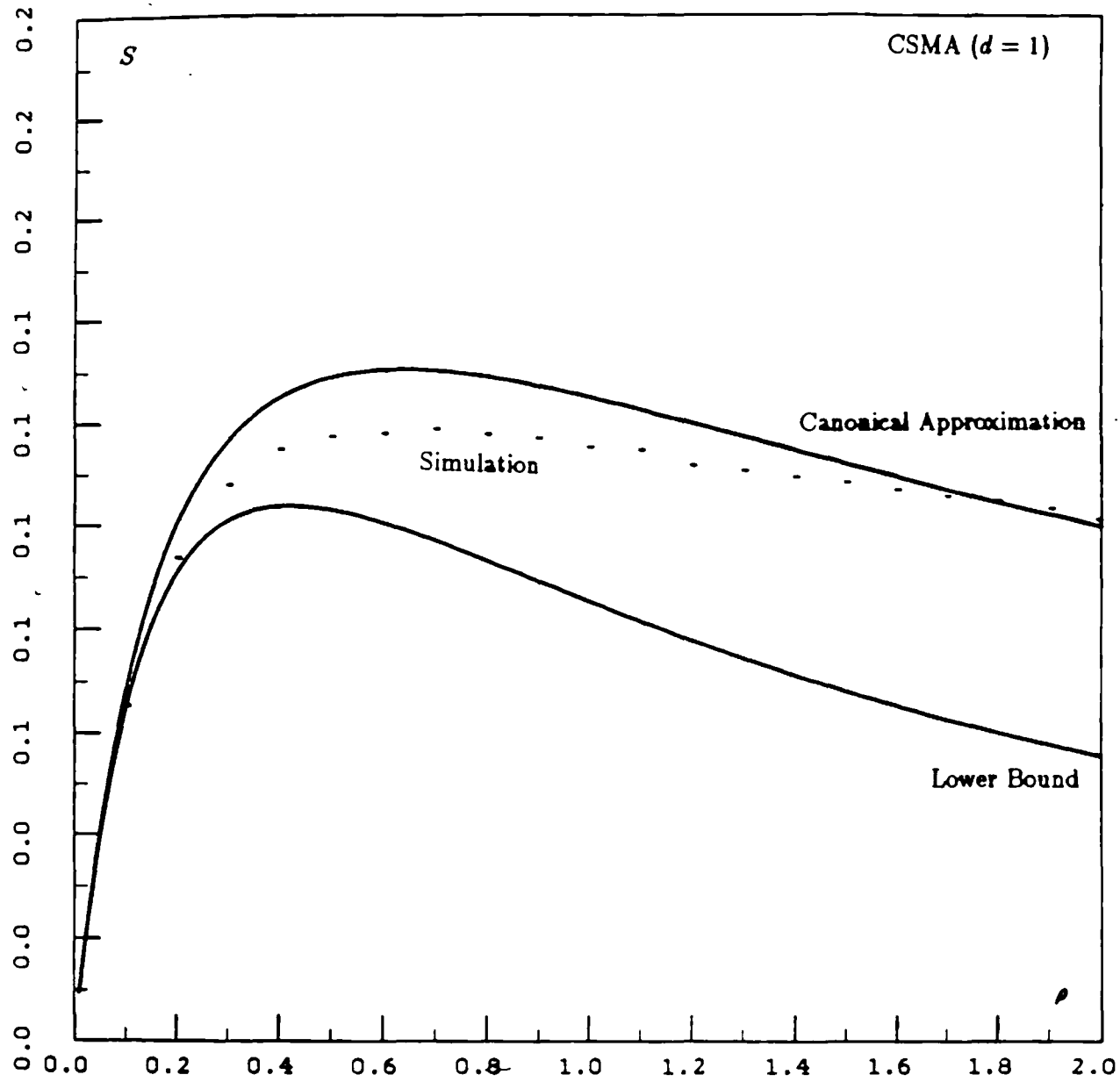

Figure 4.4.1: CSMA Nodal Throughput for Zero Capture.

Now, let us apply the approximation as given by equation 4.4.6. The probability α is easily found to be

$$\alpha = \frac{Z_{N \setminus B_{i,j}}}{Z_N} = -Res[Z_G(t_0)] = \frac{1 + \sqrt{1 + 4\rho}}{2\sqrt{1 + 4\rho}}$$

The "vulnerable period" approximation is then

$$S_i \approx \frac{\rho}{\sqrt{1 + 4\rho}(1 + \alpha\rho)} \left(\frac{2}{1 + \sqrt{1 + 4\rho}} \right)^2$$

The corresponding curves are shown in Figure 4.4.1. The "vulnerable period" approximation is close to the results of the simulation runs on a 100-node tandem. For low values of ρ, the lower bound curve is close to the simulation results but diverges as the load is increased. This is easily explained. For low ρ, it is likely that a node is silent because it has no data. For high ρ, it is likely that it has data but is blocked by an active neighboring node. The comparison with simulation results indicate that the "vulnerable period" approximation is fairly accurate.

4.5. CONCLUSION

This chapter presented an application of the new method of canonical approximation to analyze CSMA and C-BTMA protocols with perfect capture for a number of network topologies. A lower bound and an easy approximation for CSMA with zero capture was presented. The method allows us to analyze a number of network topologies with relative ease. Further work will extend the application of the new method to study other multiple access protocols and network topologies.

4.6. APPENDIX

In this appendix we give a proof of lemma 4.2.1:

Lemma 4.2.1. The function $f(t) = 1 - t - \rho t^n = 0$ where $\rho > 0$ has n distinct roots. There is only one positive root t_0. This root is the smallest in magnitude among all other roots.

Proof. First, let us show that every root of $f(t) = 0$ is simple. If t_i is a root of order 2 or higher then obviously $f(t_i) = 0$ and $f'(t_i) = 0$. Now,

$$f'(t_i) = -1 - n\rho t_i^{s-1} = -1 - \frac{n(1-t_i)}{t_i}$$

If $f'(t_i) = 0$ then $t_i = n/(n-1)$. But then $f(n/(n+1)) \neq 0$. This contradiction shows that every root of $f(t) = 0$ is simple.

Since $f(0) = 1 > 0$ and $f(1) = -\rho < 0$ the function has a root t_0 satisfying $0 < t_0 < 1$. Moreover, since $f'(t) < 0$ for $t > 0$ it follows that $f(t)$ is a decreasing function for positive t and hence, the root is unique.

For any other root t_1 one can write $t_1 = R\cos\theta + iR\sin\theta$ where $\theta \neq 0$. Since t_1 satisfies $1 - t_1 - \rho t_1^n = 0$ one obtains

$$g(R,\theta) = 1 - R\cos\theta - \rho R^n \cos n\theta = 0$$

But $g(R,\theta) > 1 - R - \rho R^n = f(R)$. The function $f(R)$ has only one positive real root $R = t_0$. Moreover, the function $f(R)$ is decreasing for larger R. Therefore, $g(R,\theta)$ is bounded below by a decreasing function intersecting the X-axis at t_0. But this implies $t_1 > t_0$. Therefore, t_0 is the smallest in magnitude among all the roots. ∎

CHAPTER 5

CANONICAL APPROXIMATION: DATABASE SYSTEMS

5.1. INTRODUCTION

In this chapter the method of canonical approximation is applied to analyze a general model of static locking policy in database systems with N items and p classes of transactions. The model has been recently analyzed by D. Mitra, P. Weinberger and S. Lavenberg ([Mitr84, Lave84]). They obtained an exact recursive algorithm to compute the partition function, whose computational complexity is of $O(Np)$. The exact expressions for average concurrency, throughput and non-blocking probability for each class of transactions involve an explicit calculation of the partition function. The asymptotic ($N \mapsto \infty$) analysis ([Lave84, Mitr84]) requires negligible computation but assumes very low traffic. The central focus of the analysis is the interference phenomenon among transactions, which is the cause of conflicts in a database. This is unlike most of the previous analytic studies of locking (see Chapter 2), which concentrated on the trade-off between locking overhead and the degree of allowable parallelism ([Iran78, Poti80]).

Canonical approximation gives a simple closed form approximation to the partition function and performance measures. The approximation requires finding a root of a simple polynomial. Average concurrency, throughput and non-blocking probability for each class of transactions are computed without an explicit calculation of the partition function. Since the magnitude of Z_N can exceed a floating point range of a typical computer even for modest values of N, canonical approximation allows the design of stable algorithms to analyze the model for any range of parameters (e.g. N and load values), in particular, in heavy traffic. The complexity of canonical approximation is of $O(j_p \log N)$, where j_p is the largest number of items

which can be locked by any transaction. We will show that there is a range of load parameters, beyond which the database will not have any large transactions. Using canonical approximation, we will compute these load parameters at which this "change" takes place. This phenomenon will be explained using the analogy from the condensation of imperfect gas ([Path84]).

5.2. THE MODEL

The model described in this section is the one analyzed by D. Mitra and P. Weinberger ([Mitr84]) and by S. Lavenberg ([Lave84]). The results of this section are taken from [Mitr84]. They are presented here for the sake of completeness.

A database consists of N items. An item is the smallest entity in the database which may be locked. There are p classes of transactions. A transaction of class σ requires j_σ items. Without loss of generality, assume $j_1 \leq \cdots \leq j_p$. Thus, there are $\binom{N}{j_\sigma}$ transactions of class σ. In this model no distinction is made between items that are to be read from the others that are to be written or updated. Executing transaction makes no use of the temporal ordering of the processing of items. Transactions interfere if they request the same items.

Requests for processing arrive exogenously to the database. The stream of requests to process a particular transaction of class σ is assumed to be Poisson with the rate λ_σ. On arrival of such a request, the database lock manager determines if any of the items for the new transaction are already locked. If so, the request is cleared from the system. Otherwise, a transaction is accepted. The lock manager places locks on the required items in one atomic action.

Usually, locks are obtained during the course of a transaction and released when the transaction commits or aborts. If the part of the transaction during which locks are being obtained is short, then our model is a good approximation. In other

words, blocking in this model corresponds closely to rejecting a transaction using optimistic concurrency control ([Bern81]). Just as in optimistic concurrency control, the conflict detection is done at one time. The conflicting transaction is aborted, so that no deadlock or waiting is involved. In the optimistic concurrency control, the transaction commits or aborts at the conclusion. However, in our model, this is done at the beginning of transaction processing.

The processing time for a transaction of class σ is assumed to be distributed exponentially with the mean $1/\mu_\sigma$†. The transactions that are accepted for processing are processed concurrently.

One of the assumptions of this model is that blocked requests are lost. In real systems, blocked transactions are resubmitted for processing. However, blocking systems often provide fundamental insights even into lossless systems. D. Mitra and P. Weinberger ([Mitr84]) present a scheme for approximating performance measures for lossless systems by using the results from blocking schemes.

One can argue that it is more realistic to consider both exclusive (write) and non-exclusive (read) locking. However, the model becomes much more difficult to analyze. At this point, it is not known whether there is an efficient way to exactly compute Z_N if non-exclusive locks are allowed. A simple asymptotic expression for non-blocking probabilities for this model in low traffic was obtained recently by S. Lavenberg ([Lave84]).

With the above assumptions, the model corresponds to the interference model considered in Chapter 3. Nodes of the graph corresponds to a set of data items. If a transaction arrives requiring this set of data items, the corresponding node becomes active. Recall that the equilibrium probability distribution for the

---

† This condition can be relaxed ([Mitr84]). The processing time for a transaction can be assumed to be an independent random variable with an arbitrary but common distribution for all transactions in a class with mean $1/\mu_\sigma$

model is

$$\pi(n_1, \ldots, n_p) = \frac{\rho_1^{n_1} \cdots \rho_p^{n_p}}{Z_N}$$

where $\rho_\sigma = \lambda_\sigma / \mu_\sigma$ and the partition function Z_N is given by

$$Z_N = \sum_{i_\sigma} \alpha_N^{i_1, \ldots, i_p} \rho_1^{i_1} \cdots \rho_p^{i_p}$$

Here $\alpha_N^{i_1, \ldots, i_p}$ is the number of distinct configurations such that there are i_σ concurrent transactions of class σ.

As noted above, one can drop the assumption of exponential service time for transactions of class σ and consider any service distribution with (finite) mean $1/\mu_\sigma$.

To analyze the system, one needs to calculate the partition function Z_N. Possible configurations of the model can be described by the set S of the p-tuples $S = \{I = (i_1, \ldots, i_p) : \quad i_1 j_1 + \cdots + i_p j_p \leq N\}$ where there are i_σ transactions of class σ. Let J denote the p-tuple $(j_1, \ldots, j_p)$ and let $I'J = \sum_{k=1}^{p} i_k j_k$. It is obvious that if $I \in S$ then $I'J \leq N$. The partition function is given by the following lemma:

Lemma 5.2.1 ([Mitr84]). The partition function is given by

$$Z_N = \sum_{I \in S} \frac{N!}{(N - I'J)!} \frac{1}{(j_1!)^{i_1} \cdots (j_p!)^{i_p}} \frac{1}{i_1! \cdots i_p!} \rho_1^{i_1} \cdots \rho_p^{i_p} \qquad (5.2.1)$$

Proof. Consider the term

$$\binom{N}{I'J} \left[\frac{(I'J)!}{(j_1!)^{i_1} \cdots (j_p!)^{i_p}} \right] \left[\frac{1}{i_1! \cdots i_p!} \right]$$

The first ratio is the number of ways to select the locked items. The second ratio is the number of ways to distribute the items into the queries. The third corrects for the fact that the order of queries is not important.

Hence,

$$\alpha_N^{i_1,\ldots,i_p} = \frac{N!}{(N-I'J)!}\frac{1}{(j_1!)^{i_1}\cdots(j_p!)^{i_p}}\frac{1}{i_1!\cdots i_p!}$$

and thus the partition function is given by the expression (5.2.1). ∎

The number of terms in the above expression for the partition function is of $O(N^p)$. Direct evaluation of Z_N is computationally expensive. D. Mitra and P. Weinberger ([Mitr84]) have obtained a recursive algorithm to compute Z_N in terms of $Z_1,\ldots,Z_{N-1}$. Its computational complexity is of $O(Np)$. The computation of the average concurrency and non-blocking probability of each class requires an explicit calculation of the partition function. The asymptotic analysis (as $N \mapsto 0$) requires negligible computation but assumes low traffic (e.g. $\binom{N}{j}\rho = o(1)$).

The next lemma gives the closed-form expression for the (exponential) grand partition function. The expression for $Z_G(t)$ was obtained in [Mitr84] to produce a numerical algorithm to calculate Z_N in terms of $Z_1,\ldots,Z_{N-1}$:

Lemma 5.2.2 ([Mitr84]). The grand partition function is given by

$$Z_G(t) = \sum_{N=0}^{\infty}\frac{Z_N}{N!}t^N = \exp(t + \sum_{\sigma} c_\sigma t^{j_\sigma}) \tag{5.2.2}$$

where $c_\sigma = \rho_\sigma / j_\sigma!$.

Proof.

$$Z_G(t) = \sum_{N=0}^{\infty} t^N \sum_{I\in S}\frac{1}{(N-I'J)!}\prod_{\sigma}\frac{c_\sigma^{i_\sigma}}{i_\sigma!} = \sum_{N=0}^{\infty}\sum_{I\in S}\frac{t^{N-I'J}}{(N-I'J)!}\prod_{\sigma}\frac{(t^{j_\sigma}c_\sigma)^{i_\sigma}}{i_\sigma!}$$

∎

$$= \sum_{I\in S}\sum_{N=0}^{\infty}\frac{t^N}{N!}\prod_{\sigma}\frac{(t^{j_\sigma}c_\sigma)^{i_\sigma}}{i_\sigma!} = e^t\prod_{\sigma}e^{c_\sigma t^{j_\sigma}} = \exp(t + \sum_{\sigma} c_\sigma t^{j_\sigma})$$

5.3. CANONICAL APPROXIMATION

Let us now apply the method of canonical approximation. The grand partition function is an entire function and therefore one applies the canonical approximation method by using the saddle point. Let

$$C(t) = t + \sum_{\sigma=1}^{p} c_\sigma t^{j_\sigma} \qquad \text{and} \qquad f(t) = C(t) - (N+1)\log t \tag{5.3.1}$$

The saddle-point t_N is found from $f'(t_N) = 0$. It is the unique positive root of the following j_p-th order equation:

$$t_N C'(t_N) - (N+1) = 0 \tag{5.3.2}$$

The second derivative of $f(t)$ at the saddle-point is

$$f''(t_N) = \frac{N+1}{t_N^2} + C''(t_N) \tag{5.3.3}$$

Canonical approximation (theorem 3.2.1) gives the following expression for the partition function:

$$Z_N = \frac{N!\exp(C(t_N))}{t_N^{N+1}\sqrt{2\pi f''(t_N)}}\,(1+\epsilon_N) \tag{5.3.4}$$

The relative error ϵ_N of the canonical approximation is given by

$$\epsilon_N = \frac{1}{24}(3\nu_4 - 5\nu_3^2) + \frac{1}{1152}(168\nu_3\nu_5 + 385\nu_3^4 - 630\nu_3^2\nu_4 - 24\nu_6 + 105\nu_4) + \cdots$$

where

$$\nu_s = f^{(s)}(t_N) \Big/ \left(\sqrt{f^{(2)}(t_N)}\right)^s$$

Error Analysis and Complexity. Let us establish the accuracy of the approximation. The main result is summarized in the following lemma:

Lemma 5.3.1. The relative error of the canonical approximation ϵ_N is of $O(1/N)$.

The proof of this lemma can be found in the appendix at the end of this chapter (section 5.9). From this lemma it follows that $\epsilon_N \rightarrow 0$. Therefore, to $O(1/N)$ one has

$$Z_N \approx \frac{N!\exp(C(t_N))}{t_N^{N+1}\sqrt{2\pi f''(t_N)}} \tag{5.3.5}$$

The approximate evaluation of the partition function is reduced to finding a positive root t_N of a simple j_p-th order polynomial equation $f'(t_N) = 0$. This computation can be easily accomplished by such algorithms as the bisection method or the Newton-Raphson method ([Kron83]).

To evaluate the (computational) complexity of computing the saddle point, let us consider the bisection method. It is easy to show from 5.3.2 that $f'(0) = -(N+1) < 0$ and $f'(N+1) > 0$. Hence, the saddle point satisfies $0 < t_N < N+1$. To compute the saddle point up to the 6-th decimal digit, one would need $\log N + \log 10^6 < 20 + \log N$ iterations. For example, if the size of the database is 10^6 items, one needs less than 40 iterations. At the i-th iteration, one needs to evaluate $f'(t_N^{(i)})$ at some new estimate $t_N^{(i)}$ to the saddle point t_N. This evaluation can be performed in $O(j_p)$ arithmetic operations using the Horner's rule ([Kron83]). Therefore, using the bisection algorithm, one can compute the saddle point in $O(j_p \log N)$ arithmetic operations. This is easily accomplished for practically any N and j_p of interest.

Even in the "worst case", when there are N classes of transactions (i.e. $j_p = N$) the evaluation of the saddle-point can be done in $O(N \log N)$ arithmetic

operations†. On the other hand, the computation of Z_N for such case using the exact algorithm would require an $O(N^2)$ operations.

5.4. PERFORMANCE MEASURES

Once the approximate partition function is obtained, one can calculate a number of performance measures. The most important performance measures are the average concurrency E_σ of transactions of class σ and the non-blocking probability B_σ that an arriving transaction of class σ is not blocked. In this section, we show how to compute E_σ without an explicit calculation of Z_N.

The main result of this section is summarized by the following lemma:

Lemma 5.4.1. The average number of concurrent transactions of class σ can be computed as follows

$$E_\sigma \approx \rho_\sigma \frac{t_N^{j_\sigma}}{j_\sigma!}\left[1 + \frac{j_\sigma f^{(3)}(t_N)}{t_N[f^{(2)}(t_N)]^2} - \frac{j_\sigma(j_\sigma - 1)}{2t_N^2 f^{(2)}(t_N)}\right] \tag{5.4.1}$$

To $O(1/N)$ the above expression can be rewritten as

$$E_\sigma \approx \rho_\sigma \frac{t_N^{j_\sigma}}{j_\sigma!} \tag{5.4.2}$$

The proof of this lemma can be found in the Appendix at the end of this chapter. Note that the expression for E_σ does not involve an explicit calculation

† In this problem it is easy to show that $f''(t) > 0$ and increasing for $t \in (0, N+1)$. Thus, the Newton-Raphson method is always convergent and gives a much faster algorithm than the bisection method. For the sake of simplicity we analyzed the complexity using the bisection method

of Z_N. It can be used to compute approximately the average concurrency for a database of any size N. The complexity of computation is independent of N, whereas the accuracy increases with N.

A number of other performance measures can be computed using E_σ. The throughput of class σ transactions is

$$T_\sigma = \mu_\sigma E_\sigma \approx \lambda_\sigma \frac{t_N^{j_\sigma}}{j_\sigma!} \tag{5.4.3}$$

Since there are $\binom{N}{j_\sigma}$ transactions of class σ, the non-blocking probability for class σ transactions is

$$B_\sigma = \frac{T_\sigma}{\binom{N}{j_\sigma}\lambda_\sigma} \approx \frac{t_N^{j_\sigma}}{\binom{N}{j_\sigma} j_\sigma!} \tag{5.4.4}$$

Finally, let us give an interpretation to the saddle-point t_N. If E_σ is the concurrency of class σ, then $j_\sigma E_\sigma$ is the average number of items accessed by transactions of class σ. Let us rewrite equation 5.3.2 as follows

$$t_N + \sum_\sigma j_\sigma \frac{\rho_\sigma}{j_\sigma!} t_N^{j_\sigma} - (N+1) = 0 \tag{5.4.5}$$

From equations 5.4.5 and 5.4.2 we have

$$t_N = N + 1 - \sum_\sigma j_\sigma \frac{\rho_\sigma}{j_\sigma!} t_N^{j_\sigma} \approx N - (j_1 E_1 + \cdots j_p E_p) \tag{5.4.6}$$

But $j_1 E_1 + \cdots j_p E_p$ is the average number of items locked by the concurrent transactions of all classes. Since N is the number of items in the database, it follows

from equation (5.4.6) that the saddle-point t_N corresponds to the average number of idle items in the database.

5.5. EXAMPLES

Example 5.5.1. Let us consider the simple case of $p = 1$ and $j_1 = 1$. (The purpose of this example is to illustrate the degree of precision of canonical approximation, using the saddle-point.) Each transaction locks just one item. One can compute the exact value of $Z_N = \sum \binom{N}{i} \rho^i = (1+\rho)^N$. But let us compare it with the one obtained by canonical approximation.

The grand partition function is $Z_G(t) = \exp(C(t))$ where $C(t) = t + \rho t$. Let

$$f(t) = C(t) - (N+1)\log t = t + \rho t - (N+1)\log t$$

The saddle point t_N can be found from equation $f'(t) = 0$ to be $t_N = \frac{N+1}{(1+\rho)}$. At the saddle point

$$\exp(f(t_N)) = \left(\frac{e(1+\rho)}{N+1}\right)^{N+1} \qquad \text{and} \qquad f''(t_N) = \frac{(1+\rho)^2}{N+1}$$

Using Stirling's formula for $N!$

$$N! \approx \sqrt{2\pi N}\left(\frac{N}{e}\right)^N$$

one gets the following approximation for the partition function

$$Z_N \approx \frac{N! e^{f(t_N)}}{\sqrt{2\pi f''(t_N)}} = \left(\frac{e}{N+1}\right)^{N+1} \frac{N!(1+\rho)^N (N+1)}{\sqrt{2\pi(N+1)}} \approx (1+\rho)^N$$

Hence, $Z_N = (1+\rho)^N$ as expected. The average concurrency

$$E \approx \rho t_N = \frac{\rho(N+1)}{1+\rho} = \frac{\rho N}{1+\rho}\left[1 + \frac{1}{N(1+\rho)}\right] \tag{5.5.1}$$

For this example it is easy to show that the exact expression for the average concurrency is $\rho N/(1+\rho)$. Therefore, the relative error ϵ_N in the calculation of the average concurrency is

$$\epsilon_N = \frac{1}{N(1+\rho)} \mapsto 0 \qquad \text{as} \qquad N \mapsto \infty$$

Since there is a total of N different transactions, the non-blocking probability (equation 3.1.5)

$$B = \frac{E}{N\rho} \approx \frac{1}{1+\rho}$$

The blocking probability can be rewritten

$$1 - B \approx E/N$$

This agrees with the asymptotic results (equation 2.4.4.4) for the non-blocking probability in low traffic reported by S. Lavenberg ([Lave84]) and D. Mitra ([Mitr84]).

Example 5.5.2. Assume now that only one class of j-item transactions is allowed. The grand partition function

$$Z_G(t) = \exp\left(t + \frac{\rho}{j!}t^j\right)$$

The saddle-point t_N satisfies

$$\frac{\rho t_N^j}{(j-1)!} + t_N - (N+1) = 0$$

The average concurrency can be found as follows

$$E \approx \frac{\rho t_N^j}{j!} = \frac{N+1-t_N}{j}$$

Since every transaction locks j items, the maximum possible concurrency in the system is N/j transactions. It is easy from the equation for the saddle-point that $t_N \rightarrow 0$ as $\rho \rightarrow \infty$. Thus, the maximum possible concurrency of N/j is achieved at infinite value of the load.

Since there are $\binom{N}{j}$ possible transactions, the non-blocking probability

$$B = \frac{E}{\binom{N}{j}\rho} \approx \frac{N+1-t_N}{\binom{N}{j}j\rho}$$

The utilization of the database $U = jE$. In Figure 5.5.1 we give the curves for the utilization for different values of j. Following [Mitr84], the unit for measuring total offered traffic is chosen to be $j\binom{N}{j}\rho/N$. This is explained by the fact that $\binom{N}{j}\rho$ measures offered traffic in transactions per second and each transaction involves j/N fraction of the database. Therefore, if the locks were shared, $j\binom{N}{j}\rho/N$ would represent the fraction of the database accessed per unit of time. This explains the name of this unit of traffic, namely databases/second ([Mitr84]). It is interesting to note that the utilization curves run almost in parallel. For the same load, there is less interference among transactions for the smaller value of j. This explains why the utilization values are higher for classes with smaller value of j. The utilization increases very fast with the load, but then becomes almost linear as a function of the load. ∎

Example 5.5.3. Suppose now that only 2 classes are allowed with $j_1 = 1$ and $j_2 = 2$. (Transactions of class 1 lock 1 item and transactions of class 2 lock 2

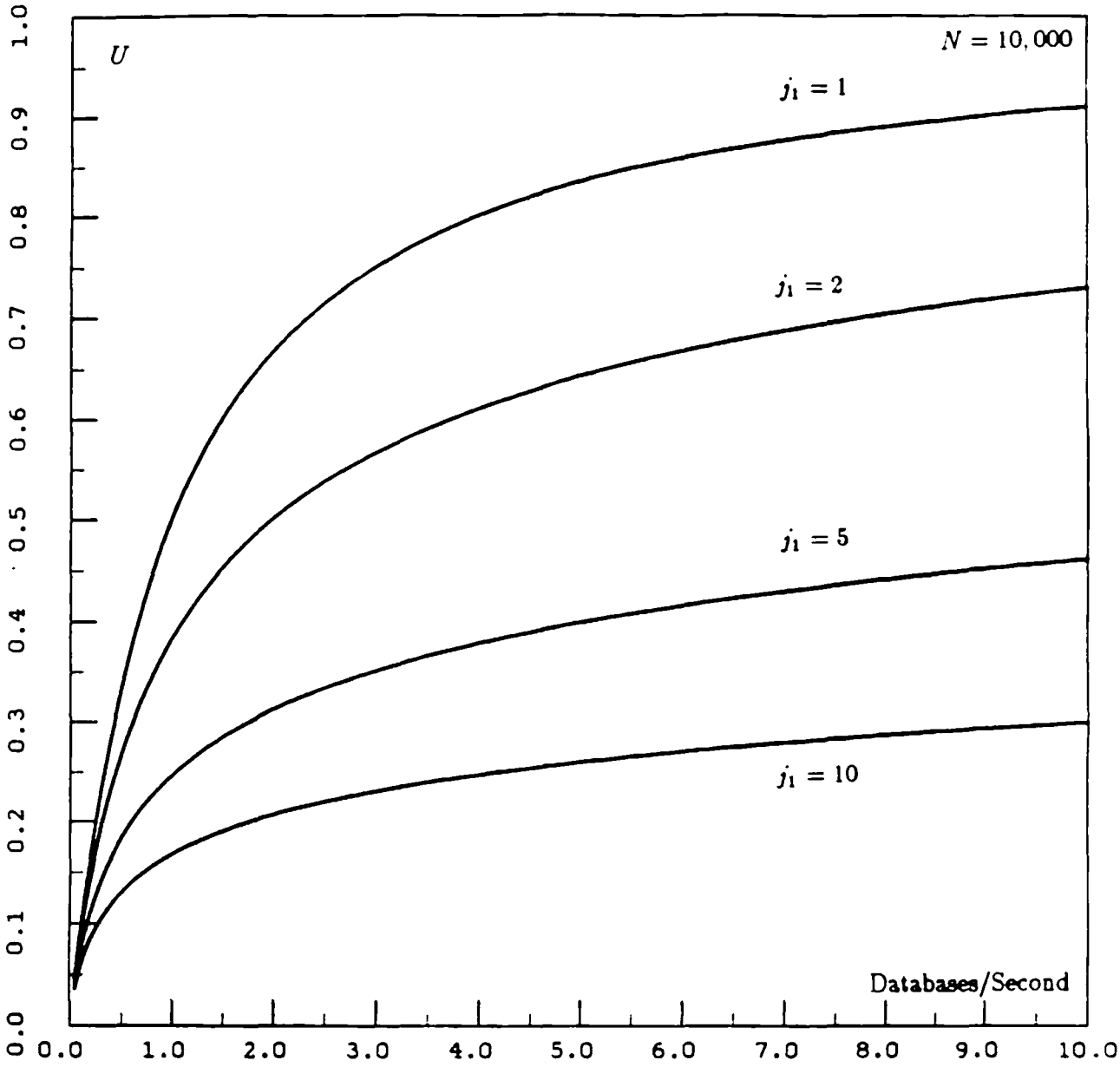

Figure 5.5.1: Database Utilization for Different One-Class Models

items.) The grand partition function

$$Z_G(t) = \exp\left((1+\rho_1)t + \frac{\rho_2}{2!}t^2\right)$$

The saddle point t_N is found by solving:

$$\rho_2 t_N^2 + (1+\rho_1)t_N - (N+1) = 0$$

The positive root gives us the following saddle-point

$$t_N = \frac{\sqrt{(1+\rho_1)^2 + 4\rho_2(N+1)} - (1+\rho_1)}{2\rho_2}$$

The concurrency of class 1 is given by

$$E_1 \approx \rho_1 t_N = \rho_1 \frac{\sqrt{(1+\rho_1)^2 + 4\rho_2(N+1)} - (1+\rho_1)}{2\rho_2}$$

The concurrency of class 2 is given by

$$E_2 \approx \frac{\rho_2}{2!}t_N^2 = \frac{1}{2}[N+1-(1+\rho_1)t_N] \approx \frac{N}{2} - \frac{(1+\rho_1)[\sqrt{(1+\rho_1)^2 + 4\rho_2(N+1)} - (1+\rho_1)]}{4\rho_2}$$

There are N possible transactions of class 1 and $\binom{N}{2}$ possible transactions of class 2. Therefore, the non-blocking probabilities B_1, B_2 for classes 1 and 2 are given by:

$$B_1 = \frac{E_1}{N\rho_1} \approx \frac{t_N}{\rho}$$

and

$$B_2 = \frac{E_2}{\binom{N}{2}\rho_2} \approx \frac{N+1-(1+\rho_1)t_N}{N(N-1)\rho_2}$$

In Figure 5.5.2 we show the curves for the database utilization from these two classes of customers. Obviously, $U_1 = E_1/N$ and $U_2 = 2E_2/N$. We fixed the load of 2-item transactions at 5 databases/second. As we increase the load of 1-item transactions, the utilization of class 2 decreases and the utilization of class 1 increases. This can be explained by the fact that there is more interference experienced by 2-item transactions than by 1-item transactions (These transactions contend over 2 items rather than over 1 item). Thus, with load increase, the fraction of the database filled with 1-item transactions will increase while that with 2-item transactions will decrease. In some sense, class 2-transactions are being "starved". A similar situation was described in [Mitr84]. In fact, for a fixed N and ρ_2 one can easily find the value of ρ_1^c at which the fractions are the same. Increasing ρ_1 beyond this point leads to starvation of class-2 transactions. This is illustrated in Figure 5.5.2.

Example 5.5.4. Assume that $1 \ll p \ll N$ and $j_\sigma = \sigma$, that is class σ transactions require σ items. Assume $\rho_i = \rho$. Note that for this model, the total traffic of class σ transactions is $\binom{N}{\sigma}\rho_\sigma$ increases with σ. The grand partition function is

$$Z_G(t) = \sum_{N=0}^{\infty} \frac{Z_N}{N!} t^N = \exp\left(t + \sum_{l=1}^{p} \frac{\rho}{l!} t^l\right) \approx \exp(t + \rho(e^t - 1))$$

The saddle point t_N satisfies

$$t_N + \rho \exp(t_N) = N + 1$$

For a fixed ρ and large N we have $t_N \approx \log\frac{N}{\rho}$. Therefore, the average concurrency of transactions of class i is

$$E_i \approx \frac{\rho}{i!} t_N^i \approx \frac{\rho}{i!}\left[\log\frac{N}{\rho}\right]^i \qquad (5.5.4.1)$$

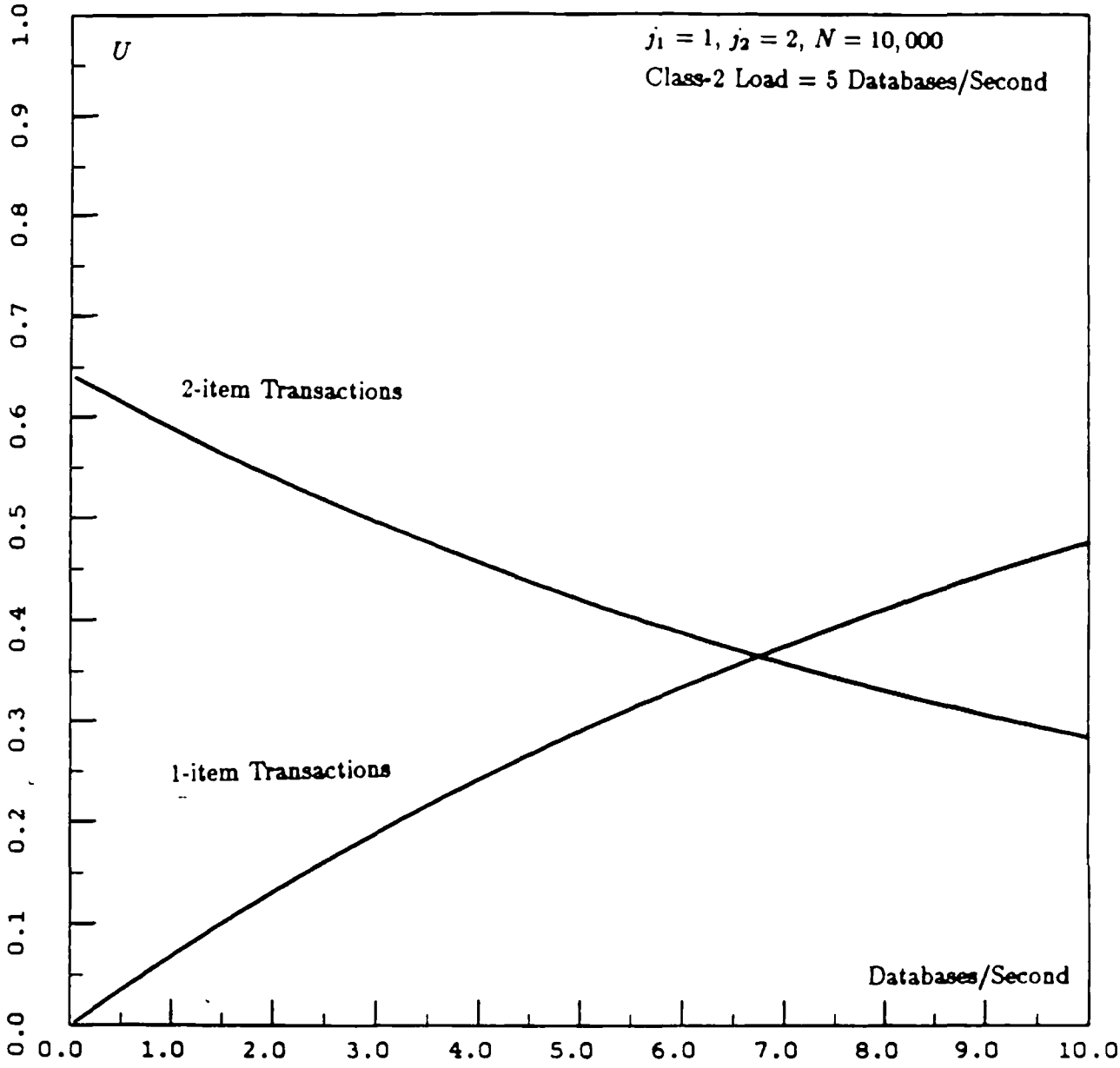

Figure 5.5.2: The "Starvation" Phenomenon

For example, if $\rho = 1/N$ then $E_i \approx 2^i/i!N$. On the other hand, if $\rho \approx 1$ (very high traffic) then $E_N \mapsto (\log N)^i/i!$

Since transactions of different classes lock different number of items in the database, it is more convenient to consider another performance measure: the fraction of the database $U_\sigma = \sigma E_\sigma/N$ accessed by class-σ transactions. Let us first calculate the maximum value of U_σ. To do so, we find the value of ρ at which $\frac{\partial U_\sigma}{\partial \rho} = 0$. Using equation 5.5.4.1 we have

$$\frac{\partial U_\sigma}{\partial \rho} = \frac{1}{\sigma!}\left(\frac{\sigma}{N}\log\frac{N}{\rho} - 1\right)$$

Solving $\frac{\partial U_\sigma}{\partial \rho} = 0$ we obtain $\log\frac{N}{\rho} = N/\sigma$ which implies that $\rho = N\exp(-N/\sigma)$. It is easy to check that at this ρ, the value of U_σ is maximized and is given by

$$\max(U_\sigma) \approx \sigma\left(\frac{N}{\sigma}\right)^\sigma \exp\left(-\frac{N}{\sigma}\right) \qquad \text{at } \rho = N\exp\left(-\frac{N}{\sigma}\right)$$

What does this mean ? For light ρ there is little interference. We would expect that the utilization U_σ increases with ρ. For larger ρ, there is more interference. Moreover, for larger ρ, transactions of lower classes (i.e. classes $1,\ldots,\sigma-1$) may have a higher chance to succeed because they lock a smaller number of items. Because of this, for higher values of ρ (i.e. for $\rho > N\exp(-N/\sigma)$), the utilization U_σ would, in fact, be decreasing with an increase in ρ. This is illustrated in Figure 5.5.3, where we show the curves of U_5 and U_6 for a database of $N = 1000$ items and $p = 20$ classes. Note that the curves intersect, which suggests that above some ρ we can expect $U_{\sigma+1} < U_\sigma$.

Let us establish this formally. We want to find the value of the load ρ such that $U_{\sigma+1} = U_\sigma$. We have

$$U_{\sigma+1} = \frac{\sigma E_\sigma}{N} \approx \frac{\rho}{\sigma!}t_N^{\sigma+1} = \frac{\rho}{(\sigma-1)!}t_N^\sigma\frac{t_N}{\sigma} \approx \frac{t_N}{\sigma}U_\sigma$$

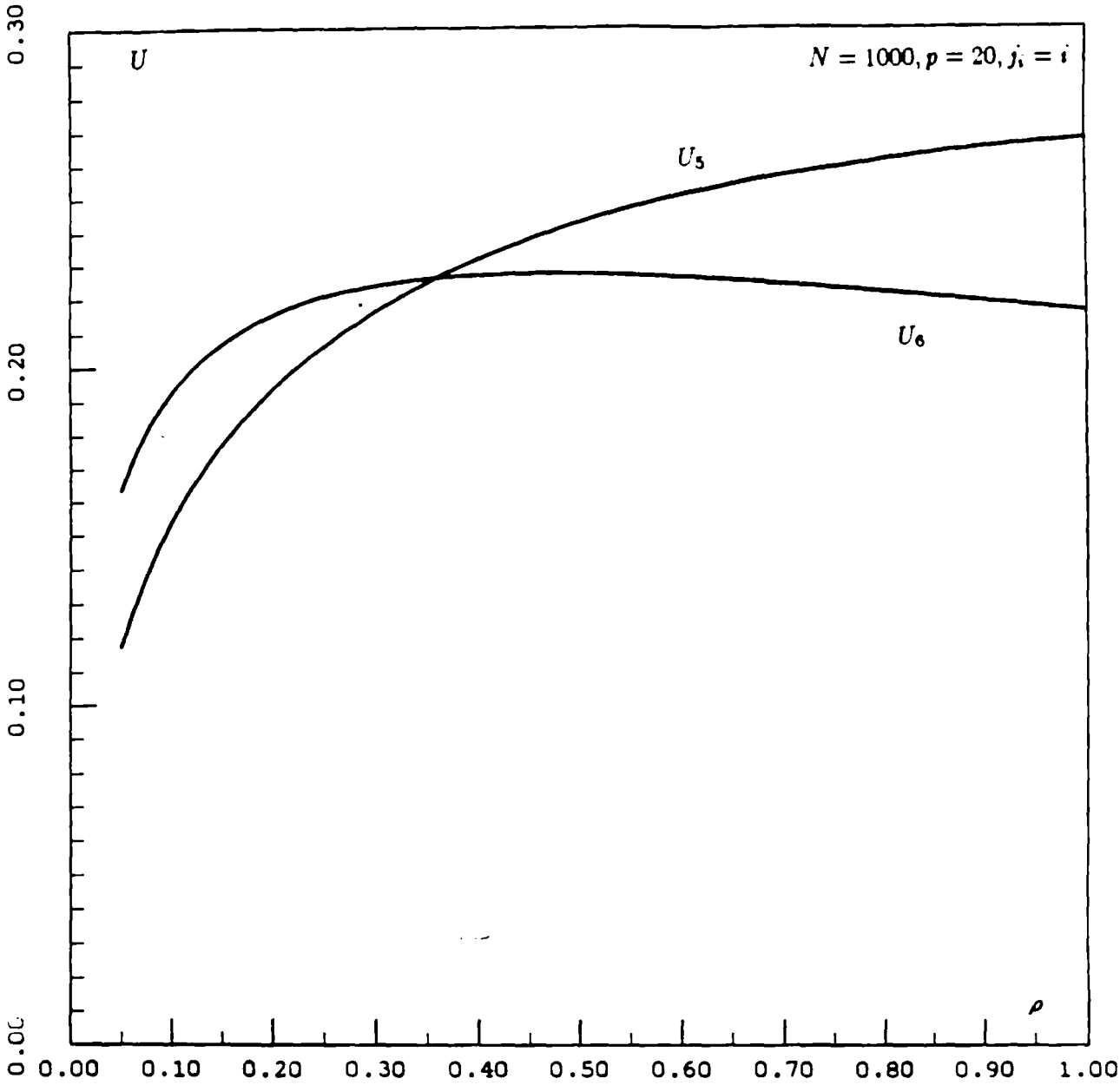

Figure 5.5.3: Database Utilization for Multi-Class Model

Therefore, $U_{\sigma+1} = U_\sigma$ for $t_N = \sigma$ which implies that $\rho = Ne^{-\sigma}$. For $\rho > Ne^{-\sigma}$ (i.e. $t_N < \sigma$) we have $U_{\sigma+1} < U_\sigma$. This can be easily explained. For the same ρ, the total traffic of class $\sigma + 1$ is $\binom{N}{\sigma+1}\rho$ and is higher than that of class σ, which is $\binom{N}{\sigma}\rho$. For light traffic (small ρ), there is little interference and most of the transactions are able to proceed. Therefore, for small ρ one would expect $U_{\sigma+1} > U_\sigma$.

Transactions of class $\sigma + 1$ lock more items than transactions of class σ. Therefore, one would expect that the fraction of the database accessed by transactions of class $\sigma + 1$ would be higher than that of class σ. As one increases the load, there is more interference. Transactions requiring smaller number of items have more chance to proceed, even though the total traffic is higher for larger classes. Thus U_σ decreases for larger σ and increases for smaller σ. This is exactly similar to the phenomena of "starvation" described in the previous example. For each σ, canonical approximation allows us to calculate the value of the load (namely, $\rho = Ne^{-\sigma}$) beyond which $U_\sigma > U_{\sigma+1}, \ldots U_p$. As we keep on increasing ρ, the transactions of more and more classes are be "starved". In the limit (as $\rho \mapsto \infty$) only class-1 transactions would be present. In Figure 5.5.3 we showed the curves

The non-blocking probability for class σ is given by

$$B_\sigma = \frac{E_\sigma}{\binom{N}{\sigma}\rho} \approx \frac{(N-\sigma)!}{N!}\left[\log\frac{N}{\rho}\right]^\sigma$$

It is easy to see that B_σ decreases to 0 quite rapidly with σ. This reflects the fact that for larger σ, an arriving transaction will find it very unlikely that the required items are available.

5.6. CONDENSATION OF THE DATABASE

In this section we will analyze the behavior of the database under heavy traffic. We will show the existence of a set of threshold parameters for the loads, above which, the average number of transactions, requiring many items, becomes insignificant (e.g. $\ll 1$). We will show that this is similar to liquid-gas condensation of an imperfect gas. Thus, one can think of this change in the behavior of a database as the database "condensation".

We start by analyzing the dependence of the saddle point t_N on the load parameters $\rho_1, \ldots, \rho_p$. In particular, we will show that for any $r \in (0, N+1)$ we can choose the loads so that $t_N < r$. To start, let us examine the equation for the saddle point t_N (derived from equation 5.3.2):

$$f'(t) = t + \frac{\rho_1}{(j_1 - 1)!} t^{j_1} + \cdots + \frac{\rho_p}{(j_p - 1)!} t^{j_p} - (N+1) = 0 \tag{5.6.1}$$

When all the ρ_σ's are 0, the saddle point is obviously $t_N = N + 1$. Let us show that if we increase any of the loads, say ρ_σ, the saddle-point decreases. Since t_N is a root of $f'(t) = 0$ we have by the implicit function theorem

$$\frac{\partial t_N}{\partial \rho_\sigma} = -\frac{\frac{\partial f'(t_N)}{\partial \rho_\sigma}}{\frac{\partial f'(t_N)}{\partial t_N}} = -\frac{t_N^{j_\sigma - 1}}{(j_\sigma - 1)! f''(t_N)} \tag{5.6.2}$$

Since $t_N > 0$ and $f''(t_N) > 0$ (equation 5.3.3), it follows from equation 5.6.2 that $\frac{\partial t_N}{\partial \rho_\sigma} < 0$. This means that the saddle point decreases with an increase in ρ_σ.

Let us show that by choosing the loads $\rho_1, \ldots, \rho_p$ sufficiently large we can make $t_N < r$ for any $r \in (0, N+1)$. It would be sufficient to show that we can choose the load of any class, say ρ_σ independently of the others so that $t_N < r$.

Indeed, take ρ_σ so that $[(N+1)(j_\sigma-1)!/\rho_\sigma]^{1/j_\sigma} = r$. In other words, r is the root of the equation $\frac{\rho_\sigma}{(j_\sigma-1)!}t^{j_\sigma} - (N+1) = 0$. But the coefficients of $f'(t_N) = 0$ (except for $t_N^{j_\sigma}$ and $-(N+1)$) are larger than those for $\frac{\rho_\sigma}{(j_\sigma-1)!}t^{j_\sigma} - (N+1) = 0$. It follows then that for such ρ_σ (and any choice of the other loads) $t_N < r$. This is illustrated in Figure 5.6.1, where we show the (typical) dependence of t_N on the loads†.

Therefore, it is possible to choose the load parameters $\rho_1, \ldots, \rho_p$ (not in a unique way, of course) so that the saddle point $t_N < r$ for any $r \in (0, N+1)$. "Smaller" values of ρ_σ's correspond to "larger" values of t_N and, conversely, "larger" values of ρ_σ's correspond to "smaller" values of t_N. This is consistent with the interpretation of the saddle point as the average number of unlocked items in the database (see equation 5.4.6): for smaller loads, the number of unlocked items is larger, for larger loads, it is smaller.

We now turn to the primary focus of this section. In the previous section we mentioned for some specific examples that for large loads, the transactions requiring many items have a lesser chance to succeed than those requiring fewer items. We now ask the following question: when will the average number of transactions, requiring many items, be significant (e.g. $\gg 1$)? To answer this, let us analyze E_l for large l. From equation 5.4.2 and Stirling's approximation to $l!$ we have

$$E_l \approx \frac{\rho_l}{l!}t_N^l \approx \frac{\rho_l}{\sqrt{2\pi l}}\left(\frac{e}{l}t_N\right)^l \tag{5.6.3}$$

From the previous discussion it follows that we can choose the loads ρ_σ's so that $t_N < \frac{l}{e}$. From equation 5.6.1 we obtain that for such loads, the number E_l is practically equal to zero. It is so, since $et_N/l < 1$ and $l \gg 1$. This means that in a macroscopic sense, the population of large transactions is not at all significant. On the other hand, if $t_N > \frac{l}{e}$, then E_l becomes large and thus large transactions would be present.

---

† To show the dependence of t_N on all the parameters $\rho_1, \ldots, \rho_p$ would require $p+1$ dimensions. Thus, we showed the dependence of t_N on just two load parameters ρ_i and ρ_j.

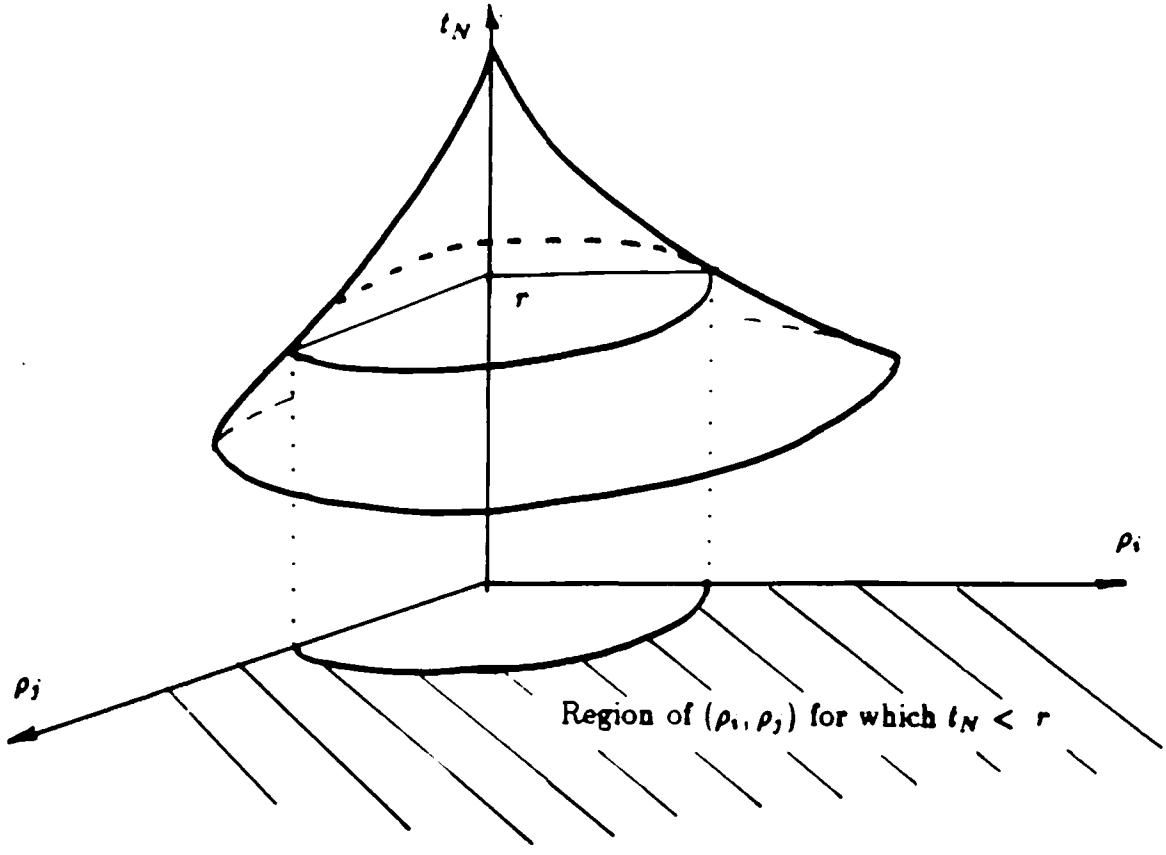

Figure 5.6.1: Saddle Point as a Function of Loads

What does this mean? When ρ_l are "large" (i.e. such that $t_N < l/e$), as soon as a transaction of class σ departs and frees j_σ items, it is likely that there will be a number of transactions competing for them. Transactions of lower classes will have a better chance to proceed, since they contend over a smaller set of items. Therefore, one would not expect large ($l \gg 1$) transactions to be present in macroscopically significant numbers. On the other hand, if ρ_l are small, then these transactions can be present since there may be very little interference. Therefore, if one starts with high values of ρ_l's, then as the ρ_l's decrease (the database "cools") we reach a point when a formation of "large" transactions takes place. Note that this point is achieved for still relatively high ρ_l.

Let us explain this phenomenon using the analogy to imperfect gas ([Path84]). A real (imperfect) gas consists basically of discrete molecules. To a fairly good approximation, the potential energy ϕ of the assembly of N molecules can be broken into pair-wise interactions. The "strength" of pairwise interaction between molecules i and j is specified by the function $f_{i,j}$. One assumes $f_{i,j} = 0$ if there is no interaction between molecules i and j. The higher the temperature of imperfect gas, the stronger is the pairwise interaction. A molecule in such a physical system corresponds to an item in a database. Two interacting molecules correspond to two items in the same transaction. To make the analogy complete, let us say that the "strength" of pairwise interaction between two items in a database is determined by probability that these items are locked by the same transaction. The higher the load on the database, the more likely it is for any two items to be in the same transaction (i.e. the stronger is the pairwise interaction). In view of the above discussion, it is more likely that these two items will be in a smaller transaction than in a larger one.

Let us make an assumption that class σ transactions lock σ items and classes run from 1 to N. The reason for introducing these assumptions is to make the formulas "identical" to those obtained in statistical physics. The statistical mechanical interpretation of the results is valid for the general case as well.

One can associate a graph of N nodes in 1-1 correspondence with interacting molecules: there is an edge between points i and j whenever $f_{i,j} \neq 0$. The collection of graphs that link up any l points will be called an l-cluster. Thus, a transaction locking l items corresponds to an l-cluster. Figure 5.6.1. shows the 4 graphs corresponding to a 3-cluster. A 3-cluster corresponds to an active transaction locking 3 items. With such an l-cluster, one defines the so-called l-cluster integrals b_l as follows†

$$b_l = \frac{1}{V l!} \int (\text{contribution from an } l\text{-cluster}) \tag{5.6.4}$$

The partition function Z_N can be evaluated in terms of these l-cluster integrals b_l as follows ([Path84])

$$Z_N = \sum_{m_l \geq 0}^{N\ '} \left[\prod_{l=1}^{N} \frac{(V b_l)^{m_l}}{m_l!} \right] \qquad \text{with} \qquad \sum_{l=1}^{N} l m_l = N \tag{5.6.5}$$

For the database system, the contribution of the l-item transaction to the partition function depends on ρ_l. In fact, if one defines $b_l = \rho_l / V l!$, the above expression for the partition function of an imperfect gas of N molecules with pairwise interactions can be written in the same form as partition function for a database‡. The partition function is computed from its exponential grand partition function

$$Z_G(t) = \exp(V b_1 t + \cdots + V b_N t^N) \tag{5.6.6}$$

---

† There is also a factor of $1/\lambda^{3(l-1)}$, where λ denotes the thermal de Broglie wavelength of the particles. One assumes $\lambda = 1$ for the sake of the simplicity of notation.

‡ It is assumed that every molecule is a part of some cluster. This is not exactly true for our model of the database since some of the items may remain unlocked by any transaction. However, this difference is insignificant in this context.

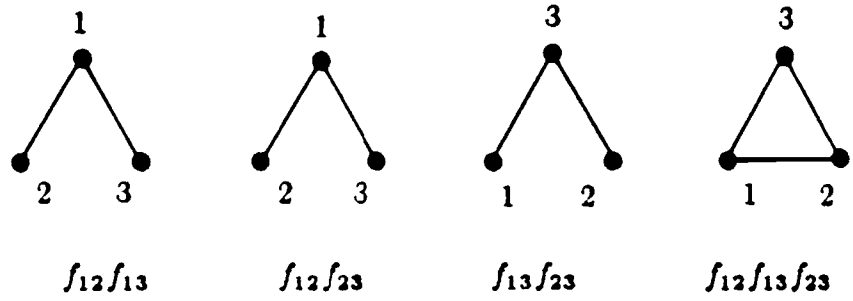

Figure 5.6.2: A 3-cluster and the Corresponding Product Terms

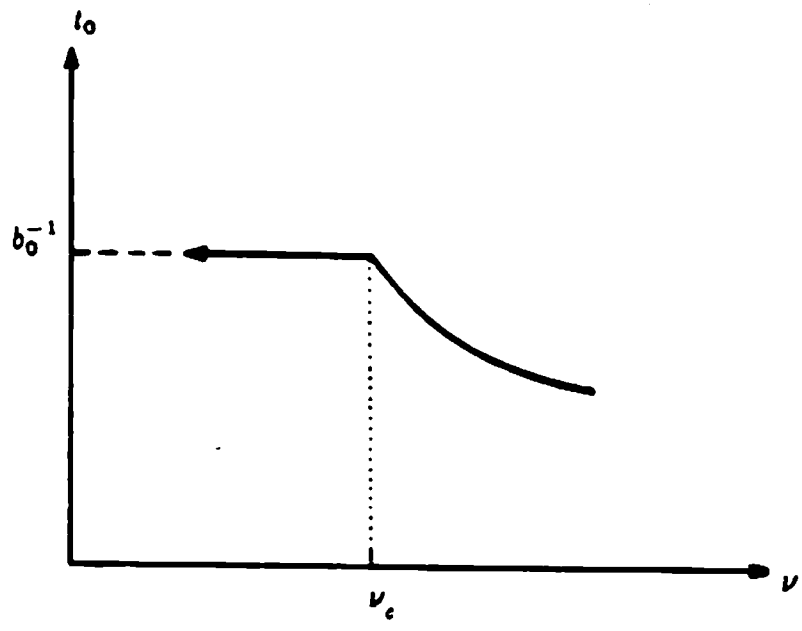

Figure 5.6.3: t_0 and ν in the Region of a Gas-Liquid Transition

using the saddle-point method ([Path84]). The expression 5.6.6. for the grand partition function for imperfect gas is "similar" to the expression 5.2.2. for the grand partition function for the database model. Let m_l^* denote the average value of m_l. It can be shown ([Path84]) from equation 5.6.4 that

$$m_l^* = V b_l t_N^l \tag{5.6.7}$$

where t_N is the saddle-point used to calculate the partition function Z_N (equation 5.6.5) from the grand partition function $Z_G(t)$ (equation 5.6.6). Using $b_l = \rho_l / V l!$, we can rewrite this as $m_l^* = \rho_l t_N^l / l!$ in complete analogy with the expression for the average concurrency of transactions of class l (equation 5.4.2).

Mayer ([Maye38] has shown that for this system of interacting particles, one has $b_l = k_l b_0^l$ for large l. Here b_0 is a function of the absolute temperature calculated from cluster integrals. From 5.6.5, one obtains that the population of large-sized clusters $m_l^* \approx V k_l (b_0 t_N)^l$†. As long as $t_N < b_0^{-1}$, the population m_l^* of l-clusters is practically 0 since $b_l t_N < 1$ and $l \gg 1$. This means that in the *macroscopic* sense, the population of large-sized clusters is not significant. In a database system, this corresponds to high values of the loads ρ_l so that the number of large transactions $E_l \approx 0$.

On the other hand, if $t_N > b_0^{-1}$ then m_l^* becomes exceedingly large. This means that as t_N increases and passes through the critical value $t_c = b_0^{-1}$, a formation of large-sized clusters (i.e. condensation) takes place. In a database system this corresponds to the "low" values of ρ_l so that $E_l \gg 1$.

Figure 5.6.2. shows the relation between the specific volume $\nu = V/N$ in the condensation region for imperfect gas. The horizontal portion in the figure represents a preferential formation of large-sized clusters, and hence the appearance of a two-phase state, in the system. If t_N is increased (even slightly) above the

† Here k_l is also a function of the variable l. However, this dependence does not play any significant role in this context.

critical value b_0^{-1}, the density $1/\nu$ becomes exceedingly large: there is a formation of large-size clusters. Similarly, if t_N is decreased (even slightly) below the critical value of b_0^{-1}, the density becomes small: there are almost no large-size clusters.

Thus, one can think of the corresponding phenomenon in the behavior of the database as that of a "database condensation". It means that there are values ρ_l^* of the loads so that for $\rho_l > \rho_l^*$ the throughput of transactions of class l decreases and becomes almost 0. The loads should be tuned as to not allow this to happen. Canonical approximation allows one to compute analytically the corresponding "threshold" loads for each class of transactions.

5.7. CONCLUSION

In this chapter the method of canonical approximation was applied to study the exact locking models of database systems. Closed form expressions were obtained for average concurrency, non-blocking probability, and other database performance measures. The computations are reduced to finding a positive root of a simple polynomial equation. Performance measures are expressed in terms of this root without an explicit calculation of the partition function. They can be obtained for any values of parameters. Using these closed form expressions, we established the existence of of "database condensation", which was explained using the analogy from the theory of imperfect gas. Further work will extend the applications of the method to other locking models of database systems.

5.8. APPENDIX

In this appendix we present the proof of lemma 5.3.1 and lemma 5.4.1 used in the chapter.

Lemma 5.3.1. ϵ_N is of $O(1/N)$.

Proof. By theorem 3.2.1, it is enough to show that the dominant term of ϵ_N are of $O(1/N)$. Let us rewrite this term

$$\epsilon_N = \frac{1}{24}(3\nu_4 - 5\nu_3^2) + \frac{1}{1152}(168\nu_3\nu_5 + 385\nu_3^4 - 630\nu_3^2\nu_4 - 24\nu_6 + 105\nu_4) \quad (5.3.1)$$

The derivatives of $f(t)$ can be found as follows

$$f^{(s)}(t_N) = C^{(s)}(t_N) + (-1)^s \frac{(s-1)!(N+1)}{t_N^s} \quad (5.3.2)$$

Hence,

$$\nu_s = \frac{t_N^s C^{(s)}(t_N) + (-1)^s (s-1)!(N+1)}{\left[t_N^2 C^{(2)}(t_N) + (N+1)\right]^{s/2}} \quad (5.3.3)$$

It is enough to give an estimate of ν_s for $s \leq 6$. First, let us note the following

$$t_N^2 C^{(2)}(t_N) = \sum_\sigma j_\sigma(j_\sigma - 1)\rho_\sigma \frac{t_N^{j_\sigma}}{j_\sigma!} \geq \sum_\sigma j_\sigma \rho_\sigma \frac{t_N^{j_\sigma}}{j_\sigma!} = t_N C'(t_N) = N + 1$$

Therefore,

$$t_N^s C^{(s)}(t_N) = \sum_\sigma j_\sigma \cdots (j_\sigma - s + 1)\rho_\sigma \frac{t_N^{j_\sigma}}{j_\sigma!}$$

$$\leq (j_p - 1)\cdots(j_p - s + 1) t_N C'(t_N)$$

$$= (j_p - 1)\cdots(j_p - s + 1)(N+1) \ll j_p^{s-1}(N+1)$$

Therefore, we obtain the following (pessimistic !!!) upper bound

$$\nu_s \ll \frac{j_p^{s-1} + (-1)^s(s-1)!}{2^{s/2}(N+1)^{s/2-1}} \tag{5.3.4}$$

On the other hand, since

$$t_N^2 \dot{C}^{(2)}(t_N) \leq (j_p - 1) t_N C'(t_N) = (j_p - 1)(N+1)$$

we obtain the following lower bound

$$\begin{aligned} \nu_s &\geq \frac{t_N^s C^{(s)}(t_N) + (-1)^s(s-1)!(N+1)}{j_p^{s/2}(N+1)^{s/2}} \\ &> \frac{(-1)^s(s-1)!(N+1)}{j_p^{s/2}(N+1)^{s/2}} \mapsto 0 \end{aligned} \tag{5.3.5}$$

Substituting these bounds from 5.3.4 and 5.3.5 into the expression 5.3.1 of the most dominant term of relative error, one can conclude that ϵ_N is of $O(1/N)$. ∎

Let us now prove the second lemma

Lemma 5.4.1. The average number of concurrent transactions of class σ can be computed as follows

$$E_\sigma = \rho_\sigma \frac{t_N^{j_\sigma}}{j_\sigma!} \left[1 + \frac{j_\sigma f^{(3)}(t_N)}{t_N [f^{(2)}(t_N)]^2} - \frac{j_\sigma (j_\sigma - 1)}{2 t_N^2 f^{(2)}(t_N)} \right] \tag{5.4.1}$$

To $O(1/N)$ we have

$$E_\sigma \approx \rho_\sigma \frac{t_N^{j_\sigma}}{j_\sigma!} \tag{5.4.2}$$

Proof. From 5.2.1 it follows that

$$E_\sigma = \rho_\sigma \frac{\partial}{\partial \rho_\sigma} \log Z_N = \frac{\rho_\sigma}{Z_N} \frac{\partial Z_N}{\partial \rho_\sigma} \tag{5.4.3}$$

By the Chain rule

$$\frac{\partial Z_N}{\partial \rho_\sigma} = \frac{\partial^* Z_N}{\partial \rho_\sigma} + \frac{\partial Z_N}{\partial t_N} \frac{\partial t_N}{\partial \rho_\sigma} \tag{5.4.4}$$

The partial derivative $\frac{\partial^* Z_N}{\partial \rho_\sigma}$ is taken explicitly with respect to ρ_σ, treating t_N as an independent variable. The first term of 5.4.4 gives

$$\begin{aligned} \frac{\partial^* Z_N}{\partial \rho_\sigma} &\approx \frac{N! \exp(C(t_N))}{\sqrt{2\pi} t_N^{N+1} f^{(2)}(t_N)} \left[\frac{t_N^{j_\sigma}}{j_\sigma!} \sqrt{f^{(2)}(t_N)} - \frac{t_N^{j_\sigma - 2}}{2(j_\sigma - 2)! \sqrt{f^{(2)}(t_N)}} \right] \\ &= \frac{t_N^{j_\sigma}}{j_\sigma!} Z_N - \frac{t_N^{j_\sigma - 2}}{2(j_\sigma - 2)! f^{(2)}(t_N)} Z_N \end{aligned} \tag{5.4.5}$$

For the second term of 5.4.4 we have

$$\frac{\partial Z_N}{\partial t_N} \frac{\partial t_N}{\partial \rho_\sigma} = - \frac{\exp(C(t_N)) f^{(3)}(t_N)}{t_N^{N+1} 2 f^{(2)}(t_N) \sqrt{2\pi f^{(2)}(t_N)}} \frac{\partial t_N}{\partial \rho_\sigma} = - \frac{f^{(3)}(t_N)}{f^{(2)}(t_N)} Z_N \frac{\partial t_N}{\partial \rho_\sigma} \tag{5.4.6}$$

Since t_N satisfies $f'(t_N) = 0$, we have by the implicit function theorem

$$\frac{\partial t_N}{\partial \rho_\sigma} = - \frac{\frac{\partial f'(t_N)}{\partial \rho_\sigma}}{\frac{\partial f'(t_N)}{\partial t_N}} = - \frac{t_N^{j_\sigma - 1}}{(j_\sigma - 1)! f^{(2)}(t_N)} \tag{5.4.7}$$

Therefore, equation 5.4.3 can be rewritten as follows

$$\frac{\partial Z_N}{\partial \rho_\sigma} \approx Z_N \frac{t_N^{j_\sigma}}{j_\sigma!} \left[1 + \frac{j_\sigma f^{(3)}(t_N)}{t_N [f^{(2)}(t_N)]^2} - \frac{j_\sigma (j_\sigma - 1)}{2 t_N^2 f^{(2)}(t_N)} \right] \tag{5.4.8}$$

Equations 5.4.3 and 5.4.8. imply 5.4.1. To show equation 5.4.2 let us rewrite equation 5.4.1 as follows:

$$E_\sigma = \rho_\sigma \frac{t_N^{j_\sigma}}{j_\sigma!}[1+\epsilon] \tag{5.4.9}$$

where ϵ can be written as follows (from equation 5.4.8)

$$\epsilon = \frac{j_\sigma[t_N^3 C^{(3)}(t_N) - 2(N+1)]}{[t_N^2 C^{(2)}(t_N) + (N+1)]^2} - \frac{j_\sigma(j_\sigma - 1)}{t_N^2 C^{(2)}(t_N) + (N+1)} \tag{5.4.10}$$

Since $j_\sigma \leq j_p$ we have

$$t_N^3 C^{(3)}(t_N) \ll (j_p - 1)(j_p - 2) t_N C'(t_N) = (j_p - 1)(j_p - 2)(N+1) \tag{5.4.11}$$

Similarly,

$$t_N^2 C^{(2)}(t_N) \gg \frac{t_N C'(t_N)}{j_p} = \frac{N+1}{j_p} \tag{5.4.12}$$

Therefore, from equations 5.4.10, 5.4.11 and 5.4.12 we obtain

$$\epsilon \ll \frac{j_\sigma(j_p^2 - 3j_p)}{(N+1)(1+1/j_p)^2} - \frac{j_p(j_p-1)}{(N+1)(1+1/j_p)} \tag{5.4.13}$$

The error term is therefore of $O(1/N)$. Hence, to $O(1/N)$ the average concurrency of class σ is given by equation 5.4.2. ∎

CHAPTER 6

CANONICAL APPROXIMATION: CLOSED MARKOVIAN NETWORKS

6.1. INTRODUCTION

This chapter applies the canonical approximation method to analyze single-class closed queueing networks. A network of this class consists of N customers circulating among M nodes. Each node may consist of a single constant rate exponential server (a load-independent node), m such exponential servers (an m-server node) or infinite number of such exponential servers (an infinite server or IS node). The problem is to compute a number of performance measures for each node. Among the key performance measures are average queue length, average waiting time, throughput and utilization.

Canonical approximation gives simple closed-form expressions for these performance measures. The new algorithms, based on these expressions, are stable, give excellent precision (even for small N and M) and require $O(M)$ arithmetic operations for for both load-dependent and load-independent cases. They require a fixed size storage for intermediate computations. The computation of the performance measures does not require the explicit computation of the normalization constant Z_N of the steady-state probability distribution. This makes it possible to analyze networks for any N. The new algorithms are very simple and can be implemented by a Pascal program of less than 100 lines of code. The asymptotic analysis ($N \mapsto \infty$) for the case, where all the nodes are load-independent, requires negligible computation and can be implemented on a pocket calculator. This is unlike most of the other exact or approximate methods ([Buze73, Chan82, Eager84, Schw79]), whose computational complexity is usually at least of $O(NM)$ and whose storage requirements for intermediate computations is usually of $O(N)$. On the other hand,

let us note that many of the algorithms for single-class queueing networks have been extended to analyze multi-class queueing networks, where the service distribution of a customer may depend on his class. We do not yet know how to apply canonical approximation to analyze multi-class queueing networks (see Chapter 8). In section 6.7 we will compare our algorithms with some well-known computational algorithms for closed Markovian networks with regard to precision, computational complexity, and storage requirements.

The importance of fast algorithms for the performance analysis of queueing network models of computer systems is well known ([Buze71, Kell80, Klei76]). As noted by McKenna ([McKe84]), "closed Markovian queueing networks have emerged as one of the most important tools for modeling computer systems, computer communication systems, on-line computer networks, and other real time computer systems ([Klei76, Saue81]). This was because of the discovery of an important class of such networks that were analytically tractable, the so-called product form networks ([Bask75, Kell80]). Unfortunately, until quite recently, the use of these models was confined to relatively small networks, since their use in large networks involves intractably large calculations or approximations involving errors of unknown magnitude, questions of uniqueness, unknown range of applicability, and problems with convergence ([Bard79, Buze73, Chan82, Eage84, Schw79])." The book by Bruell and Balbo ([Brue84]) presents an excellent overview of different algorithms for closed queueing networks.

The computational expense of an exact solution of a product form queueing network can be prohibitive when the network has multiple closed customer classes. As noted by Eager ([Eage84]), there are two major motivations for considering fast approximate solution techniques for single-class queueing networks. First, in utilizing some of the approximate solution techniques ([Zaho80]), it may be necessary to analyze a very large number of single-class networks to obtain some bounds for the analysis of multiple-class networks. Secondly, the complexity of the existing exact and approximate algorithms typically increases with N and M. Thus, these

algorithms may not be appropriate for very large networks. Moreover, some of these algorithms are unstable even for moderate values of N and M (see section 6.8). Finally, fast algorithms for single-class networks are of theoretical interest, since they often provide insight that can be applied to the multiple closed queueing networks.

6.2. THE MODEL

Consider a closed network of M nodes with N jobs (customers). Let $\mu_i(k)$ be the expected service time at node i when there are k jobs at that node. For the load-independent case, $\mu_i(k)$ is independent of k, whereas for the load-dependent case, $\mu_i(k)$ depends on the number of jobs at node i. Let p_{ij} be the routing probability that when a job finishes at node i, it will seek service at node j. It is well-known that the steady-state probability distribution that there are n_i customers at node i is given by the Gordon-Newell solution ([Gord67])

$$P(n_1, \ldots, n_M) = \frac{1}{Z_N} \prod_{i=1}^{M} \frac{X_i^{n_i}}{D_i(n_i)} \tag{6.2.1}$$

where:

(1) $X_i = e_i \mu_i$. The e_i are the solutions of M linear equations

$$e_i = \sum_{j=1}^{M} e_j p_{ji}$$

which equate the flows into and out of each station. Note that the above product form solution is unique up to a multiplicative constant. (This is because there are

$M-1$ independent equations in determining the e_i's.) If one sets $e_1 = 1$, the other e_i's can be uniquely determined. This can be done by the standard techniques of solving systems of linear equations ([Kron83]), the computational complexity of which are of $O(M^3)$.

Let us note that in analyzing the time complexity and storage requirements of any algorithm for the above models, one does not count the time to compute e_i's and the storage for X_i.

(2)

$$D_i(n_i) = \begin{cases} 1 & n_i = 0 \\ \prod_{j=1}^{n_i} d_j & n_i > 0 \end{cases}$$

Here d_j are positive functions for queue dependent mean service times. For the load-independent case, $d_j = 1$. For the load-dependent node with an infinite server, $d_j = j$. In other words, μ_i/d_j is the service time at node i when there are j customers at that node.

(3) Z_N is the partition function given by

$$Z_N = \sum_S \prod_{i=1}^{M} \frac{X_i^{n_i}}{D_i(n_i)} \tag{6.2.2}$$

Here the set S of possible state configurations is obviously

$$S = \{(n_1, \ldots, n_M): \quad n_1 + \cdots + n_M = N\} \tag{6.2.3}$$

6.3. DETERMINATION OF THE GRAND PARTITION FUNCTION

To apply canonical approximation to evaluate Z_N, we need to compute the grand partition function

$$Z_G(t) = \sum_{N=0}^{\infty} Z_N t^N \tag{6.3.1}$$

To do this, we define A_i to be the random variable over the state-space S (equation 6.2.3) denoting the total number of customers at node i. Let the probability distribution of A_i be given by $F_i(A_i)$. Let $g_i(t)$ be the generating function for the probability distribution of A_i, namely

$$g_i(t) = \sum_{n_i=0}^{\infty} F_i(n_i) t^{n_i} \tag{6.3.2}$$

Then it can be shown ([Thom82, Will76]) that the grand partition function is

$$Z_G(t) = \sum_{i=0}^{\infty} Z_N t^N = \prod_{i=1}^{M} g_i(t) \tag{6.3.3}$$

The above expression (equation 6.3.3) for the grand partition function $Z_G(t)$, stated in terms of the product of generating functions $g_i(t)$, allows one to replace any number of nodes by a composite equivalent node (aggregation). This construction is given in detail in ([Thom82]).

Before applying the method of canonical approximation, let us note here that the computation of Z_N from its generating function $Z_G(t)$ has been considered before. A number of researchers ([Will76, Thom82]) have used the generating function formulation for multiclass closed queueing networks to derive the "convolution" formulae. Although they considered a more general multi-class model, they did not give a direct formula for performance measures which does not involve iterative numerical computation of Z_N even for a single class model. Moore ([Moor72]) obtained an algorithm to compute Z_N for the load-independent case. It requires an $O(M)$ arithmetic operations and uses the partial fraction expansion of $Z_G(t)$ in the simple case when all of the poles are distinct. Lam ([Lam77]) has extended the computation of Z_N for the load-independent case when not all of the poles of $Z_G(t)$ are distinct. Both of these only apply to load-independent networks. Moreover, both of these have poor numerical properties. The explicit calculation of the partition function and of the performance measures in these algorithms involves the summation of the terms of $O(N!)$ (constituting the partition function). In contrast, we will apply canonical approximation to obtain simple and stable algorithms to analyze any single-class queueing networks (of any size) with high precision.

6.4. CANONICAL APPROXIMATION

Let μ_i/d_j denote the expected service time at processor i when there are j jobs at that node (equation 6.2.1). Define

$$Y_{i,j} = \prod_{k=1}^{j} \frac{X_i}{d_j} \tag{6.4.1}$$

By Jackson's theorem ([Klei75a]), the probability distribution of A_i, the number of customers at node i, is that of a Markovian queue with the same arrival

and service time (as for node i in the closed queueing network). Therefore, the generating function for A_i becomes

$$g_i(t) = 1 + Y_{i1}t + Y_{i2}t^2 + \cdots \tag{6.4.2}$$

Of practical interest are the case of a load-independent server, an infinite server (IS) station, and an m-server station.

Load-Independent Server. In this case, the service rate at each node is independent of the number of customers in the queue. Thus, $D_i(n_i) = 1$. The generating function $g_i(t)$ of the probability distribution of A_i is therefore

$$g_i(t) = \sum_{k=0}^{\infty} P(A_i = k)t^k = \sum_{k=0}^{\infty} X^k t^k = \frac{1}{1 - X_i t} \tag{6.4.3}$$

Infinite Server Station. In this case, node i consists of an infinite number of identical servers with rate μ_i. For such a node one has

$$g_i(t) = 1 + X_i t + \frac{(X_i t)^2}{2!} + \frac{(X_i t)^3}{3!} + \cdots = \exp(X_i t) \tag{6.4.4}$$

m-Server Station. In this case, node i consists of m identical servers with rate μ_i. For the m-server station

$$g_i(t) = 1 + X_i t + \frac{X_i^2}{2!}t^2 + \cdots + \frac{X_i^{m_i}}{m_i!}t^{m_i} + \sum_{n_i > m_i} \frac{X_i^{n_i}}{m_i! m_i^{n_i - m_i}} t^{n_i}$$

$$= \sum_{k=0}^{m-1} \frac{X_i^k}{k!}t^k + \frac{(X_i t)^m}{m!\left[1 - \frac{X_i t}{m}\right]} \tag{6.4.5}$$

If all of the stations are of the IS type, then $(X_1 + \cdots + X_M)^N / N!$ is the exact expression for the partition function. One can therefore assume that there

is at least one load-independent or m-server station i. It follows then that $Z_G(t)$ is a meromorphic function with at least one pole. Canonical approximation based on residues (theorem 3.2.1) is easy to apply only in the case when the smallest pole is simple. Moreover, even for this case, if the other poles are not simple, it is difficult to estimate the relative error for a given N. As we shall see, canonical approximation based on the saddle point overcomes these difficulties.

To apply the method, define $f(t)$ by

$$f(t) = \log Z_G(t) - (N+1)\log t = \sum_{i=1}^{M} \log g_i(t) - (N+1)\log t \tag{6.4.6}$$

The saddle-point t_N is found from

$$f'(t_N) = \sum_{i=1}^{M} \frac{g_i'(t_N)}{g_i(t_N)} - \frac{N+1}{t_N} = 0 \tag{6.4.7}$$

Canonical approximation (theorem 3.2.1) gives the following expression for the partition function

$$Z_N = \frac{Z_G(t_N)}{t_N^{N+1}\sqrt{2\pi f^{(2)}(t_N)}}[1+\epsilon_N] \tag{6.4.9}$$

The relative error of canonical approximation is given by

$$\begin{aligned}\epsilon_N =& \frac{1}{24}(3\nu_4 - 5\nu_3^2) \\ &+ \frac{1}{1152}(168\nu_3\nu_5 + 385\nu_3^4 - 630\nu_3^2\nu_4 - 24\nu_6 + 105\nu_4^2) + \cdots\end{aligned} \tag{6.4.10}$$

where ν_s are defined by

$$\nu_s = f^{(s)}(t_N) \Big/ \left(\sqrt{f^{(2)}(t_N)}\right) \tag{6.4.11}$$

ERROR ANALYSIS. Let us analyze the relative error of canonical approximation. To do so, let us establish a bound on the first term of the relative error.It is given by the following lemma:

Lemma 6.4.1. The first (dominant) term $\epsilon_N^{(1)}$ of the relative error of canonical approximation satisfies $|\epsilon_N^{(1)}| \ll 0.04$.

The proof of this lemma can be found in the Appendix at the end of this chapter (section 6.12). The above result says that the first (most dominant) term of the relative error is well below 0.04. From theorem 3.2.1, the rest of the terms for the relative error are of the same order as the first term. In the actual computations of the exact and approximate partition functions for over 40 random networks with $N > 20$ and $M > 5$, the relative error was below 5%.

Complexity. Let us analyze the complexity of canonical approximation. The most essential part is the calculation of t_N. From theorem 3.1.2, the saddle point $t_N \in (0, \min(m_i/X_i))$. Without loss of generality, assume $X_1/m_1 = \max(X_i/m_i)$. Since $f'(0) < 0$ and $f'(m_1/X_1) > 0$, one can apply the bisection algorithm ([Kron79]). The method is stable and convergent.

To estimate the number of iterations needed to compute t_N, note that X_i's are defined up to a multiplicative constant. Without loss of generality, one can assume that $m_1/X_1 = 1$. Thus, $t_N \in (0, 1)$. Since at each step of the bisection method the error is halved, to compute the saddle point with accuracy 2^{-k}, one would need at most k iterations. For example, with $k = 20$, one can compute the

saddle point up to the sixth decimal point in at most 20 iterations. Thus, for all practical purposes, the number of iterations is bounded by some constant. Since each evaluation of $f'(t_N)$ takes an $O(M)$ arithmetic operations, the computation of t_N can also be accomplished in an $O(M)$ arithmetic operations. There are no additional storage requirements.

6.5. PERFORMANCE MEASURES

Having obtained the closed-form approximation for Z_N, we can now compute a number of important performance measures for each node: utilization, throughput, mean queue length, and mean waiting time. We will show how to obtain these measures without explicitly calculating the partition function. The main result of this section is the following lemma:

Lemma 6.5.1. For a separable single-class closed queueing network, the average queue length $Q_N^{(i)}$ at node i can be computed as follows:

$$Q_N^{(i)} \approx \frac{X_i}{g_i(t_N)} \frac{\partial g_i(t_N)}{\partial X_i} - \frac{X_i}{2f^{(2)}(t_N)} \frac{\partial^* f^{(2)}(t_N)}{\partial X_i} + \frac{X_i f^{(3)}(t_N)}{2[f^{(2)}(t_N)]^2} \frac{\partial f'(t_N)}{\partial X_i} \qquad (6.5.1)$$

where the partial derivative $\frac{\partial^* f^{(2)}(t_N)}{\partial X_i}$ is taken explicitly with respect to X_i, treating t_N as an independent variable.

The proof of this lemma is given in the appendix at the end of this chapter. Let us show how to compute the average queue lengths for different types of servers.

Load-Independent Server. For this server, $g_i(t) = 1/(1 - X_i t)$. Therefore, one obtains the following expression for the average queue length at node i:

$$Q_N^i \approx \frac{X_i t_N}{1 - X_i t_N} \left[1 - \frac{X_i}{(1 - X_i t_N)^2 t_N f^{(2)}(t_N)} + \frac{f^{(3)}(t_N)}{2(1 - X_i t_N) t_N [f^{(2)}(t_N)]^2} \right] \qquad (6.5.2)$$

We call a node i a bottleneck node if $X_i = \max_j(X_j)$. Let us show that for the non-bottleneck nodes, to $O(1/N)$ we have the following expression

$$Q_N^i \approx \frac{X_i t_N}{1 - X_i t_N} \tag{6.5.3}$$

First, note that

$$f^{(2)}(t_N) \geq \left(\frac{N+1}{t_N}\right)^2 \qquad \text{and} \qquad X_i t_N < m_1 X_i / X_1$$

Therefore, for the second term in the brackets we get

$$0 < \frac{X_i}{(1 - X_i t_N)^2 t_N f^{(2)}(t_N)} < \frac{X_i m_1 X_1}{(X_1 - m_1 X_i)^2 (N+1)^2} = O(1/N^2) \tag{6.5.4}$$

For the third term in the brackets we can show

$$\begin{aligned} \frac{f^{(3)}(t_N)}{2(1 - X_i t_N) t_N [f^{(2)}(t_N)]^2} &< \frac{1}{2(1 - X_i t_N) t_N \sqrt{f^{(2)}(t_N)}} \\ &\ll \frac{X_1}{2(X_1 - m_1 X_i)(N+1)} = O(1/N) \end{aligned} \tag{6.5.5}$$

Therefore, from equations 6.5.4 and 6.5.5, it follows that to $O(1/N)$ the queue length for the non-bottleneck node is given by 6.5.3.

The accuracy of the queue length approximation at node i (equation 6.5.3) depends on how "close" X_i is to X_1/m_1. Thus, in actual computations, one should compute the upper bound (equations 6.5.4 and 6.5.5) for the relative error. If it seems large, then one should consider including the error terms and use the equation 6.5.1. to calculate the average queue length.

A similar observation applies to the calculation of the mean waiting time below. First, let us compute the utlization $U_N^{(i)}$ at node i. From the Gordon-Newell

formula (equation 6.2.1) it follows that

$$Q_N^{(i)} = \frac{1}{Z_N}\sum_{j=1}^{N} X_i^k Z_{N-j} = X_i \frac{Z_{N-1}}{Z_N} + \frac{1}{Z_N}\sum_{j=1}^{N-1} X_i^j Z_{N-j-1} = U_N^{(i)}(1+Q_{N-1}^{(k)}) \quad (6.5.6)$$

Therefore, the utilization of node i is given by

$$U_N^{(i)} = \frac{Q_N^{(i)}}{1 + Q_{N-1}^{(i)}} \quad (6.5.7)$$

The throughput of node i is given by

$$T_N^{(i)} = \frac{U_N^{(i)}}{\mu_i} \quad (6.5.8)$$

The mean waiting time $W_N^{(i)}$ (both time spent waiting in the queue and time spent in service) is easily found from the Little's formula :

$$W_N^{(i)} = \frac{Q_N^{(i)}}{T_N^{(i)}} \quad (6.5.9)$$

The calculation of these performance measures is reduced to calculating the saddle points t_{N-1} and t_N. The computation of utilization, throughput, mean queue length, and average waiting time, may be performed in $O(M)$ arithmetic operations. These measures are easily expressed in terms of the saddle point without the explicit computation of the partition function. The algorithm requires a fixed size storage for intermediate computations vs. $O(N)$ of most of the other algorithms

([Buze73, Chan80, Reis80]). The accuracy of the approximation is of $O(1/N)$. This compares with the $O(MN)$ computational complexity and the $O(N)$ storage requirements of the convolution algorithm ([Buze73]) or the mean-value analysis algorithm ([Reis80]).

Infinite Server Station. For this station $g_i(t) = \exp X_i t$. From equation 6.5.1, the queue length $Q_N^{(i)}$ is given by

$$Q_N^{(i)} \approx X_i t_N + \frac{X_i f^{(3)}(t_N)}{2[f^{(2)}(t_N)]^{3/2}} \tag{6.5.10}$$

The average waiting time is easily obtained from the Little's formula:

$$W_N^{(i)} = \frac{Q_N^{(i)}}{X_i} \approx t_N + \frac{f^{(3)}(t_N)}{2[f^{(2)}(t_N)]^{3/2}} \tag{6.5.11}$$

m-Server Station. For this station

$$g_i(t) = \sum_{k=0}^{m-1} \frac{X_i^k}{k!} t^k + \frac{(X_i t)^m}{m!\left[1 - \frac{X_i t}{m}\right]} \tag{6.5.12}$$

Let us note the following

$$\frac{\partial g_i(t)}{\partial t} = X_i \sum_{k=0}^{m-2} \frac{X_i^k}{k!} t^k + \frac{m}{t} \frac{(X_i t)^m}{m!\left[1 - \frac{X_i t}{m}\right]} + \frac{(m-1)! X_i}{(X_i t)^m} \left(\frac{(X_i t)^m}{m!\left[1 - \frac{X_i t}{m}\right]} \right)^2 \tag{6.5.13}$$

Similarly,

$$\frac{\partial g_i(t)}{\partial X_i} = t \sum_{k=0}^{m-2} \frac{X_i^k}{k!} t^k + \frac{m}{X_i} \frac{(X_i t)^m}{m!\left[1 - \frac{X_i t}{m}\right]} + \frac{(m-1)! t}{(X_i t)^m} \left(\frac{(X_i t)^m}{m!\left[1 - \frac{X_i t}{m}\right]} \right)^2 \tag{6.5.14}$$

Therefore, all the terms for this station needed in the computation of the queue length (equation 6.5.1) can be computed in $O(m)$ arithmetic operations. Similarly, for other performance measures.

6.6. ANALYSIS OF THE LOAD-INDEPENDENT CASE

In this section we restrict our attention to the case where each node has a load independent server with rate μ_i. We will establish a simple "pocket calculator" algorithm to compute the performance measures.

Let us recall that the queue length can be computed as follows

$$Q_N^{(i)} \approx \frac{X_i t_N}{1 - X_i t_N} \left[1 - \frac{X_i}{(1 - X_i t_N)^2 t_N f^{(2)}(t_N)} + \frac{f^{(3)}(t_N)}{2(1 - X_i t_N) t_N [f^{(2)}(t_N)]^2} \right] \tag{6.6.1}$$

In the previous section, it was shown that the queue lengths for the non-bottleneck nodes can be approximated to $O(1/N)$ as follows

$$Q_N^{(i)} \approx \frac{X_i t_N}{1 - X_i t_N} \tag{6.6.2}$$

Since the total number of customers is N, the average queue length for the bottleneck nodes $1, \ldots, d_1$ can be found as follows

$$Q_N^{(i)} \approx \frac{1}{d_1} \left[N - \sum_{j=d_1+1}^{M} \frac{X_j t_N}{1 - X_j t_N} \right] \tag{6.6.3}$$

In the previous section it was shown that the utilization can be computed as follows:

$$U_N^{(i)} = \frac{Q_N^{(i)}}{1 + Q_{N-1}^{(i)}} \tag{6.6.4}$$

Since the throughput is $T_N^{(i)} = U_N^{(i)}/\mu_i$, by Little's formula, the average waiting time for "non-bottleneck" nodes i $(i > d_1)$ is

$$W_N^{(i)} \approx \frac{X_i t_N}{T_N^{(i)}(1 - X_i t_N)} \tag{6.6.5}$$

For the "bottleneck" nodes j $(j = 1, \ldots d_1)$, the average waiting time is easily found from equations 6.6.2 and 6.6.3 to be

$$W_N^{(j)} \approx \frac{1}{d_1 T_N^{(i)}} \left[N - \sum_{i=d_1+1}^{M} Q_N^{(i)} \right] \tag{6.6.6}$$

Just as it is the case for the average queue length, the waiting time for "bottleneck" nodes is asymptotically linear in the number of customers, whereas for other nodes it is (asymptotically) independent of N.

Asymptotic Analysis. Let us analyze the network as $N \mapsto \infty$. By theorem 3.1.2. one can approximate the saddle point $t_N \approx 1/X_1$. Then, the queue length, mean waiting time, and utilization of the non-bottleneck nodes can be computed by equations 6.6.2, 6.6.4 and 6.6.6 respectively. One obtains

$$\begin{cases} Q_\infty^{(i)} \approx \frac{X_i}{X_1 - X_i} \\ U_\infty^{(i)} = \frac{X_i}{X_1} \\ W_\infty^{(i)} \approx \frac{X_1 \mu_i}{X_1 - X_i} \end{cases} \tag{6.6.7}$$

For a bottleneck node j $(j = 1, \ldots, d_1)$, these measures can be computed by equations 6.6.3, 6.6.5 and 6.6.6 to yield

$$\begin{cases} Q_\infty^{(j)} = \frac{1}{d_1} \left[N - \sum_{i=d_1+1}^{M} \frac{X_i}{X_1 - X_i} \right] \\ U_\infty^{(j)} = 1 \\ W_\infty^{(j)} = \frac{\mu_j}{d_1} \left[N - \sum_{i=d_1+1}^{M} \frac{X_i}{X_1 - X_i} \right] \end{cases} \tag{6.6.8}$$

Note that asymptotically, the queue lengths of the non-bottleneck nodes are bounded, whereas for the non-bottleneck nodes they are proportional to N. A similar observation applies to the mean queue lengths. The behavior of the network is governed by the bottleneck nodes $1, \ldots, d_1$.

The above discussion suggests a simple algorithm to analyze such closed queueing networks (where all M nodes have load-independent servers) for large N. The input is a (unsorted) list of X_i's:

Step 1. Scan the list, searching for the $X^* = \max X_i$ and noting the number of times m that this maximum occurs.

Step 2. Scan the list again, computing the sum

$$S = \sum_j \frac{X_j}{X^* - X_j}$$

for all j with $X_j \neq X^*$.

Step 3. Scan the list again. For each X_i in the list, compute the average queue length $Q_N^{(i)}$ at node i as follows

$$Q_N^{(i)} = \begin{cases} \frac{X_i}{X_m - X_i} & \text{if } X_i \neq X^* \\ \frac{1}{m}[N - S] & \text{otherwise} \end{cases}$$

The algorithm can be implemented on a simple pocket calculator. One can write a similar "pocket calculator" algorithm for the computation of the waiting time and utilization.

EXAMPLE. Consider a random network consisting of 20 load independent nodes. The values for the X_i's are summarized below:

$X_1 = 0.974$	$X_2 = 3.662$	$X_3 = 3.044$	$X_4 = 1.613$	$X_5 = 0.542$
$X_6 = 0.942$	$X_7 = 0.309$	$X_8 = 3.566$	$X_9 = 3.189$	$X_{10} = 0.244$
$X_{11} = 3.354$	$X_{12} = 1.961$	$X_{13} = 1.896$	$X_{14} = 2.692$	$X_{15} = 4.419$
$X_{16} = 2.047$	$X_{17} = 3.999$	$X_{18} = 2.180$	$X_{19} = 0.098$	$X_{20} = 0.668$

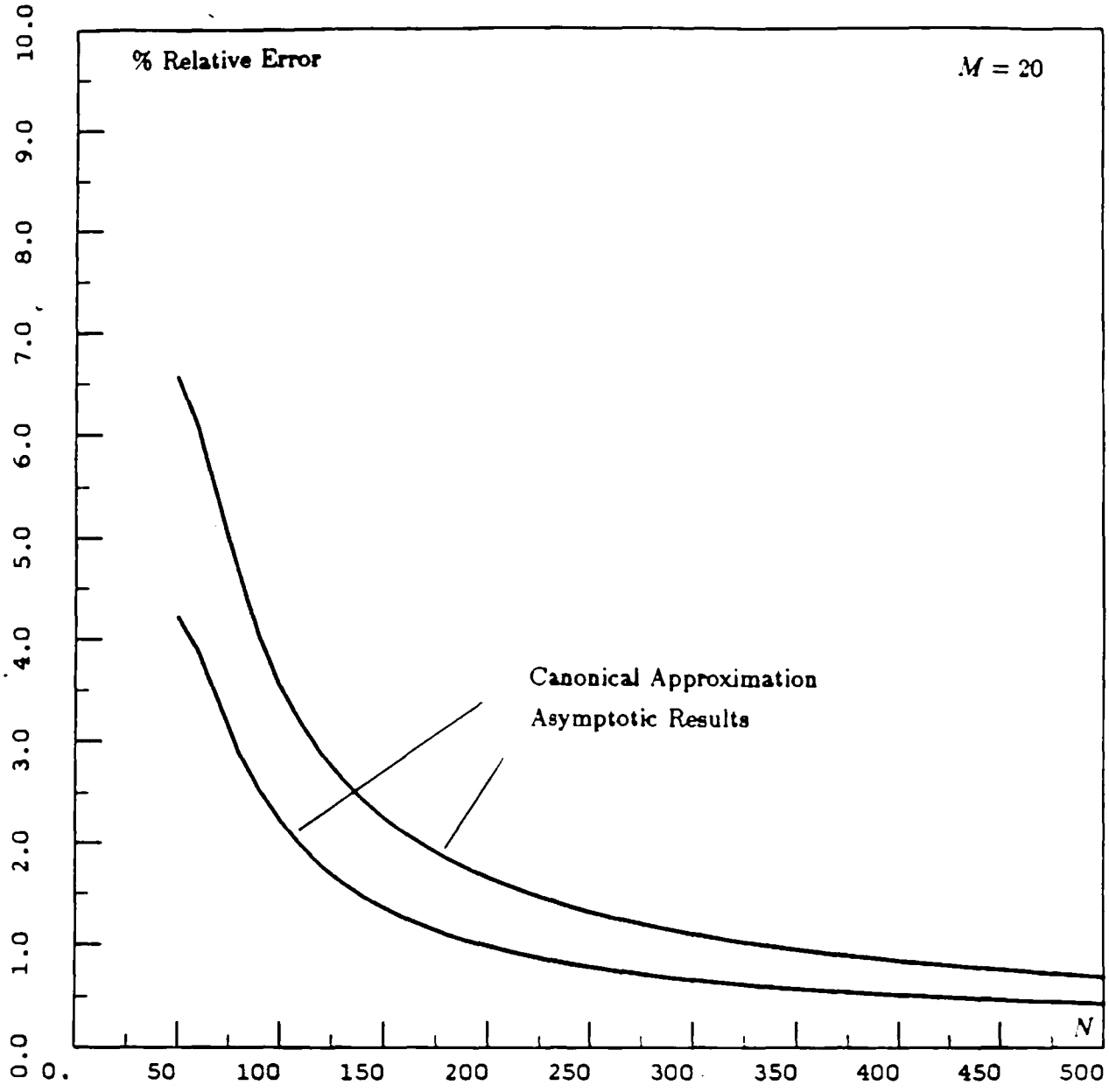

Figure 6.6.1: Average Error per Node for Load-Independent Case

For a given N, at each node we compute the relative error for the queue length calculated by equation 6.6.5 and by the above "calculator" program. (The exact queue length for each node is calculated by the mean-value algorithm). We then add up the magnitudes of the relative error at each node and divide it by the number of nodes. This gives us an average (per node) percentage relative error. The corresponding curves are illustrated in Figure 6.6.1. The average error obtained from equation 6.6.5 is referred to as the canonical approximation. The average error from the "calculator" program is referred to as the asymptotic approximation. As one can see from the corresponding curves in Figure 6.6.1, for $N > 100$ customers, the average relative error is well below 2% for the canonical approximation. If one uses the "calculator" algorithm, the error is slightly higher, but within 1% for $N > 300$. Since the "calculator" program requires trivial computations, it can be used to analyze load-independent networks with a very good accuracy.

6.7. COMPARISON WITH SOME OTHER ALGORITHMS

Let us briefly compare the method of canonical approximation with some of the other exact and approximation methods. Let us note that these methods have been designed primarily for multi-class queueing systems. An excellent survey is given in [Heid84].

The three exact algorithms to be considered are the convolution algorithm ([Buze73]), the MVA (mean-value analysis) algorithm ([Reis80]), and the local balance algorithm for a normalizing constant (LBANC) ([Chan80]). The convolution and LBANC require the explicit evaluation of Z_N. Since Z_N can exceed the floating point range of most machines with even small parameter values, these algorithms are not stable unless the proper scaling is used. The MVA avoids this problem by computing the averages without an explicit computation of Z_N. For the load-independent case, the evaluation of Z_N by these algorithms requires an $O(NM)$

multiplications and an $O(MN)$ additions. For the load-dependent case, the computational complexity is of $O(MN^2)$. A comparison of these algorithms is described in ([Chan80, Lave83]).

The above algorithms are exact. There has been a considerable amount of research on designing efficient approximation schemes. A good survey can be found in [Eage84]. Let us mention just a few of these.

The performance bound hierarchy (PBH) developed by Eager and Sevcik ([Eage84]) attempts to estimate the optimistic and pessimistic bounds on the performance measures from a network with $N - i$ customers (level i bound) and then applying the MVA equations until the original number of customers is reached. The optimistic or pessimistic character of the initial estimates is preserved by the recursion equations used by the solution algorithm. The level N bounds correspond to the exact applications of the MVA. The basic difficulty is obtaining good initial bounds for an initial level i. Moreover, it is difficult to estimate the error since only pessimistic and optimistic bounds are available. The methodology can be applied for multi-class queueing models, although the method becomes much more complicated.

Most of the approximation methods typically rely on the MVA for motivation ([Bard79, Chan82, Eage84]). A notable exception to these methods is the asymptotic expansion of the partition function developed by J. McKenna and D. Mitra ([Mcke84]). The expansion is based on the integral representation of $N_i!$ where N_i denotes the number of customers of class i. The partition function is then expressed in terms of some "large" parameter N reflecting the size of the network. It is assumed that the network is in "heavy usage" which in practical terms means that none of the processors are more than 90% utilized. Typically, less than 4 terms of the expansion are required to achieve satisfactory results. The corresponding computer program (PANACEA) requires about 6000 lines of code ([McKe84]). The method is particularly useful when the network has many classes, each with many customers.

When applied to single-class closed queueing networks, many of these methods are of theoretical interest, since it is typically not very costly to solve these networks exactly, unless N is very large. For such networks, canonical approximation is an attractive alternative to other methods. It is very simple, stable and easy to implement. Unlike most of the previous exact or approximation methods, it is not recursive. The algorithm computes simple closed-form approximation to the performance measures in $O(M)$ arithmetic operations. There are no additional storage requirements. Error bounds for the performance measures can be easily computed. The accuracy increases with N, while the complexity stays the same. The asymptotic analysis ($N \mapsto \infty$) when all the nodes are load-independent requires negligible computation and can be implemented on a pocket calculator.

6.8. SOME EXAMPLES

Example 1. The first example is a network of N customers consisting of M identical nodes. Thus, $X_i = X$. For this network, one can easily compute the exact value of the partition function and the average performance measures. The purpose of the example is to compare these exact values with the approximate values obtained by canonical approximation. The relative error of the approximation will be shown to be of $O(1/N)$.

For this network, the grand partition function

$$Z_G(t) = \frac{1}{(1 - Xt)^M} \tag{6.8.1}$$

Let $f(t)$ be

$$f(t) = -M\log(1 - Xt) - (N + 1)\log t \tag{6.8.2}$$

The saddle point t_N is easily computed from

$$f'(t) = \frac{MX}{1 - Xt} - \frac{N + 1}{t} = 0 \tag{6.8.3}$$

to be

$$t_N = \frac{N+1}{MX + (N+1)X} \tag{6.8.4}$$

From lemmas 6.2.1 and 6.2.2, one can easily compute the exact value of the partition function

$$Z_N = \binom{N+M-1}{M-1} X^N \tag{6.8.5}$$

On the other hand, canonical approximation gives the following

$$\begin{aligned} Z_N &\approx \frac{Z_G(t_N)}{t_N^{N+1}\sqrt{2\pi f^{(2)}(t_N)}} = \frac{X^N (N+M+1)^{N+M}\sqrt{M}}{M^M (N+1)^N \sqrt{2\pi(N+1)(N+M+1)}} \\ &= X^N \frac{\sqrt{2\pi(N+M+1)}\left(\frac{N+M+1}{e}\right)^{N+M+1} M(N+1)}{\sqrt{2\pi(N+1)}\left(\frac{N+1}{e}\right)^{N+1}\sqrt{2\pi M}\left(\frac{M}{e}\right)^M (N+M+1)^2} \end{aligned} \tag{6.8.6}$$

Using the Stirling approximation

$$N! = \sqrt{2\pi N}\left(\frac{N}{e}\right)^N \left[1 + \frac{1}{12N} + \frac{1}{288N^2} - \frac{139}{51,840N^3} + \cdots\right] \tag{6.8.7}$$

one can rewrite equation 6.8.6 as follows

$$Z_N \approx \frac{X^N (N+M+1)! M(N+1)}{(N+1)! M! (N+M+1)^2} = X^N \binom{N+M-1}{M-1}\left[1 - \frac{1}{N+M+1}\right] \tag{6.8.8}$$

Comparing with the exact expression 6.8.5, one obtains concludes that the relative error in computing Z_N by canonical approximation (ignoring the error terms in the Stirling approximation to $N!$) is of $O(1/N)$.

Let us now turn to the computation of the queue length. Since all the nodes are identical, one knows that the exact average queue length per node is

N/M. In the calculation of the queue length using the canonical approximation, one needs to know $f^{(2)}(t_N)$ and $f^{(3)}(t_N)$. From 6.8.3 one obtains

$$f^{(2)}(t_N) = \frac{(N+1)^2 + M(N+1)}{Mt_N^2} \text{ and } f^{(3)}(t_N) = \frac{(N+1)^3 - 2M^2(N+1)}{M^2 t_N^3} \tag{6.8.9}$$

Substituting these into the expression for the queue length (equation 6.6.11) one obtains

$$Q_N \approx \frac{N+1}{M}\left[1 - \frac{1}{M} + \frac{(N+1)^2 - 2M^2}{M(N+1)(N+M+1)}\right] \tag{6.8.10}$$

This can be rewritten as follows

$$Q_N \approx \frac{N}{M}[1+\epsilon_N] \tag{6.8.11}$$

where the relative error is

$$\epsilon_N = -\frac{1}{M(N+1)} - \frac{1}{(N+1)(N+M+1)} + \frac{1}{N} - \frac{1}{MN(N+1)} - \frac{1}{N(N+1)(N+M+1)}$$

It is easy to see that

$$|\epsilon_N| \ll \frac{1}{N} + \frac{2}{MN} + \frac{2}{MN^2} = O(1/N) \tag{6.8.12}$$

The relative error of canonical approximation is of $O(1/N)$. Similar conclusion holds for other performance measures.

Example 2. Let us consider the example analyzed by Eager ([Eage84]). The purpose of this example is to show how to analyze a network which contains different types of servers†. The network consists of 9 nodes with $N = 50$ customers. The first 8 nodes are load independent with the following parameters:

$$X_1 = 0.25 \quad X_2 = 0.2 \quad X_3 = 0.17 \quad X_4 = 0.16$$
$$X_5 = 0.08 \quad X_6 = 0.06 \quad X_7 = 0.05 \quad X_8 = 0.03$$

† An example of a network with only load-independent nodes was presented at the end of section 6.6.

The last, 9-th node is an infinite server node with $X_9 = 15$. The grand partition function for this network is

$$Z_G(t) = \exp(X_9 t) \prod_{i=1}^{8} \frac{1}{1 - X_i t}$$

The saddle-point is easily found from the equation

$$f'(t) = X_9 + \sum_{i=1}^{8} \frac{X_i}{1 - X_i t} - \frac{N+1}{t} = 0$$

to be $t_N = 2.94$. From these, one can easily compute $f^{(2)}(t_N) = 7.25$ and $f^{(3)}(t_N) = -1.96$. The queue length at the infinite server node is

$$Q_{50}^{(9)} \approx X_9 t_N + \frac{X_9 f^{(3)}(t_N)}{2[f^{(2)}(t_N)]^{3/2}} \approx 43.36$$

Comparing this with the exact value of $Q_{50}^{(9)} = 43.74$ one can see that the relative error ϵ_9 is 0.86%.

The queue lengths at load-independent nodes can be found from equation 6.6.1:

$$Q_N^i = \frac{X_i t_N}{1 - X_i t_N} \left[1 - \frac{X_i}{(1 - X_i t_N)^2 t_N f^{(2)}(t_N)} + \frac{f^{(3)}(t_N)}{2(1 - X_i t_N) t_N [f^{(2)}(t_N)]^2} \right]$$

The absolute values of the relative error ϵ_i for the queue length at node i are summarized below

$$\begin{array}{llll} \epsilon_1 = 4.41\% & \epsilon_2 = 0.19\% & \epsilon_3 = 0.56\% & \epsilon_4 = 0.51\% \\ \epsilon_5 = 1.04\% & \epsilon_6 = 0.08\% & \epsilon_7 = 0.33\% & \epsilon_8 = 4.06\% \end{array}$$

6.9. CONCLUSION

This chapter presented an application of the new method of canonical approximation to some closed markovian queueing networks. The method allows direct computation of global performance averages with high accuracy. Further work will extend the application of the new method to study multi-class closed queuing networks.

6.10. APPENDIX
PROOF OF TWO LEMMAS

In this section we give the proof of lemma 6.4.1 and lemma 6.5.1.

Lemma 6.4.1. The first (dominant) term $\epsilon_N^{(1)}$ of the relative error of the canonical approximation satisfies $|\epsilon_N^{(1)}| \ll 0.04$.

Proof. To start, introduce the following notation

$$h_i(t) = \log g_i(t_N) \tag{1}$$

It is easy to show that $h_i(t) > 0$ for $t > 0$. The derivatives of $f(t)$ at the saddle-point t_N can be written as follows

$$f^{(s)}(t_N) = \frac{1}{t_0^s}\left[\sum_{i=1}^{M} t_N^s h_i^{(s)}(t_N) + (-1)^s(s-1)!(N+1)\right] \tag{2}$$

The expression for ν_s (equation 6.4.11) can be rewritten as follows

$$\nu_s = \frac{\left[\sum_{i=1}^{M} t_N^s h_i^{(s)}(t_N) + (-1)^s (s-1)!(N+1)\right]}{\left[\sum_{i=1}^{M} t_N^2 h_i^{(2)}(t_N) + (N+1)\right]^{s/2}} \tag{3}$$

To $O(1/N)$, the first (dominant) term $\epsilon_N^{(1)}$ is

$$\epsilon_N^{(1)} = \frac{1}{24}(3\nu_4 - 5\nu_3^2)$$

$$= \frac{3\sum_{i=1}^{M} t_N^4 h_i^{(4)}(t_N)\left(N+1+\sum_{j=1}^{M} t_N^2 h_j^{(2)}(t_N)\right) - 5\left(\sum_{i=1}^{M} t_N^3 h_i^{(3)}(t_N)\right)^2}{24\left[\sum_{i=1}^{M} t_N^2 h_i^{(2)}(t_N) + (N+1)\right]^3}$$

$$\ll \frac{3\sum_{i,j=1}^{M} t_N^6 h_i^{(4)}(t_N) h_j^{(2)}(t_N) + 3(N+1)\sum_{i=1}^{M} t_N^4 h_i^{(4)}(t_N) - 5\left(\sum_{i=1}^{M} t_N^3 h_i^{(3)}(t_N)\right)^2}{24\left[\left(\sum_{i=1}^{M} t_N^2 h_i^{(2)}(t_N)\right)^3 + 3(N+1)\left(\sum_{i=1}^{M} t_N^2 h_i^{(2)}(t_N)\right)^2\right]}$$

It is easy to establish the following two inequalities

$$3(N+1)\sum_{i=1}^{M} t_N^4 h_i^{(4)}(t_N) \le 3(N+1)\left(\sum_{i=1}^{M} t_N^2 h_i^{(2)}(t_N)\right)^2$$

and

$$3\sum_{i,j=1}^{M} t_N^6 h_i^{(4)}(t_N) h_j^{(2)}(t_N) - 5\left(\sum_{i=1}^{M} t_N^3 h_i^{(3)}(t_N)\right)^2 < 3\sum_{i\neq j} t_N^6 h_i^{(4)}(t_N) h_j^{(2)}(t_N)$$

$$< \left(\sum_{i=1}^{M} t_N^2 h_i^{(2)}(t_N)\right)^3$$

Therefore,

$$\epsilon_N^{(1)} = \frac{1}{24}(3\nu_4 - 5\nu_3^2) \ll \frac{1}{24} \approx 0.04 \tag{4}$$

Similarly, it is easy to show that $\epsilon_N^{(1)} \gg -0.04$. Therefore, $|\epsilon_N^{(1)}| \ll 0.04$

∎

Now let us prove lemma 6.5.1.

Lemma 6.5.1. For a separable single-class closed queueing network the average queue length $Q_N^{(i)}$ at node i can be computed as follows:

$$Q_N^{(i)} \approx \frac{X_i}{g_i(t_N)} \frac{\partial g_i(t_N)}{\partial X_i} - \frac{X_i}{2f^{(2)}(t_N)} \frac{\partial^* f^{(2)}(t_N)}{\partial X_i} + \frac{X_i f^{(3)}(t_N)}{2[f^{(2)}(t_N)]^2} \frac{\partial f'(t_N)}{\partial X_i} \tag{5}$$

where the partial derivative $\frac{\partial^* f^{(2)}(t_N)}{\partial X_i}$ is taken explicitly with respect to X_i, treating t_N as an independent variable.

Proof. From the Gordon-Newell formula (equation 6.2.1) the average queue length at node i is

$$Q_N^{(i)} = \sum_{j=0}^{N} jP(n_i = j) = \frac{X_i}{Z_N} \frac{\partial Z_N}{\partial X_i} \tag{6}$$

By the chain rule,

$$\frac{\partial Z_N}{\partial X_i} = \frac{\partial^* Z_N}{\partial X_i} + \frac{\partial Z_N}{\partial t_N} \frac{\partial t_N}{\partial X_i} \tag{7}$$

The derivative $\frac{\partial^* Z_N}{\partial X_i}$ is taken "explicitly" with respect to X_i, treating t_N

as an independent variable. The first term of 7 gives

$$\frac{\partial^* Z_N}{\partial X_i} \approx \frac{1}{\sqrt{2\pi} t_N^{N+1}} \left[\frac{Z_G(t_N)}{g_i(t_N)\sqrt{f^{(2)}(t_N)}} \frac{\partial g_i(t_N)}{\partial X_i} - \frac{Z_G(t_N)}{2[f^{(2)}(t_N)]^{3/2}} \frac{\partial^* f^{(2)}(t_N)}{\partial X_i} \right] \tag{8}$$

$$= Z_N \left[\frac{1}{g_i(t_N)} \frac{\partial g_i(t_N)}{\partial X_i} - \frac{1}{2f^{(2)}(t_N)} \frac{\partial^* f^{(2)}(t_N)}{\partial X_i} \right]$$

The "explicit" partial derivative

$$\frac{\partial^* f^{(2)}(t_N)}{\partial X_i} = \frac{\partial}{\partial X_i} \left(\frac{g_i^{(2)}(t_N) g_i(t_N) - [g_i'(t_N)]^2}{[g_i(t_N)]^2} \right) \tag{9}$$

The second term of 7

$$\frac{\partial Z_N}{\partial t_N} \frac{\partial t_N}{\partial X_i} = \left(f'(t_N) Z_N - Z_N \frac{f^{(3)}(t_N)}{2f^{(2)}(t_N)} \right) \frac{\partial t_N}{\partial X_i} = -Z_N \frac{f^{(3)}(t_N)}{2f^{(2)}(t_N)} \frac{\partial t_N}{\partial X_i} \tag{10}$$

Since t_N is defined in terms of X_i by the equation $f'(t_N) = 0$, one has by the implicit function theorem

$$\frac{\partial t_N}{\partial X_i} = -\frac{1}{f^{(2)}(t_N)} \frac{\partial f'(t_N)}{\partial X_i} = \frac{g'(t_N)}{f^{(2)}(t_N)} \frac{\partial g_i(t_N)}{\partial X_i} - \frac{g_i(t_N)}{f^{(2)}(t_N)} \frac{\partial g_i'(t_N)}{\partial X_i} \tag{11}$$

The formula 5 follows from equations 6 - 10 ∎

CHAPTER 7

CANONICAL APPROXIMATION: CROSSBAR INTERCONNECTION

7.1. INTRODUCTION

The objective of this chapter is to apply the method of canonical approximation to analyze a crossbar interconnection network. The performance analysis of such networks has been motivated by telephone switching systems ([Bene65]) and the development of multiprocessor computer systems ([Bhan73, Bhya83, Pate81]). Recent developments towards wideband switches for future visual and data communications ([Huan84]) have spurred increased interest in the design and analysis of interconnection networks operating in asynchronous packet switching mode. In such systems connection requests may arrive from independent sources with quite different traffic characteristics (e.g. facsimile, voice, video).

The previous work on the analysis of the crossbar interconnection networks ([Bhan75, Bhya83, Pate81, Stre70]) assumed that interference is present only because of the contention at the intermediate switches or at the outputs (memory modules). This is a realistic model for multiprocessor systems, since a processor does not initiate two or more concurrent requests. In the model of a wideband crossbar switch considered here, however, there is an additional interference due to the possibility of two concurrent requests to the same input. This makes the model more difficult to analyze.

In this chapter we apply canonical approximation to obtain a simple way of calculating the non-blocking probabilities and the average concurrency of each class of connection requests. We consider two models of an $N \times N$ crossbar interconnection network. In the first model, there are different classes of connection

requests characterized by different statistics of arrivals and connection times. In each class, there are connection requests from any input to any output. Thus, for an $N \times N$ crossbar, there are N^2 possible connection requests in each class. This would be called the "equally likely" case model. The computation of the performance measures for this case reduces to solving a simple quadratic equation. The second model is characterized by the existence of a class of N possible connection requests of the form (i, j_i) from a node i to a specific (depending on i) output j_i. We would call this the "favorite-case". For this case, the computation of the performance measures reduces to solving a simple cubic equation. In addition, we obtain simple asymptotic (as $N \mapsto \infty$) expressions for these performance measures and compare some of our results with previous work.

7.2. THE MODEL

Consider an $N \times N$ crossbar network ([Krus83, Pate81]). We assume that an input can be connected to just one output. There are p classes of connection requests. A request (i, j) of class σ for a connection from input i to output j arrives according to Poisson distribution with rate λ_σ, and an established path is used for a period of time with mean $1/\mu_\sigma$. To avoid hardware and control complexity, there are no store-and-forward buffers at the intermediate switches in the network. Therefore, connection requests interfere if they try to access the same input or output. It is assumed that the blocked connection requests are cleared from the system. Such a system can be described by an interference model as presented in Chapter 3. The state of the system can be described by the set of p-tuples

$$S = \{(i_1, \ldots, i_p): \qquad i_1 + \cdots + i_p \leq N\}$$

where i_σ is the number of concurrent transmissions of class σ. Recall (theorem

3.2.1) that the steady-state probability distribution $\pi(i_1, \ldots, i_p)$ is given by

$$\pi(i_1, \ldots, i_p) = \frac{\rho_1^{i_1} \cdots \rho_p^{i_p}}{Z_N} \tag{7.2.1}$$

where $\rho_\sigma = \lambda_\sigma / \mu_\sigma$ and the partition function Z_N is given by

$$Z_N = \sum_{i_\sigma} \alpha_N^{i_1, \ldots, i_p} \rho_1^{i_1} \cdots \rho_p^{i_p}$$

Here $\alpha_N^{i_1, \ldots, i_p}$ is the number of distinct configurations with i_σ concurrent connections of class σ.

A number of important performance measures can be obtained once the partition function is calculated. The most important of these for a crossbar switch are the bandwidth E_N and the non-blocking probability B_N

Let us note here that the above model is not confined to a crossbar switch with input/output interference. We can consider a system consisting of two sets of N resources. An arriving request requires two resource (one from each set). For example, we can consider a database system where each transaction requires exclusive locking on two items in two different locations (assuming there are N items at each location). Such models are clearly "isomorphic" to the model of a crossbar, considered here.

7.3. ANALYSIS OF THE "EQUALLY LIKELY" CASE

The purpose of this section is to analyze the "equally likely" case. In this model connection requests are distinguished only by their statistics. Let us start with deriving an expression for the partition function. We have the following lemma:

Lemma 7.3.1. Let $\rho = \rho_1 + \cdots + \rho_p$. The partition function Z_N is given by

$$Z_N = N!\rho^N L_N\left(-\frac{1}{\rho}\right) \tag{7.3.1}$$

where $L_N(x)$ denotes the Laguerre polynomial of degree N.

Proof. To calculate the partition function, we first need to calculate $\alpha_N^{i_1,\ldots,i_p}$ - the number of configurations with i_σ connections of class σ. There are $\binom{N}{i_1,\ldots,i_p}$ ways to choose i_σ inputs for connections of type σ and $\binom{N}{i_1,\ldots,i_p}$ ways to choose i_σ outputs for connection of type σ. Once i_σ inputs and i_σ outputs are chosen, there are $i_\sigma!$ ways to achieve all permutations of those inputs and outputs.

Hence,

$$\alpha_N^{i_1,\ldots,i_p} = \binom{N}{i_1,\ldots,i_p}^2 i_1!\cdots i_p!$$

It is easy to show that we can rewrite this in the following way:

$$\alpha_N^{i_1,\ldots,i_p} = \binom{N}{i_1+\cdots+i_p}^2 (i_1+\cdots+i_p)!\binom{i_1+\cdots+i_p}{i_1,\ldots,i_p}$$

The partition function for such a network is

$$Z_N = \sum_{i_1+\cdots+i_p\le N} \alpha_N^{i_1,\ldots,i_p}\rho_1^{i_1}\cdots\rho_p^{i_p}$$

$$= \sum_{i_1+\cdots+i_p\le N} \binom{N}{i_1+\cdots+i_p}^2 (i_1+\cdots+i_p)!\left[\sum_{i_\sigma}\binom{i_1+\cdots+i_p}{i_1,\ldots,i_p}\rho_1^{i_1}\cdots\rho_p^{i_p}\right]$$

$$= \sum_{i=0}^{N}\binom{N}{i}^2 i!(\rho_1+\cdots+\rho_p)^i = \sum_{i=0}^{N}\binom{N}{i}^2 i!\rho^i$$

Let us rewrite the partition function as follows

$$Z_N = \sum_{i=0}^{N} \binom{N}{i} \frac{N!}{(N-i)!} \rho^i = \rho^N N! \sum_{i=0}^{N} (-1)^i \binom{N}{N-i} \frac{1}{i!} \left(-\frac{1}{\rho}\right)^i \tag{7.3.2}$$

From the definition of Laguerre polynomials of degree N

$$L_N(x) = \sum_{i=0}^{N} (-1)^i \binom{N}{N-i} \frac{x^i}{i!}$$

we immediately obtain 7.3.1. ∎

The expression for the partition function involves Laguerre polynomials $L_N(x)$ with $x < 0$. The standard expansions of $L_N(x)$ are available only for $x > 0$ ([Szeg39]). Laguerre polynomials have a radically different behavior for $x < 0$ (monotonically increasing) than for $x > 0$ (which has alternating signs for even and odd N when $x > 0$). For example, it is easy to check that for $x > 0$ there is no saddle point ([Henr77]).

Let us apply the method of canonical approximation. We need to compute the grand partition function for Z_N. This is easy. From the generating function of Laguerre polynomials ([Henr77])

$$\sum_{N=0}^{\infty} L_N(x) t^N = \frac{1}{1-t} \exp\left(\frac{-xt}{1-t}\right) \tag{7.3.3}$$

we obtain the following expression for the (exponential) grand partition function

$$Z_G(t) = \sum_{N=0}^{\infty} \frac{Z_N(\rho)}{N!} t^N = \sum_{N=0}^{\infty} L_N\left(-\frac{1}{\rho}\right) (\rho t)^N = \frac{1}{1-\rho t} \exp\left(\frac{t}{1-\rho t}\right) \tag{7.3.4}$$

Define $f(t)$ by

$$f(t) = \log Z_G(t) - (N+1)\log t = \left(\frac{t}{1-\rho t}\right) - \log(1-\rho t) - (N+1)\log t$$

Solving $f'(t) = 0$, we obtain a simple quadratic equation for the saddle point

$$(N+2)\rho^2 t^2 - (2N\rho + 3\rho + 1)t + (N+1) = 0 \tag{7.3.5}$$

Canonical approximation (theorem 3.2.1) gives the following expression for the partition function

$$Z_N = \frac{N! \exp\left(\frac{t_N}{1-\rho t_N}\right)}{(1-\rho t_N) t_N^{N+1} \sqrt{2\pi f''(t_N)}} [1 + \epsilon_N] \tag{7.3.6}$$

The relative error ϵ_N of the canonical approximation is given by

$$\epsilon_N = \frac{1}{24}(3\nu_4 - 5\nu_3^2) + \frac{1}{1152}(168\nu_3\nu_5 + 385\nu_3^4 - 630\nu_3^2\nu_4 - 24\nu_6 + 105\nu_4) + \cdots$$

where

$$\nu_s = f^{(s)}(t_N) \Big/ \left(\sqrt{f''(t_N)}\right)^s$$

The derivatives of $f(t)$ can be computed as follows

$$f^{(s)}(t_N) = \frac{s!\rho^{s-1}}{(1-\rho t_N)^{s+1}} + \frac{(s-1)!\rho^s}{(1-\rho t)^s} + \frac{(-1)^s(s-1)!(N+1)}{t_N^s} \tag{7.3.7}$$

It is easy to compute the first terms of the relative error. In Figure 7.3.1 we show the exact relative error for different N and some values of ρ. Because of the numerical overflow in the computation of the partition function, we calculate the exact relative error for relatively small N and relatively high traffic. Recall that there are N^2 connection requests of each class, each with load ρ_σ. The total offered traffic τ_σ of class σ is therefore $N^2\rho_\sigma$. As seen from Figure 7.3.1, the relative error

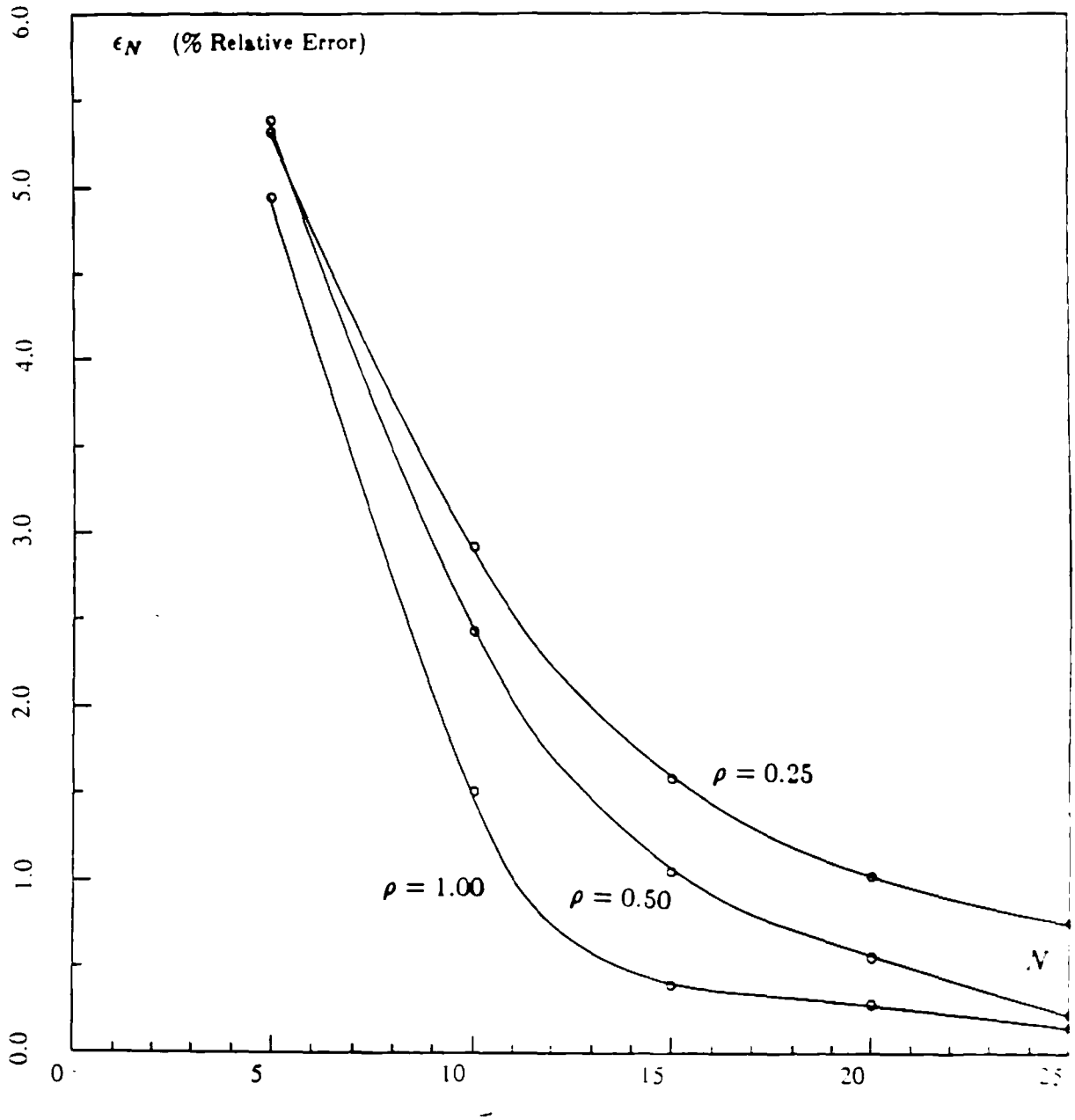

Figure 7.3.1: Relative Error for Different ρ and N

is small (less than 6%) even for very small values of N. It is decreasing for larger N, as should be expected from the method of canonical approximation.

Thus, for large N we have

$$Z_N \approx \frac{N!\exp\left(\frac{t_N}{1-\rho t_N}\right)}{(1-\rho t_N)t_N^{N+1}\sqrt{2\pi f''(t_N)}} \tag{7.3.8}$$

The approximate evaluation of the partition function is reduced to solving a quadratic equation.

Performance Measures. The most important performance measures of interconnection networks are the average number of concurrent transmissions and the non-blocking probability for each class. The expression for the bandwidth is given by the following lemma:

Lemma 7.3.2. The average number of concurrent transmissions of class σ is given by

$$E_\sigma \approx \rho_\sigma t_N(N+2) - \frac{2\rho\rho_\sigma(N+2)}{(1-\rho t_N)^2 f''(t_N)} + \frac{\rho_\sigma f^{(3)}(t_N)}{2[f''(t_N)]^2}\frac{2t_N+\rho t_N(1-\rho t_N)}{(1-\rho t_N)^3} \tag{7.3.9}$$

For large N, it can be computed as follows

$$E_\sigma \approx N\rho_\sigma t_N \tag{7.3.10}$$

The proof of this lemma can be found in the appendix at the end of this chapter (section 7.6). In Figure 7.3.2 we compare the bandwidth computed from the exact partition function (equation 7.3.2) and from the approximate ("asymptotic") expression 7.3.6. for a 20 × 20 crossbar and different values of ρ. As seen from these curves, even for relatively small N, the asymptotic formula for the average concurrency gives an excellent approximation. As before, we chose relatively heavy traffic to be able to compute the exact expression for the concurrency.

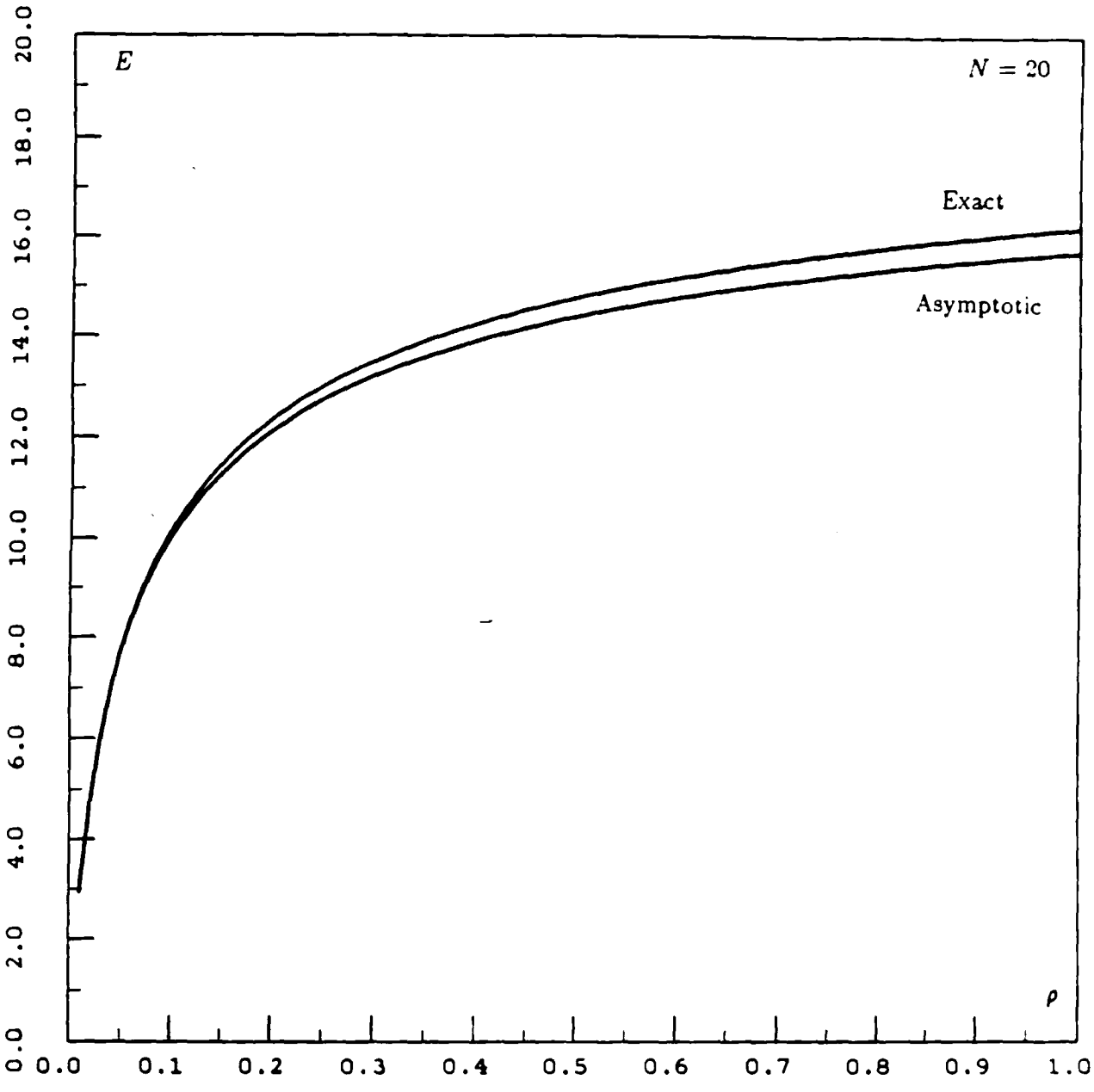

Figure 7.3.2: Bandwidth Comparison vs. Load

From equation 7.3.4 one can approximate t_N as follows

$$t_N \approx \frac{1}{\rho} - \frac{1}{\rho\sqrt{N\rho}} \tag{7.3.11}$$

Therefore, one can rewrite equation 7.3.10 as follows

$$E_N \approx \frac{N\rho_\sigma}{\rho} - \frac{N\rho_\sigma}{\rho\sqrt{N\rho}} \tag{7.3.12}$$

In particular, if there is only one class of connection requests, the average concurrency is given by

$$E_N \approx N - \sqrt{\frac{N}{\rho}} \tag{7.3.13}$$

The above results tell us that the full bandwidth of N is obtained at infinite ρ. It is interesting to compare our results to some previous results for the analysis of crossbar. In synchronous case, the bandwidth is proportional to N and is approximately $N(1 - \exp(-p))$ where p is the probability of a service request within a memory cycle ([Pate81, Stre70]). In fact, we can rewrite the expression for the bandwidth (equation 7.3.13) as $E_N \approx N(1 - \exp(-\log\frac{\rho}{N}))$ in exact analogy to the results obtained under the assumption of synchronous case. Let us note once again that in the previous analysis of a crossbar switch, the interference arises only over the memory contention. In our model, on the other hand, it can also arise both from concurrent requests to use the same input or output.

There are N^2 connections in each class. Therefore, the non-blocking probability B_σ for class σ is given by

$$B_\sigma \approx \frac{E_\sigma}{\rho N^2} \mapsto \frac{t_N}{N} \tag{7.3.14}$$

Note that the non-blocking probability that a request can proceed is independent of the class of the request. It does depend on ρ since t_N depends on ρ.

7.4. ANALYSIS OF THE "FAVORITE" CASE

Let us now analyze the so-called "favorite" case: Assume that there is a class of N possible requests of the form (i, j_i) from input i to a particular ("favorite") output j_i (depending on i). Without loss of generality, assume that processor $j_i = i$. Assume that a favorite output request comes with rate λ_1 and ordinary requests come at rate λ_2. The arrival process is assumed to be Poisson as before. The corresponding average service times are assumed to be distributed exponentially with parameters $1/\mu_1$ and $1/\mu_2$ respectively. It is easy to extend this to the case where there are different classes of connection requests. Let $\rho_1 = \lambda_1/\mu_1$ and $\rho_2 = \lambda_2/\mu_2$. Assume $\rho_2 \neq 0$ (the problem is trivial to analyze if $\rho_2 = 0$)

First, let us derive the expression for the partition function. It is given by the following

Lemma 7.4.1. The system partition function is given by the following expression:

$$Z_N = N! \sum_{i=0}^{N} \frac{\rho_1^i}{i!} \rho_2^{N-i} L_{N-i}\left(-\frac{1}{\rho_2}\right) \tag{7.4.1}$$

where as before $L_N(x)$ denotes Laguerre polynomial of degree N.

Proof. To write down the partition function, we need to calculate $\alpha_N^{i,j}$, which is the number of configurations for an $N \times N$ crossbar with i favorite memory and j ordinary connection requests. The number of ways to have i favorite memory connections is obviously $\binom{N}{i}$. This is because once one chooses i processors, the corresponding i memory modules are uniquely determined. This leaves a crossbar

of $N-i$ processors and $N-i$ memory modules for j ordinary connections.

$$\alpha_N^{i,j} = \binom{N}{i}\binom{N-i}{j}^2 j!$$

Therefore, the partition function is

$$Z_N = \sum_{i+j=0}^{N} \binom{N}{i}\binom{N-i}{j}^2 j!\rho_1^i\rho_2^j = \sum_{i=0}^{N}\binom{N}{i}\rho_1^i\left[\sum_{j=0}^{N-i}\binom{N-i}{j}^2 j!\rho_2^j\right]$$

$$= \sum_{i=0}^{N}\binom{N}{i}\rho_1^i(N-i)!\rho_2^{N-i}L_{N-i}\left(-\frac{1}{\rho_2}\right) = N!\sum_{i=0}^{N}\frac{\rho_1^i}{i!}\rho_2^{N-i}L_{N-i}\left(-\frac{1}{\rho_2}\right)$$

To apply the method, we need to compute the grand partition function. We can compute the (exponential) grand partition function as follows

$$Z_G(t) = \sum_{N=0}^{\infty}\frac{Z_N}{N!}t^N = \sum_{N=0}^{\infty}\left[\sum_{i=0}^{N}\frac{\rho_1^i t^i}{i!}L_{N-i}\left(-\frac{1}{\rho_2}\right)(\rho_2 t)^{N-i}\right]$$

$$= \left(\sum_{s=0}^{\infty}\frac{\rho_1^s t^s}{s!}\right)\left(\sum_{m=0}^{\infty}L_m\left(-\frac{1}{\rho_2}\right)(\rho_2 t)^m\right) \tag{7.4.2}$$

$$= \frac{1}{1-\rho_2 t}\exp\left(\rho_1 t + \frac{t}{1-\rho_2 t}\right)$$

Note that if $\rho_1 = 0$ (the equally likely case) the partition and grand partition functions are the same as those obtained for the equally likely case. Define the function $f(t)$ by

$$\begin{aligned} f(t) &= \log Z_G(t) - (N+1)\log t \\ &= \left(\rho_1 t + \frac{t}{1-\rho_2 t}\right) - \log(1-\rho_2 t) - (N+1)\log t \end{aligned} \tag{7.4.3}$$

The first derivative of $f(t)$ is

$$f'(t) = \rho_1 + \frac{1}{(1-\rho_2 t)^2} + \frac{\rho_2}{(1-\rho_2 t)} - \frac{N+1}{t} \tag{7.4.4}$$

The computation of the saddle point is reduced to solving a simple cubic equation. Note that the second and higher derivatives of $f(t)$ do not depend on ρ_1 and can be computed as in the equally likely case (equation 7.3.6) by taking $\rho_2 = \rho$.

The canonical approximation gives the following expression for the partition function

$$Z_N \approx \frac{N! \exp\left(\rho_1 t_N + \frac{t_N}{1-\rho_2 t_N}\right)}{(1-\rho_2 t_N) t_N^{N+1} \sqrt{2\pi f''(t_N)}} \tag{7.4.5}$$

Performance Measures. Let us first compute the average number of favorite memory requests. The main result is summarized in the following lemma.

Lemma 7.4.2. The average number of "equally likely" requests can be computed as before. The average number of "favorite memory" requests can be computed as

$$E_1 \approx t_N \rho_1 + \frac{\rho_1 f^{(3)}(t_N)}{2[f''(t_N)]^2} \tag{7.4.6}$$

For large N, it can be rewritten as

$$E_1 \approx t_N \rho_1 \tag{7.4.7}$$

The proof can be found in the Appendix at the end of this chapter (section 7.6). As in the previous case, for large N the non-blocking probabilities are

$$B_1 = B_2 \approx \frac{t_N}{N} \tag{7.4.8}$$

If we let $\rho_1 = \rho_2$ then $E_2 = NE_1$. This is easily explained. The total offered load at input i from "favorite" requests is $\tau_1 = \rho_1$. The total offered load

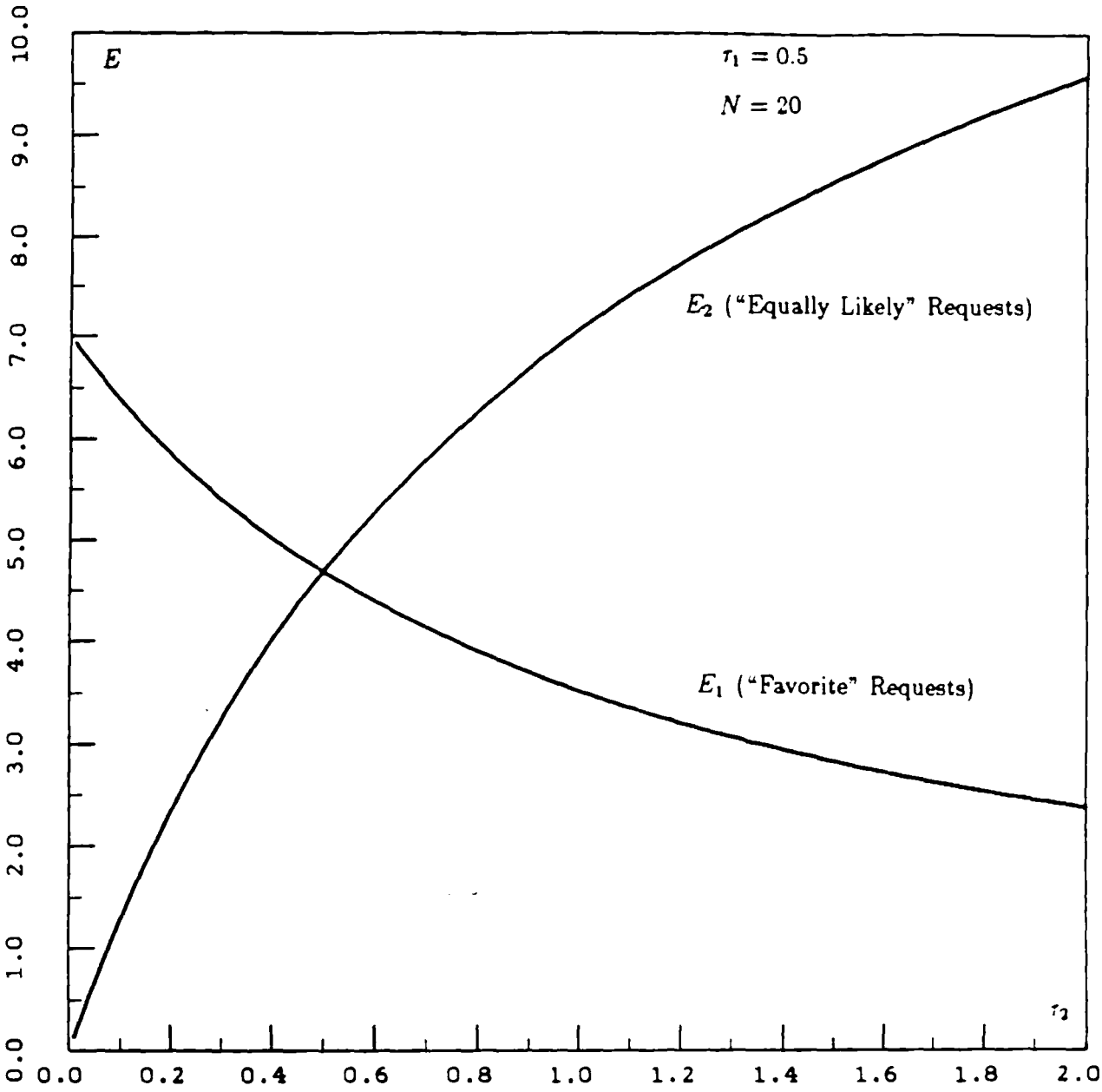

Figure 7.4.1: The "Starvation Phenomenon" in "Favorite" Case

at input i from "equally likely" requests is $\tau_2 = N\rho_2$ (all requests of the form (i, j) with $j = 1, \ldots N$). Thus, looking at input i, we would expect there to be N times as many "equally likely" requests than "favorite" requests. This is equivalent to saying that $E_2 = NE_1$.

Figure 7.4.1 gives the average concurrency of "favorite" and "equally likely" requests for $N = 20$ and fixed $\tau_1 = 0.5$. When $\tau_2 = 0$, only "favorite" requests are present. There is no interference among them. As one increases τ_2, there are more and more "equally likely" requests. Thus, E_2 increases and E_1 decreases. At some point τ_2^* one has $E_1 = E_2$. For $\tau_2 > \tau_2^*$ one gets $E_2 > E_1$. Further increase in τ_2 leads to "starvation" of "favorite" requests - only "equally likely" requests will be present.

7.5. CONCLUSION

This chapter presented an application of canonical approximation to the analysis of different models of a crossbar switch. The method gives an extremely simple way of calculating some performance measures without an explicit calculation of the partition function. The computations are reduced to solving a simple polynomial equation. Further work will extend the application of the method to the analysis of other systems, particularly of the multi-stage interconnection networks.

7.6. APPENDIX

In this appendix we provide the proofs to lemma 7.3.2 and lemma 7.4.2. Let us start with lemma 7.3.2:

Lemma 7.3.2. The average number of concurrent transmissions of class σ is

given by

$$E_\sigma \approx \rho_\sigma t_N(N+2) - \frac{2\rho\rho_\sigma(N+2)}{(1-\rho t_N)^2 f''(t_N)} + \frac{\rho_\sigma f^{(3)}(t_N)}{2[f''(t_N)]^2}\frac{2t_N + \rho t_N(1-\rho t_N)}{(1-\rho t_N)^3} \tag{1}$$

For large N, it can be computed as follows

$$E_\sigma \approx N\rho_\sigma t_N \tag{2}$$

Proof. The equation for the bandwidth is

$$E_\sigma = \frac{\rho_\sigma}{Z_N}\frac{\partial Z_N}{\partial \rho_\sigma} \tag{3}$$

By the chain rule, the partial derivative can be written as follows

$$\frac{\partial Z_N}{\partial \rho_\sigma} \approx \frac{\partial^* Z_N}{\partial \rho_\sigma} + \frac{\partial Z_N}{\partial t_N}\frac{\partial t_N}{\partial \rho_\sigma} \tag{4}$$

The derivative $\frac{\partial^* Z_N}{\partial \rho_\sigma}$ is taken "explicitly" with respect to ρ_σ, treating t_N as an independent variable. The first term of 4. gives

$$\frac{\partial^* Z_N}{\partial \rho_\sigma} = \frac{1}{\sqrt{2\pi}t_N^{N+1}}\left[\frac{Z_G(t_N)(t_N^2 + t_N - \rho t_N^2)}{(1-\rho t_N)^3\sqrt{f''(t_N)}} - \frac{Z_G(t_N)}{2[f''(t_N)]^{3/2}}\frac{\partial^* f''(t_N)}{\partial \rho_\sigma}\right]$$

From $f'(t_N) = 0$ it follows that

$$\frac{t_N^2 + t_N - \rho t_N^2}{(1-\rho t_N)^2} = (N+2)t_N$$

The explicit partial derivative can be found as follows

$$\frac{\partial^* f''(t_N)}{\partial \rho_\sigma} = \frac{\partial^*}{\partial \rho_\sigma}\left[\frac{N+1-(N+2)\rho^2 t_N^2}{t_N^2(1-\rho t_N)^2}\right] = \frac{2\rho(N+2)}{(1-\rho t_N)^3}$$

Therefore, equation 4. can be rewritten as follows

$$\frac{\partial^* Z_N}{\partial \rho_\sigma} \approx Z_N \left[(N+2)t_N - \frac{2\rho(N+2)}{(1-\rho t_N)^3 f''(t_N)} \right] \tag{5}$$

Let us now compute the second term of 4. We have

$$\frac{\partial Z_N}{\partial t_N}\frac{\partial t_N}{\partial \rho_\sigma} \approx \left(f'(t_N)Z_N - Z_N \frac{f^{(3)}(t_N)}{2f''(t_N)} \right) \frac{\partial t_N}{\partial \rho_\sigma} = -Z_N \frac{f^{(3)}(t_N)}{2f''(t_N)} \frac{\partial t_N}{\partial \rho_\sigma} \tag{6}$$

Since t_N is defined by equation $f'(t_N) = 0$, we have by the implicit function theorem

$$\frac{\partial t_N}{\partial \rho_\sigma} = -\frac{1}{f''(t_N)} \frac{\partial^* f'(t_N)}{\partial \rho_\sigma} = -\frac{2t_N + \rho t_N(1-\rho t_N)}{f''(t_N)(1-\rho t_N)^3} \tag{7}$$

Equations 3 - 7 imply 1. To prove 2, let us rewrite 1 as follows

$$E_\sigma \approx N\rho_\sigma t_N[1+\epsilon]$$

where

$$\epsilon = \frac{2}{N} - \frac{2\rho(N+2)}{Nt_N(1-\rho t_N)^2 f''(t_N)} + \frac{f^{(3)}(t_N)}{2N[f''(t_N)]^2} \frac{2+\rho(1-\rho t_N)}{(1-\rho t_N)^3} \tag{8}$$

Obviously, $2/N \mapsto 0$. Since

$$f''(t_N) = \frac{\rho}{(1-\rho t_N)} \left[\frac{1}{(1-\rho t_N)^2} + \frac{\rho}{1-\rho t_N} \right] + \frac{\rho}{(1-\rho t_N)^3} + \frac{N+1}{t_N^2}$$

$$> \frac{\rho(N+1)}{(1-\rho t_N)t_N} + + \frac{N+1}{t_N^2} = \frac{N+1}{t_N^2(1-\rho t_N)}$$

we obtain for the second term in 8

$$\frac{2\rho(N+2)}{Nt_N(1-\rho t_N)^2 f''(t_N)} < \frac{2\rho(N+2)(1-\rho t_N)}{N(1+\rho(1-\rho t_N))}$$

$$= \left(1+\frac{2}{N}\right) \frac{2\rho(1-\rho t_N)}{1+\rho(1-\rho t_N)} < 2\left(1+\frac{2}{N}\right)(1-\rho t_N) \mapsto 0$$

At the last derivation we used the fact that $t_N \mapsto \rho$. Similarly, this is also the case for the last term in 8. Thus, for large N, expression 2 is valid ∎

Let us now prove the second lemma in this appendix.

Lemma 7.4.2. The average number of "equally likely" requests can be computed as before (lemma 7.3.1). The average number of "favorite" memory requests can be computed as

$$E_1 \approx t_N \rho_1 + \frac{\rho_1 f^{(3)}(t_N)}{2[f''(t_N)]^2} \tag{9}$$

For large N, it can be rewritten as

$$E_1 \approx t_N \rho_1 \tag{10}$$

Proof. To calculate E_2, we can proceed as in lemma 7.3.1. It is easy to show that all partial derivatives (e.g. $f'(t_N)$, $f^{(3)}(t_N)$) are independent of ρ_1. Therefore, the average concurrency E_2 can be computed as in lemma 7.3.1. For large N, we have $E_2 \approx N\rho_2 t_N$. Note that the average concurrency E_2 does depend on ρ_1 since t_N depends on ρ_1.

Let us calculate E_1. By the chain rule, the partial derivative can be written as follows

$$\frac{\partial Z_N}{\partial \rho_1} = \frac{\partial^* Z_N}{\partial \rho_1} + \frac{\partial Z_N}{\partial t_N}\frac{\partial t_N}{\partial \rho_1} \tag{11}$$

The derivative $\frac{\partial^* Z_N}{\partial \rho}$ is taken "explicitly" with respect to ρ, treating t_N as an independent variable. Noting that $\frac{\partial^* f''}{\partial \rho_1} = 0$ one obtains for the first term in 11 that

$$\frac{\partial^* Z_N}{\partial \rho} \approx t_N Z_N \tag{12}$$

The second term in 11 can be written as follows

$$\frac{\partial Z_N}{\partial t_N}\frac{\partial t_N}{\partial \rho_1} = \left(f'(t_N)Z_N - Z_N\frac{f^{(3)}(t_N)}{2f''(t_N)}\right)\frac{\partial t_N}{\partial \rho_1} = -Z_N\frac{f^{(3)}(t_N)}{2f''(t_N)}\frac{\partial t_N}{\partial \rho_1} \tag{13}$$

Since t_N is defined in terms of ρ_1 by the equation $f'(t_N) = 0$, we have by the implicit function theorem

$$\frac{\partial t_N}{\partial \rho_1} = -\frac{1}{f''(t_N)} \frac{\partial^* f'(t_N)}{\partial \rho_1} = -\frac{1}{f''(t_N)} \tag{14}$$

Equations 12 - 14 imply 8. To prove 9, let us rewrite 8 as follows

$$E_1 \approx t_N \rho_1 \left[1 + \frac{f^{(3)}(t_N)}{2t_N[f''(t_N)]^2}\right] \tag{15}$$

Since

$$f^{(3)}(t_N) < \frac{6(N+1)\rho_2^2}{t_N(1-\rho_2 t_N)^2} \qquad \text{and} \qquad f''(t_N) > \frac{N+1}{t_N^2(1-\rho_2 t_N)}$$

we obtain

$$\frac{f^{(3)}(t_N)}{2t_N[f''(t_N)]^2} < \frac{3t_N}{N+1} \mapsto 0$$

Therefore,

$$E_1 \approx \rho_1 t_N \qquad \blacksquare$$

CHAPTER 8

SUMMARY AND DIRECTIONS FOR FUTURE RESEARCH

8.1. SUMMARY OF THIS WORK

In this thesis we analyzed a variety of distributed systems modelled by time-reversible Markov chains: multihop packet radio networks, database systems, (single-class) closed queueing networks and interconnection networks. We introduced a new method of canonical approximation to compute the partition function and performance averages for these systems. The technique is based on a method of steepest descent, which was introduced by Darwin and Fowler ([Darw22]) into statistical mechanics.

The new method is computationally simple and gives closed form approximation to a variety of performance measures. The computational complexity of the method is independent of the system size, whereas the precision increases with the system size. We showed how it can be applied to analyze a variety of distributed systems. A detailed summary of the results of this thesis was presented in Chapter 1 (section 1.3).

Finally, let us note that the grand partition function $Z_G(t)$ is just a z-transform of Z_N. Therefore, one can view canonical approximation as a new method to approximately "invert" the z-transform.

8.2. POSSIBLE EXTENSIONS

There are a number of interesting extensions which are of major interest

in the analysis of computer systems discussed in this thesis.

- It should be interesting to investigate if one can extend this approximation for the case where the grand partition function is a function of several variables. Many of the results of classical one-variable analysis are not easily generalizable to multi-variable case. Multi-dimensional complex analysis is currently a very active area of research ([Aize83]).

If canonical approximation can be extended, this may give a powerful method for inverting multi-dimensional transforms. This will allow to solve a number of interesting problems. For example, of major theoretical and practical interest are computational algorithms to analyze multi-class closed queueing networks. It can be shown that the grand partition function for markovian multi-class queueing models is a meromorphic function of several variables (the number of variables equals to the number of classes). Thus, if canonical approximation could be extended to multi-dimensional case, it may give an algorithm to compute the performance averages for such networks in time independent of the population sizes. This would be a major accomplishment in the field of performance analysis.

- It is possible to consider asymmetric interference, non-exclusive interference (i.e. agents are allowed to proceed concurrently with some probability), and synchronous mode of operation. In the analysis of systems presented in this work, it is assumed that blocked arrivals are cleared. The obvious way to modify this is to assume that blocked arrivals are regenerated from the same statistics as new arrivals. Such an approximation is routinely used in analyzing multi-access schemes ([Klei76]). A more adequate solution is to introduce a blocked state and somehow account for it in the expression for the partition function.

- In the analysis of static locking models for database systems, it was assumed that locks are acquired in exclusive mode only. It is more realistic and interesting (certainly more difficult) to try to analyze that model when both exclusive and non-exclusive locks are allowed. Some results have been obtained ([Lave84]).

- In the analysis of packet radio networks, only CSMA and C-BTMA were considered. There are a number of protocols which admit the product form solution and which could be analyzed by the new method. It should be interesting to analyze different network topologies such as random graphs. Finally, one should apply similar ideas to analyze these protocols in synchronous mode.

- In the analysis of interconnection networks, no multi-stage networks (e.g. shuffle-exchange, delta, cube) were analyzed. Interestingly, there are "similar" problems in statistical mechanics such as the dimer problem of placing diatomic molecules on multi-partite graphs. Are there any approximations to analyze such systems ?

These are just some of the immediate extensions stemming from this thesis. It is believed that some of these questions can be solved based on the ideas developed in this work.

8.3. OTHER DIRECTIONS FOR FUTURE RESEARCH

In this work we introduced a new method of approximation similar to that developed in statistical mechanics. Interestingly, the main criticism of using the statistical mechanics analogies to systems analysis has been the skepticism concerning the ability of computing the partition function ([Bene65]). This work has shown how one can use the methods developed in statistical physics to compute the partition function and apply it to the analysis of a variety of systems.

On the other hand, we considered only product-form solution models whose analysis required the computation of the partition function Z_N and for which the grand partition function $Z_G(t)$ is easily computable. Although many interesting product-form solution models can be analyzed using canonical approximation, it is of major interest to develop new approximations for other product-form and non-

product form solution models. It would be interesting to see if one can use other approximation methods† from statistical mechanics in the performance analysis.

One of such methods of statistical mechanics to consider is the method of **Cluster Expansions**. For a number of interacting physical systems, one can obtain the approximation for the partition function in the form of *series expansion*. The main terms in this expansion denote the corresponding non-interfering results, while the subsequent terms denote the corrections arising from intercomponent interactions in the system. A systematic method of carrying out these expansions, in the case of real gases obeying classical statistics, was first developed by Meyer ([Maye37]) and is called the *cluster expansion* ([Path84]). It would be interesting to see if we can apply the same ideas to analyze some non-product form models (e.g. finite buffer queueing networks). For example, consider a Jackson network of queues. Under the assumption that no interference occurs between any two queues in the system, the partition function of a network is the product of the partition functions of the respective queues. As soon as queues start interacting with each other (e.g. by blocking if the buffers are finite at corresponding nodes), the non-interference model is no longer valid. By analogy to cluster expansions, one can try to admit such interference by including adequate terms in the partition function: the first term in the series will be the Jacksonian non-interference element while higher order terms reflect mutual blocking configurations.

8.4. CONCLUSION

It is often believed that cross-fertilization of ideas from different areas of science leads to fruitful results and future directions of scientific research. This

† To avoid repetition of the previous section, we do not mention multi-dimensional saddle point and the (approximate) integral expansion of the Laplace inversion formula.

work has demonstrated the use of the methods and ideas from statistical mechanics in the performance analysis of computer systems. These analogies give a powerful computational method of canonical approximation for systems analysis.

This thesis presented an approach of applying the methods of statistical mechanics to performance analysis of computer systems. Future work will extend the applications to more complex and realistic models of interference and import other powerful computational methods for performance analysis.

BIBLIOGRAPHY

[Abra82] M. Abramowitz, I.A. Stegun, "Handbook of Mathematical Functions", *Dover*, 1982.

[Abra70] N. Abramson, "The ALOHA System - Another Alternative for Computer Communications", *Proc. AFIPS Fall Joint Conference*, AFIPS Press, 1970, pp. 281-285.

[Aize83] I.A. Aizenberg, A.P. Yuzhakov, "Integral Representations and Residues in Multidimensional Complex Analysis", *Amer. Math. Soc.*, Translations of Mathematical Monographs, **58**, Providence, 1983.

[Alfh66] L.V. Alfhors, "Complex Analysis", *McGraw-Hill*, New York, 1966.

[Alle78] A.O. Allen, "Probability, Statistics and Queueing Theory with Computer Science Applications", *Academic Press*, 1978.

[Bada80] D.A. Badal, "Concurrency Control Overhead or Closer Look at Blocking vs. Non-Blocking Concurrency Control Mechanisms", *Proc. Third Berkeley Sympsium on Distributed Systems*, Berkeley, CA, 1980, pp. 85-103.

[Bard79] Y. Bard, "Some Extensions to Multiclass Queueing Network Analysis", *Performance of Computer Systems*, North-Holland, New York, 1979.

[Bask75] F. Baskett, K.M. Chandy, R.R. Muntz, F. Palacios, "Open, Closed and Mixed Networks of Queues with Different Classes of Customers", *J. Assoc. Comput. Mach.*, **22**, 4, April 1975, pp. 248-260.

[Bene65] V.E. Beneš, "Mathematical Theory of Connecting Networks and Telephone Traffic", *Academic Press*, New York, 1965.

[Bern78] P.A. Bernstein, J.B. Rothnie, N. Goodman, C.A. Papadimitriou, "The Concurrency Control Mechanism of SDD-1: A System for Distributed Databases", *IEEE Trans. on Software Engineering*, **4**, 3, May 1978, pp. 154-169.

[Bern81] P.A. Bernstein, N. Goodman, "Concurrency Control in Distributed Database Systems", *ACM Comput. Surveys*, **13**, 1981, pp. 185-221.

[Bhan73] D.P. Bhandarkar, S.H. Fuller, "A Survey of Techniques for Analyzing Memory Interference in Multiprocessor Computer Systems", *Carnegie-Mellon University*, Technical Report, Pittsburg, Pa, April 1973

[Bhan75] D.P. Bhandarkar, "Analysis of Memory Interference in Multiprocessors", *IEEE Trans. on Computers*, **C-24**, 9, September 1975, pp. 897-908.

[Bhan77] D.P. Bhandarkar, "Some Performance Issues in Multiprocessor System Design", *IEEE Trans. on Computers*, **C-26**, 5, May 1977, pp. 506-511.

[Bhuy83] L.N. Bhuyan, C.W. Lee, "An Interference Analysis of Interconnection Networks", *Proc. International Conference on Parallel Processing*, Miami, Fl, August 1983, pp. 2-9.

[Boor80] R.R. Boorstyn, A. Kershenbaum, "Throughput Analysis of Multihop Packet Radio", *Proc. International Conference on Communications*, Seattle, Wa, June 1980.

[Braz84] J.M. Brazio, F.A. Tobagi, "Theoretical results in Throughput Analysis of Multihop Packet Radio Networks", *Proc. IEEE International Conference on Communications*, Amsterdam, May 1984, pp. 448-455.

[Braz85] J.M. Brazio, F.A. Tobagi, "Throughput Analysis of Spread Spectrum Multihop Packet Radio Networks", *Proc. INFOCOM 1985*, Washington D.C., March 1985, pp. 256-265.

[Brue80] S.C. Bruell, G. Balbo, "Computational Algorithms for Closed Queueing Networks", *North-Holland*, 1980.

[Brui81] N.G. De Bruijn, "Asymptotic Methods in Analysis", *Dover Publications*, New York, 1981.

[Buze71] J.P. Buzen, "Queueing Network Models of Multiprogramming", Ph.D.

Thesis, *Harvard University*, May 1971.

[Buze73] J.P. Buzen, "Computational Algorithms for Closed Queueing Networks with Exponential Servers", *Comm. Assoc. Comput. Mach.*, **16**, 9, September 1973, pp. 527-531.

[Cape79] J.I. Capetanakis, "Genealized TDMA: The Multiaccessing Tree Protocol", *IEEE Trans. on Communications* **COM-27**, 10, October 1979, pp. 1476-1484.

[Carl75] A.B. Carleial, M.E. Hellman, "Bistable Behavior of ALOHA-type Systems", *IEEE Trans. on Communications*, **COM-23**, 4, April 1975, pp. 401-409.

[Chan75] K.M. Chandy, U. Herzog, L. Woo, "Parametric Analysis of Queueing Network Models", *IBM J. Res. Develop.*, **19**, January 1975, pp. 36-42.

[Chan80] K.M. Chandy, C.H. Sauer, "Computational Algorithms for Product Form Queueing Networks", *Comm. Assoc. Comput. Mach.*, **23**, 10, October 1980, pp. 573-583.

[Chan82] K.M. Chandy, D. Neuse, "Linearizer: A Heuristic Algorithm for Queueing Network Models of Computer Systems", *Comm. Assoc. Comput. Mach*, **25**, 1982, pp. 126-134.

[Ches83] A. Chesnais, E. Gelenbe, I. Mitrani, "On the Modelling of Parallel Access to Shared Data", *Comm. Assoc. Comput. Mach.*, **26**, 3, March 1983, pp. 196-202.

[Coop81] R.B. Cooper, "Introduction to Queueing Theory", *North-Holland*, New York, 1981.

[Cops65] E.T. Copson, "Asymptotic Expansions", *Cambridge University Press*, 1965.

[Dani54] H.E. Daniels, "Saddlepoint Approximations in Statistics", *Ann. Math. Stat.*, **Vol. 25**, 1954, pp. 631-650.

[Darw22] C.G. Darwin, R.H. Fowler, "On the Partition of Energy", *Phil. Mag.*, **44**,

1922, pp. 450-479.

[Diaz81] D.M. Diaz, J.R. Jump, "Analysis and Simulation of Buffered Delta Networks", *IEEE Trans. on Computers*, **C-30**, 4, April 1981, pp. 273-282.

[Eage84] D.L. Eager, "Bounding Algorithms for Queueing Network Models of Computer Systems", *Ph.D. Thesis*, University of Toronto, 1984.

[Eswa76] K.P. Eswaran, J.N. Gray, R.A. Lorie, I.L. Traiger, "The Notions of Consistency and Predicate locks in a Database System", *Comm. Assoc. Comput. Mach.*, **19**, 11, November 1976, pp. 624-634.

[Fayo77] G. Fayolle, E. Gelenbe, J. Labetoulle, "Stability and Optimal Control of the Packet Switching Broadcast Channel", *J. Assoc. Comput. Mach.*, **24**, 3, July 1977, pp. 375-386.

[Fell66] W. Feller, "An Introduction to Probability Theory and its Applications", *J. Wiley and Sons*, N.Y., N.Y., 1966.

[Ferd71] A.E. Ferdinand, "An Analysis of the Machine Interference Model", *IBM J. Res. Develop.*, **2**, February 1971, pp. 129-142.

[Ferg77] M. Ferguson, "An Approximate Analysis of Delay for Fixed and Variable Length Packets in an Unslotted ALOHA Channel", *IEEE Trans. on Communications*, **COM-25**, 7, July 1977, pp. 644-654.

[Fran85] P. Franaszek, J.T. Robinson, "Limitations of Concurrency in Transaction Processing", *ACM Trans. on Database Systems*, **10**, No.1, March 1985, pp. 1-28.

[Gall82] B.I. Galler, "Concurrency Control Performance Issues", *Technical Report CSRG-147*, University of Toronto, 1982.

[Gibb02] J.W. Gibbs, "Elementary Principles in Statistical Mechanics", *Yale University Press*, New Haven, 1902.

[Goke73] R. Goke, G.J. Lipovski, "Banyan networks for Partitioning on Multipro-

cessor Systems", *Proc. First Conference on Computer Architecture*, University of Florida, 1973, pp. 21-28.

[Gord67] W.J. Gordon, G.F. Newell, "Closed Queueing Systems with Exponential Servers", *Oper. Res.*, **15**, 1967, pp. 254-265.

[Goya84] A. Goyal, T. Agerwala, "Performance Analysis of Future Storage Systems", *IBM J. Res. Dev.*, **28**, No.1, January 1984, pp. 95-108.

[Gray78] J. Gray, "Notes on Database Operating Systems", *IBM Res. Report No. RJ 2188*, February 1978.

[Gray82] J. Gray, P. Homan, R. Obermarck, H. Korth, "A Straw Man Analysis of Waiting and Deadlock", *IBM Res. Report No. 3066*, February 1982.

[Heid84] P. Heidelberger, S.S. Lavenberg, "Computer Performance Evaluation Methodology", *IBM Res. Report No. 10493*, April 1984.

[Hell67] H. Hellerman, "Digital Computer System Principles", *McGraw-Hill*, New York, 1967, pp. 228-229.

[Henr77] P. Henrici, "Applied and Computational Complex Analysis", *J. Wiley*, New York, 1977.

[Huan84] A. Huang, S. Knauer, "Starlite: A Wideband Digital Switch", *Proc. INFOCOM 1984*, pp. 121-125.

[Iran79] K.B. Irani, H.L. Lin, "Queueing Network Models for Concurrent Transaction Processing in a Database System", *Proc. ACM SIGMOD International Conference on Management of Data, Boston*, MA, 1979, pp. 134-142.

[Jack57] J.R. Jackson, "Networks of Waiting Lines", *Oper. Res.*, **5**, 1957, pp 518-521.

[Jack63] J.R. Jackson, "Jobshop-like Queueing Systems", *Mgmt Sci.*, **10**, 1963, pp. 131-142.

[Jury64] E.I. Jury, "Theory and Application of the z-Transform Method", *J. Wiley*, New York, 1964.

[Kahn78] R. Kahn, S. Gronemeyer, J. Burchfield, R. Kunzelman, "Advances in Packet Radio Technology", *Proc. IEEE* **66**, November 1978, pp. 1468-1496.

[Kapl85] M.A. Kaplan, E. Gulko, "Analyic Properties of Multiple-Access Trees", *IEEE Trans. on Information Theory*, **IT-31**, 2, March 1985, pp. 255-263.

[Kauf81] J.S. Kaufman, "Blocking in a Shared Resource Environment", *IEEE Trans. on Communications*, **COM-29**, 10, October 1981, pp. 1474-1485.

[Kell80] F.P. Kelly, "Reversibility and Stochastic Networks", *J. Wiley and Sons*, New York, 1980.

[Klei75a] L. Kleinrock, "Queueing Systems, Volume I: Theory", *J. Wiley and Sons*, New York, 1975.

[Klei75b] F.A. Tobagi, L. Kleinrock, "Packet Switching in Radio Channels: Part I - Carrier Sense Multiple-Access Modes and Their Throughput-Delay Characteristics", *IEEE Trans. on Communications*, **COM-23**, 12, December 1975, pp. 1400-1416.

[Klei76] L. Kleinrock, "Queueing Systems, Volume II: Computer Applications", *J. Wiley and Sons*, New York, 1976.

[Klei80] L. Kleinrock, M. Scholl, "Packet Switching in Radio Channels: New Conflict-Free Multiple Access Schemes", *IEEE Trans. on Communications*, **COM-28**, 7, July 1980, pp. 1015-1029.

[Kolc78] V.F. Kolchin, B.A. Sevast'yanov, V.P. Chistyakov, "Random Allocations", *J. Wiley*, New York, 1978.

[Kron79] L.I. Kronsjo, "Algorithms, Their Complexity and Efficiency", *J. Wiley and Sons*, New York, 1979.

[Krus83] C.P. Kruskal, M. Snir, "The Performance of Multistage Interconnection Networks for Multiprocessors", *IEEE Trans. on Computers*, **C-32**, 12, December 1983, pp. 1091-1098.

[Kuro84] J.F. Kurose, M. Schwartz, Y. Yemini, "Multiple-Access Protocols and Time-Constrained Communication", *ACM Comput. Surveys*, **16**, 1, March 1984, pp. 43-71.

[Lam75] S.S. Lam, L. Kleinrock, "Packet Switching in a Multiaccess Broadcast Channel: Dynamic Control Procedures", *IEEE Trans. on Communications*, **COM-23**, 9, September 1975, pp. 891-904.

[Lam77] S.S. Lam, "An Extension of Moore's Result for Closed Queueing Networks", *IBM J. Res. Dev.*, July 1975, pp. 384-387.

[Lamp78] L. Lamport, "Time, Clocks and the Ordering of Events in a Distributed System", *Commun. Assoc. Comput. Mach.*, **21**, 7, July 1978, pp. 558-565..

[Lang76] T. Lang, H.S. Stone, "A Shuffle-Exchange Network with Simplified Control", *IEEE Trans. on Computers*, **C-25**, January 1976, pp. 55-66.

[Lang82] A.M. Langer, A.W. Shum, "The Distribution of Granule Accesses Made by Database Transactions", *Comm. Assoc. Comput. Mach.*, **25**, 11, November 1982, pp. 831-832.

[Lave83] S.S. Lavenberg, "Computer Performance Modeling Handbook", *Academic Press*, New York, 1983.

[Lave84] S. Lavenberg, "A Simple Analysis of Exclusive and Shared Lock Contention in a Database System", *IBM Res. Report No.10356*, January 1984.

[Lawr75] D.H. Lawrie, "Access and Alignment of data in an Array Processor". *IEEE Trans. on Computers*, **C-25**, 12, December 1975, pp. 1145-1155.

[Magl83] B. Maglaris, R. Boorstyn, A. Kershenbaum, "Extensions to the Analysis of Multihop Packet Radio Networks", *Proc. INFOCOM 1983*, San Diego, California.

April 1983, pp. 390-395.

[Mars82a] M.A. Marsan, "Bounds on Bus and Memory Interference in a Classs of Multiple-Bus multiprocessor Systems", *Proc. Third International Conference on Distributed Computing Systems*, New York, N.Y., October 1982, pp. 792-798.

[Mars82b] M.A. Marsan, G. Balbo, G. Conte, F. Gregoretti, "Comparative Performance Analysis of Single Bus Multiprocessor Architectures", *IEEE Trans. on Computers*, **C-31**, 12, December 1982, pp. 1179-1191.

[Mars82c] M.A. Marsan, M. Gerla, "Markov Models for Multiple Bus Multiprocessor Systems", *IEEE Trans. on Computers*, **C-31**, 3, March 1982, pp. 239-248.

[Marsan82d] M.A. Marsan, G. Balbo, G. Conte, F. Gregoretti, "Modelling Bus Contention and Memory Interference in a Multiprocessor System", *IEEE Trans. on Computers*, **C-32**, 1, January 1983, pp. 60-72.

[Maye40] J.E. Mayer, M.G. Mayer, "Statistical Mechanics", *J. Wiley and Sons*, New York, 1940.

[McKe84] J. McKenna, D. Mitra, "Asymptotic Expansions and Integral Representations of Moments of Queue Lengths in Closed Markovian Networks", *J. Assoc. Comput. Mach.*, **Vol. 31**, No. 2, April 1984, pp. 346-360.

[Medi83] J.S. Meditch, C.A. Lea, "Stability and Optimization of the CSMA and CSMA/CD Channels", *IEEE Trans. on Communications*, **COM-31**, 6, June 1983, pp. 763-774.

[Metz76] J.J. Metzner, "On Improving Utilization in ALOHA Networks", *IEEE Trans. on Communications*, **COM-24**, April 1976, pp. 447-448.

[Mitr84] D.Mitra, P.J. Weinberger, "Probabilistic Models of Database Locking: Solutions, Computational Algorithms and Asymptotics", *J. Assoc. Comput. Mach.*, **31**, 4, October 1984, pp. 855-879.

[Moor72] F.R. Moore, "Computational Model of a Closed Queueing Network with

Exponential Servers", *IBM J. Res. Develop.*, **11**, November 1972, pp. 567-572.

[Morr84] R.J.T. Morris, W.S. Wong, "Performance of Concurrency Control Algorithms with Nonexclusive Access", *Proc. 10-th International Symposium on Computer Performance*, Paris, France, December 1984, pp. 87-102.

[Mudg82] T.N. Mudge, B.A. Makrucki, "An Approximate Queueing Model for Packet Switched Multistage Interconnection Networks", *Proc. Third International Conference on Distributed Computer Systems*, Miami, Fl, October 1982, pp. 556-562.

[Munt75] R.R. Muntz, "Analytic Modelling of Interactive Systems", *Proc. IEEE Special Issue on Interactive Computer Systems*, **63**, June 1975, pp. 946-953.

[Nels84] R. Nelson, "The Stochastic Cusp, Swallowtail and Hyperbolic Umbilic Catastrophies as Manifest in a Simple Communication Model", *Proc. Performance 1984*, North-Holland, 1984, pp. 207-224.

[Nels85] R. Nelson, L. Kleinrock, "Rude-CSMA: A Multihop Channel Access Protocol", *IEEE Trans. on Communications*, **COM-23**, 8, August 1985, pp. 785-791.

[Path84] R.K. Pathria, "Statistical Mechanics", *Pergamon Press*, New York, 1984.

[Pate79] J.N. Patel, "Processor-Memory Interconnections for Multiprocessors", *Proc. Sixth International Symposium on Computer Architecture*, April 1979, pp. 168-177.

[Pate81] J.N. Patel, "Performance of Processor-Memory Interconnections for Multiprocessors", *IEEE Trans. on Computers*, **C-30**, 10, October 1981, pp. 771-780.

[Peas77] M.C. Pease, "An Adaptation of the Fast Fourier Transform for Parallel Processing", *J. Assoc. Comput. Mach.*, **15**, April 1977, pp. 252-264.

[Pins84] E. Pinsky, Y. Yemini, "A Statistical Mechanics of Interconnection Networks", *Proc. Performance'84*, E. Gelenbe (editor), North-Holland 1984.

[Pins85a] E. Pinsky, Y. Yemini, "The Canonical Approximation in Performance

Analysis", *Proc. International Seminar on Computer Networks and Performance Evaluation*, Tokyo, Japan, September 1985.

[Pfis85] G.F. Pfista, V.A. Norton, "Hot Spot Contention and Combining in Multistage Interconnection Networks", *IEEE Trans. Computers*, **C-34**, No. 10, October 1985, pp. 943-948.

[Poti80] D. Potier, P. Leblanc, "Analysis of Locking Policies in Database Management Systems", *Commun. Assoc. Comp. Mach.*, **23**, October 1980, pp. 584-593.

[Pres74] C.J. Preston, "Gibbs States on Countable Sets", *Cambridge University Press*, 1974.

[Ravi72] C.N. Ravi, "On the Bandwidth and Interference in Interleaved Memory Systems", *IEEE Trans. on Computers*, **C-21**, 8, August 1972, pp. 899-902.

[Reed78] D.P. Reed, "Naming and Synchronization in Decentralized Computer Systems", *MIT Laboratory for Computer Science*, MIT/LCS/TR-205, September 1978.

[Reis80] M. Reiser, S.S. Lavenberg, "Mean-Value Analysis of Closed Multichain Queueing Networks", *J. Assoc. Comput. Mach.*, **27**, 1980, pp. 313-322.

[Ries77] D.R. Ries, M. Stonebraker, "Effects of Locking Granularity in a Database Management System", *ACM Trans. on Database Systems*, **2**, 4, September 1977, pp. 233-246..

[Ries79] D.R. Ries, M. Stonebraker, "Locking Granularity Revisited", *ACM Trans. on Database Systems*, **4**, 4, June 1979, pp. 210-227.

[Rior62] J.Riordan, "Stochastic Service Systems", *J. Wiley*, New York, 1962.

[Rior78] J.S. Riordan, "An Introduction to Combinatorial Analysis", *J. Wiley and Sons*, New York, 1978.

[Robe74] L.G. Roberts, "Data by the Packet", *IEEE Spectrum* **11**, February 1974, pp. 46-51.

[Roon53] P.G. Rooney, "Some Remarks on Laplace's Method", *Trans. Royal Soc. Canada*, **47**, 3, June 1953, pp. 29-34.

[Rose78] D.J. Rosenkranz, R.E. Stearns, P.M. Lewis, "System level Concurrency Control for Distributed Database Systems", *ACM Trans. on Database Systems*, **3**, 2, June 1978, pp. 178-199.

[Rubi83] I. Rubin, "Synchronous and Channel-Sense Asynchronous Dynamic Group Random Access Schemes for Multiple Access Communications", *IEEE Trans. on Comm.*, **COM-31**, 9, September 1983, pp. 1063-1077.

[Ruel69] D. Ruelle, "Statistical Mechanics: Rigorous Results", *Benjamin Inc.*, New York, 1969.

[Schw79] P. Schweitzer, "Approximate Analysis of Multiclass Closed Networks of Queues", *Proc. International Conf. on Stochastic Control and optimization*, Free University of Amsterdam, Amsterdam, The Netherlands, 1979.

[Schr60] E. Schrödinger, "Statistical Thermodynamics", *Cambridge University Press*, 1960.

[Sevc81] K. Sevcik, "Data Base System Performnace Prediction Using an Analytical Model", *Proc. Seventh International Conference on Very Large Data Bases*, Cannes, France, September 1981, pp. 182-198.

[Shum76] A.W. Shum, "Queueing Models for Computer Systems with General Service Time Distributions", Ph.D. Thesis, *Harvard University*, 1976.

[Shum81] A.W. Shum, P.G. Spirakis, "Performance Analysis of Concurrency Control Methods in Database Systems", *Proc. Performance 1981*, F.J. Kylstra (editor), North-Holland, 1981, pp. 1-19.

[Silv83a] J.A. Silvester, L. Kleinrock, "On the Capacity of Single-Hop Slotted ALOHA Networks for Various Traffic Matrices and Transmission Strategies", *IEEE Trans. on Communications*, **COM-31**, 8, August 1983, pp. 983-991.

[Silv83b] J.A. Silvester, L. Kleinrock, "On the Capacity of Multi-Hop Slotted ALOHA Networks with Regular Structure", *IEEE Trans. on Communications*, **COM-31**, 8, August 1983, pp. 974-982.

[Smit78] A.J. Smith, "Sequentiality and Prefetching in Data Base Systems", *ACM Trans. on Database Systems*, **3**, 3, September 1978, pp. 223-248.

[Spit71] F. Spitzer, "Random Fields and Interacting Particle Systems", *Amer. Math. Soc.*, Providence, Rhode Island, 1971.

[Stoe80] J. Stoer, R. Bulirsch, "Introduction to Numerical Analysis", *Spinger-Verlag*, New York, 1980.

[Stone71] H.S. Stone, "Parallel Processing with the Perfect Shuffle", *IEEE Trans. on Computers*, **C-20**, 2, February 1971, pp. 153-161.

[Stre70] W.D. Strecker, "Analysis of the Instruction Execution Rate in Certain Computer Structures", Ph.D. Thesis, *Carnegie-Mellon University*, Pittsburg, Pa., 1970.

[Szeg39] G. Szego, "Orthogonal Polynomials", *Amer. Math. Soc.*, Providence, R.I., 1937.

[Taga83] H. Tagaki, L. Kleinrock, "Analysis of Polling Schemes", UCLA Computer Science Technical Report, *University of California*, Los Angeles, CA, 1983.

[Tay84] Y.C. Tay, "A Mean Value Performance Model for Locking in Databases", Ph.D. Dissertation, *Harvard University*, 1984.

[Temp81] H.N.V. Temperley, "Graph Theory and Applications", *J. Wiley and Sons*, New York, 1981.

[Than81] S. Thanawastien, V.P. Nelson, "Interference Analysis of Shuffle/Exchange Networks", *IEEE Trans. on Computers*, **C-30**, 8, August 1981, pp. 545-556.

[Thom82] A. Thomasian, B. Nadji, "Algorithms for Queueing Network Models

of Multiprogrammed Computers Using Generating Functions", *Computer Performance*, **Vol. 2**, No. 3, September 1981, pp.99-124.

[Thom83] A. Thomasian, I.K. Ryu, "A Decomposition Solution to the Queueing Network Model of the Centralized DBMS with Static Locking", *Proc. ACM SIGMETRICS Conference on Measurement and Modelling of Computer Systems*, Minneapolis, MN, August 1983, pp. 82-92.

[Thur82] K.J. Thurber, "Distributed Processor Communication Architecture", *Plenum Press*, Massachusetts, 1982.

[Toba75] F.A. Tobagi, L. Kleinrock, "Packet Switching in Radio Channels: Part II - The Hidden Terminal Problam in Carrier Sense Multiple-Access and the Busy-Tone Solution", *IEEE Trans. on Communications*, **COM-23**, 12, December 1975, pp. 1417-1433.

[Toba76] F.A. Tobagi, L. Kleinrock, "Packet Switching in Radio Channels: Part III - Polling and (Dynamic) Split-Channel Reservation Multiple Access", *IEEE Trans. on Communications*, **COM-24**, 8, August 1976, pp. 832-844.

[Toba77] F.A. Tobagi, L. Kleinrock, "Packet Switching in Radio Channels: Part IV - Stability Considerations and Dynamic Control in Carrier Sense Multiple Access", *IEEE Trans. on Communications*, **COM-25**, 10, October 1977, pp. 1103-1119.

[Toba80] F.A. Tobagi, "Analysis of a Two-Hop Centralized packet Radio Network - Part I: Slotted ALOHA", *IEEE Trans. on Communications*, **COM-28**, 2, February 1980, pp. 196-207.

[Toba83] F.A. Tobagi, J.M. Brazio, "Throughput Analysis of Multihop Packet Radio Networks under Various Channel Access Schemes", *Proc. INFOCOM 1983*, San Diego, California, April 1983, pp. 381-389.

[Toba85] F.A. Tobagi, D.H. Shur, "Performance Evaluation of Channel Access Schemes in Multihop Packet Radio Networks with Regular Structure by Simulation", *CSL Technical Report No. 85-278*, Stanford University, 1985.

[Widd46] D.V. Widder, "The Laplace Transform", *Princeton University Press*, 1946.

[Will76] A.C. Williams, R.A. Bhandiwad, "A Generating Function Approach to Queueing Network Analysis of Multiprogrammed Computers", *Networks*, **6**, 1976, pp. 1-22.

[Wint47] A. Wintner, "The Fourier Transforms of Probability Distributions", *John Hopkins University*, Baltimore, 1947.

[Yemi80a] Y. Yemini, L. Kleinrock, "Interfering Queueing Processes in Packet Switched Broadcast Communication", *Proc. IFIP Congress*, IFIP, Tokyo, 1980.

[Yemi83] Y. Yemini, "A Statistical Mechanics of Distributed Resource Sharing Mechanisms", *Proc. INFOCOM 1983*, San Diego, California, April 1983, pp. 531-540.

[Zaho80] J. Zahorjan, "The Approximate Solution of Large Queueing Network Models", Ph.D. Thesis, *Technical Report CSRG-122*, Computer Systems Research Group, University of Toronto, 1980.

[Zima79] J.M. Ziman, "Models of Disorder", *Cambridge University Press*, 1979.

APPENDIX 1

A STATISTICAL MECHANICS OF CONCURRENCY

In this appendix we present an analogy between the model of interference presented in Chapter 3 and the interaction (hard sphere) models of a lattice gas ([Ruel69]) in statistical mechanics.

The general theory of statistical mechanics states that all the equilibrium properties of a system of interacting components are known if one can calculate the system partition function:

$$Z = \sum_{i} \exp(-\beta \epsilon_i) \tag{1.1}$$

where the summation is over all microstates of the system (enumerated by i); ϵ_i is the energy of i-th microstate and $\beta = 1/kT$. Here T is the absolute temperature and k is the Boltzmann constant. This is the generating function, with one term for each permissible configuration of the components, each term being given a weight related to the energy of the corresponding configuration.

All thermodynamic functions describing the system (energy, entropy, pressure) are obtained in terms of the partition function and its derivatives. The equilibrium probability that the system is in microstate i is given by the Gibbs canonical distribution:

$$\pi_i = \frac{\exp(-\beta \epsilon_i)}{Z} \tag{1.2}$$

This is "similar" to the form of probability distribution for the interference model given by theorem 3.1. In fact, if $p = 1$ (only one class of requests) and we set $T = -1/k \log \rho$, the partition function of an interference model can be rewritten as:

$$Z = \sum_{A \in J} e^{-|A|/kT} \tag{1.3}$$

in complete analogy to the statistical mechanics model. Note that $\rho = 0$ corresponds to $T = 0$, and $\rho = 1$ corresponds to $T = \infty$, so that traffic increase is associated with a rise in "temperature". Thus, "temperature" measures the activity level of an interference system.

Pursuing the analogy further (see [Yemi83]), let us interpret the energy associated with a microstate. Define the following pair-interaction potential function $U(x, y)$ ([Ruel69, Pres74]) between nodes x, y of the interference graph

$$U(x,y) = \begin{cases} \frac{1}{2}\log\rho & \text{if } x = y \\ -\infty & \text{if } x \text{ is a neighbor of } y \\ 0 & \text{otherwise} \end{cases} \tag{1.4}$$

The potential represents a fixed "charge" of $\log\rho$ associated with an activity and an infinite repulsion among interfering agents. This fixed charge can be represented as $\log\lambda - \log\mu$. Thus, one can think that it reflects the energy of $\log\lambda$ gained by an arrival and the energy $\log\mu$ lost by a departure.

If we define the potential of the set of nodes A to be

$$\phi(A) = \sum_{x,y \in A} U(x,y) \tag{1.5}$$

the partition function of the interference model (equation 1.3) can be rewritten as follows

$$Z_N = \sum_{A \in J} e^{\phi(A)} \tag{1.6}$$

The potential $\phi(A)$ is similar to that associated with a lattice interaction model of a "hard balls" gas ([Ruel69]). In that model molecules are confined to the vertices of a lattice and the interaction between molecules depends only on

the distance between them. Pursuing this analogy, we can think of an activity in progress as a ball occupying some lattice space, thus excluding some other activities, namely those which interfere with the original one. This analogy will be used in deriving the concept of pressure for an interference system.

From the above discussion it follows that a physical system is completely described by the respective interaction potential. The steady-state distribution (and thus any macroscopic property) is given in terms of this potential. Statistical mechanics analyzes systems by identifying the microscopic interaction potential, computing the respective partition function, and deriving from it the macroscopical properties of the system. Much of the theory of statistical mechanics can be viewed as calculating or approximating the partition function for different interaction potentials. It is the ability to estimate the partition function that is the key to any system analysis in physics.

Let us pursue the analogy further. It is clear that the number of concurrent activities in an interference system is analogous to the energy of the microstate of a physical system. The average concurrency of a system "corresponds" to the average energy of a microstate. The global energy thus corresponds to the average number of concurrent transactions processed by the distributed system. Obviously, utilization corresponds to the probability of a non-zero energy microstate.

In statistical physics, entropy and pressure are defined in terms of the free energy function. Pursuing the analogy with physics further, we can define the Helmholtz free energy function

$$F = -kT\log Z_N \tag{1.7}$$

In physics, pressure P can be obtained from the relation $P = \frac{\partial F}{\partial V}$ where V is the volume of the system. What is an analogue of volume for an interference system ? To define the concept of volume for our model, let us pursue further the analogy to the "hard balls" gas ([Yemi83]). Consider a ***regular*** interference graph

with N vertices, each vertex having degree d. Assume that the number N of nodes is large, in fact so large that $\frac{\partial Z_N}{\partial N}$ is meaningful. Now consider an active node of the interference graph. The expected number of activities blocked by that node is λd. In thermodynamics, when deriving Van der-Waals equation of state, one thinks of this node as the center of a hard ball, creating an excluded volume in which other balls cannot exist ([Ruel69]). Therefore, by analogy with physics, λd denotes the *volume* occupied by an active node. The total number of such nodes (that is, the number of moles in gas) is $\frac{N}{\lambda d}$. If the volume of one mole of gas is kT, then the total volume is

$$V = \frac{NkT}{\lambda d} = \frac{-N}{\lambda d \log \rho} \tag{1.8}$$

With this definition of the volume, the pressure of the interference system is

$$P = -\frac{\partial F}{\partial V} = -\frac{1}{\log \rho}\frac{\partial \log Z_N}{\partial N}\frac{\partial N}{\partial V} = \frac{\lambda d}{Z_N}\frac{\partial Z_N}{\partial N} \tag{1.9}$$

As an immediate result, one can obtain the following "equation of state" for an interference system (similar to the Van der-Waals equation of state of a gas):

$$\frac{PV}{N} = -\frac{1}{\log \rho}\frac{\partial \log Z_N}{\partial N} \tag{1.10}$$

What does the above expression (1.9) for the pressure mean? What is $\frac{\partial \log Z_N}{\partial N}$? As before, consider an interference graph, which is regular of degree d and whose number of vertices is very large. Let G' be the graph obtained from G by adding one more vertex v of degree d. The number of independent sets with i-nodes will grow by $\alpha^i_{G'} - \alpha^i_G$. This number represents the number of configurations having i independent nodes, one of which is the added node. Therefore,

$$\frac{\partial \log Z_N}{\partial N} = \frac{1}{Z_N}\sum(\alpha^i_{G'} - \alpha^i_G)\rho^i \tag{1.11}$$

is the probability that the added node is busy. Hence, $\lambda d \frac{\partial \log Z_N}{\partial N}$ is the rate at which an activity generated by a neighbor of v is blocked by v. Therefore, the pres-

sure of an interference system represents a measure of the average rate of blocking experienced by activities in progress.

Consider now a general interference graph. Assume that there are n_1 nodes of degree d_1, n_2 nodes of degree $d_2, \ldots$, n_p nodes of degree d_p. Nodes of the same degree represent molecules of the same gas, and so the model is analogous to that of a mixture of gases. The volume occupied by nodes of degree i is clearly $V_i = \frac{n_i kT}{\lambda_i d_i}$. Total volume $V = V_1 + V_2 + \cdots + V_p$. The pressure is then

$$P = -\frac{\partial F}{\partial V} = kT\frac{\partial \log Z_N}{\partial V} = \frac{kT}{Z_N}\frac{\partial Z_N}{\partial V} = \frac{kT}{Z_N}\sum_{i=1}^{p}\frac{\partial Z_N}{\partial n_i}\frac{\partial n_i}{\partial V} \tag{1.12}$$

But $\frac{\partial V}{\partial n_i} = \frac{kT}{\lambda_i d_i}$ and thus, $\frac{\partial n_i}{\partial V} = \frac{\lambda_i d_i}{kT}$. Therefore, the following expression for the pressure is obtained:

$$P = \sum_{i=1}^{p}\frac{\lambda_i d_i}{Z_N}\frac{\partial Z_N}{\partial n_i} \tag{1.13}$$

Just as in the previous case, from this we can write the "equation of state". What is $\frac{\partial \log Z_N}{\partial n_i}$? As before, let G' be an interference graph obtained from G by adding a vertex of degree n_i. Using the arguments above, we can show that $P_i = \lambda d_i \frac{\partial \log Z_N}{\partial n_i}$ is a measure of averaged blocking from nodes of degree d_i. Hence, we can rewrite (11) as

$$P = P_1 + P_2 + \cdots + P_p \tag{1.14}$$

This is what one intuitively expects! The average rate of blocking experienced by activities is the sum of the rates of blocking from different types of nodes. The above equation (1.14) corresponds to Dalton's gas law in physics: the pressure exerted by a mixture of gases equals the sum of pressures exerted by those gases

taken separately. In our model, an analogous conclusion can be drawn: The blocking experienced by an activity from all other activities of classes $1, \ldots, p$ equals to the sum of blocking experienced from agents of each class - which is intuitively what one expects !!!

Finally, the entropy S is defined (up to an additive constant) as follows

$$S = \frac{1}{T}(E - F) \tag{1.15}$$

One of the fundamental laws of statistical mechanics is that the entropy of an isolated system in equilibrium cannot decrease. An analogous statement for the interference model is the H-theorem for reversible Markov chains ([Bene63]). It is possible to show that given an interference graph G and the concurrency E_σ for the agents of class σ, the equilibrium probability $\pi(A)$ given by theorem 3.1. is the *unique* distribution over the independent subsets of G maximizing the entropy S subject to the constraint that the average concurrency of class σ is E_σ. The proof is a simple exercise in the Lagrange's method of constrained optimization.

APPENDIX 2

THE CONCEPT OF ENSEMBLE EQUIVALENCE

In this appendix we introduce the concept of Gibbs canonical ensembles and discuss the concept of their equivalence. This equivalence is the main idea of the canonical approximation. We will continue to use the analogy between an interference system and a physical system.

To start, consider a large physical system. It is clear that for a given macrostate (e.g. given energy), such a system is equally likely to be in any one of a large number of microstates. As time passes, the system continuously switches over from one microstate to another. Over a reasonable amount of time all that one observes of the system is a behavior "averaged" over the variety of microstates. One may therefore consider at single period of time a large number of identical systems ("mental" copies). This collection is called a *microcanonical ensemble.* It is assumed that these systems are characterized by the same macrostate as that of the original system but are in different possible microstates. Thus, one would intuitively expect that the average behavior of this *ensemble* would be identical with time-averaged behavior of the given system. In other words, the thermodynamical averages of a physical system (e.g. energy, pressure) can be derived from the corresponding ensemble. Similarly, the performance averages of an interference model (e.g. average concurrency) can be derived from the corresponding ensemble.

The idea of analyzing a system from the corresponding ensemble was developed by Gibbs ([Path84]). For reasons to be explained below, he introduced three kinds of ensembles:

- **Microcanonical** - the macrostate of the system is defined by the number of molecules N, the volume V, and a range of energy values $E - \epsilon < E < E + \epsilon$
- **Canonical** - an infinite number of "mental" copies of the original system which

can share the energy (but cannot share particles).

- **Grand Canonical** - the same as canonical, but now the number of particles N as well as the energy E can vary.

In the study of a microcanonical ensemble, the basic problem consists in determining the number of distinct microstates accessible to a given system. From the asymptotic expressions for these numbers, the complete thermodynamics of the system can be derived in a straightforward manner. However for most physical systems, the mathematical problem of determining these numbers is quite complex. Also, it is difficult to guarantee that the energy is in a particular range since the energy of a system is hardly ever measured. A better alternative is to speak of a fixed temperature. This naturally leads to the concept of a canonical ensemble - a collection of M identical systems sharing the total energy MU. Thus, U stands for the average energy per system. Clearly, a canonical ensemble of interference systems is a collection of M systems of the same size N with the total concurrency MU

In a canonical ensemble, the systems are in thermal contact with each other and can share the energy. Thus, the energy E of a particular physical system in such an ensemble is necessarily a variable; in principle, it can take on any value from 0 and MU. Similarly, the average concurrency of an interference system in the corresponding canonical ensemble can take any admissible value. We can ask the following questions:

(1) What is the probability P_r that a physical system is found in any one of the states characterized by the energy E_r? It can be shown ([Path84]) that it must be of the form

$$P_r = \frac{\exp(-\beta E_r)}{Z_N} \tag{2.1}$$

The same expression would be obtained if we asked the corresponding question for an interference system. As we would expect, the answer (equation 2.1) is of the same form as the solution to the interference model (theorem 3.1.).

(2) How likely it is to find a system in the ensemble whose energy (concurrency) deviates "significantly" from U?

To answer this, we can analyze the manner in which energy (concurrency) is distributed among the different systems of the canonical ensemble. To do so, let us treat the energy (concurrency) E as a continuous variable (assume that the system is large enough for this approximation to hold). Let $\langle\ \ \rangle$ denotes the expected value of a random variable and let $(\Delta E) = \langle E^2 \rangle - \langle E \rangle^2$. The probability that the system has energy (concurrency) $E = E_r$ is

$$P(E_r) = \frac{\alpha_N^{E_r} \exp(-\beta E_r)}{\sum_r \alpha_N^{E_r} \exp(-\beta E_r)} \tag{2.3}$$

Here, $\alpha_N^{E_r}$ denotes the number of microstates (configurations) corresponding to energy (concurrency) E_r. Hence

$$P(E)dE \propto \exp(-\beta E)\alpha_N^E dE \tag{2.4}$$

The probability density $P(E)$ of the energy (concurrency) is given by the product of two factors: the "Boltzmann"-type factor which monotonically decreases with E, and the density of states, which monotonically increases with E. The product therefore passes through some extremum. One can show ([Path84]) that for a system in a canonical ensemble, this extremum must coincide (for large systems) with the average energy (concurrency) $U = \langle E \rangle$.

Expanding the logarithm of the probability density $P(E)$ around the value U, we obtain

$$\log[\exp(-\beta E)\alpha_N^E] = \left(-\beta U + \log(\alpha_N^E)\right) + \frac{1}{2}\frac{\partial^2}{\partial E^2}\log[\exp(-\beta E)\alpha_N^E](E-U)^2 + \cdots$$

Therefore,

$$P(E) \propto \exp(-\beta E)\alpha_N^E \approx e^{-\beta(U-TS)} \exp\left[\frac{(E-U)^2}{2\langle(\Delta E)^2\rangle}\right] \tag{2.5}$$

Hence, the distribution of energy (concurrency) in the canonical ensemble is of the Gaussian type with mean value U and the dispersion $\sqrt{\langle(\Delta E)^2\rangle}$. For a large number of systems, it can be shown that this dispersion becomes smaller and smaller. Therefore, for a large ensemble (large M) we have an extremely sharp distribution and as $M \mapsto \infty$ the distribution approaches a delta-function ! Because the canonical ensemble contains systems of all energies, the fact that the energy is sharply peaked around U shows that canonical and microcanonical ensembles are equivalent. A similar conclusion holds for canonical and microcanonical ensemble of an interference system.

The reason for introducing the grand canonical ensemble in statistical physics is that for a number of physical and chemical problems canonical ensemble formalism turns out to be severely limited. The motivation to develop the concept of a grand canonical ensemble is of the same nature as that which led from the microcanonical to the canonical ensemble. It comes from the realization that not only the energy, but also the number of particles can only be measured with difficulty. For the grand canonical ensemble, we take a collection of M physical systems of N particles just as for the canonical ensemble, but now the systems are allowed to exchange particles. The grand canonical ensemble for an interference system of size N is a collection of M such systems, but now we assume that they can "exchange" some of the resources between themselves. For example, take a grand canonical ensemble of M database systems, each consisting of N items. The "exchange" of resources in this context means that the databases may have variable number of items, but the total number of items in the ensemble is MN.

Mathematically, the grand canonical distribution is simply the generating function for the partition function Z_N, that is

$$Z_G(t) = \sum_{N=0}^{\infty} Z_N t^N \tag{2.6}$$

In physics, this function is called the ***grand partition function*** and the variable t is called the ***fugacity*** ([Path84]).

Since the total number of particles in the ensemble is MN, the number of particles in a system can theoretically vary from 1 to MN. However, for large M, one would expect that a "typical" system in the grand canonical ensemble has N particles. One can therefore consider fluctuations in the number of particles N in the grand canonical ensemble, but the fluctuations will be very small. This can be easily established by the law of large numbers ([Path84]).

This consideration leads us to expect that in the series defining the grand partition function (equation 2.6) there is a maximal term corresponding to the most probable number of particles, namely N, whose contribution to $Z_G(t)$ is overwhelmingly large compared with those from other terms.

Based on the above consideration, we can find the relation between the partition function Z_N and the grand partition function $Z_G(t)$. This was shown first by Darwin and Fowler ([Darw22]) using the asymptotic method of steepest descent. The method allows us to compute Z_N from the grand partition function $Z_G(t)$. In chapter 3 it was shown that using this method, the partition function Z_N can be represented as follows:

$$Z_N = F_N(t_N)[1 + \epsilon_N] \tag{2.7}$$

where the relative error ϵ_N decreases with N and is typically of $O(1/N)$. Here t_N is the unique positive point at which the derivative of $Z_G(t)/t^{N+1}$ (with respect to t) vanishes, and $F_N(t_N)$ is related to the second derivative of the grand partition function at t_N.

In statistical mechanics, this method establishes the equivalence of canonical and grand canonical ensembles. Therefore, all three ensembles - microcanonical, canonical and grand canonical are equivalent and give the same "thermodynamics". Similarly, this means that one obtains the same performance measures (e.g. average concurrency) for an interference system in any of these ensembles. The performance

measures are expressed in terms of the derivatives of Z_N with respect to some parameters (e.g. load), Therefore, the not only it gives a good approximation to Z_N, but the derivatives of Z_N are also approximated by the corresponding derivatives of $F_N(t_N)$.

Therefore, the method gives a closed-form approximation to Z_N and gives a simple way to approximate the corresponding derivatives of Z_N. This is important: it is not true, in general, that if two functions are very close by their function values at any point, then so are their derivatives. For a example, take $f_N(x) = x$ and $g_N(x) = x + \sin(N^2x)/N$. Then $|f_N(x) - g_N(x)| \leq 1/N$. On the other hand, $|f'_N(x) - g'_N(x)| = |N\cos(N^2x)$ is unbounded as $N \mapsto \infty$.

It appears that in order to evaluate the grand partition function $Z_G(t)$ for an interference model, we have to pass through the routine of calculating the partition functions Z_N. In principle, this is true. In practice, however, an explicit evaluation of the partition function is extremely difficult, whereas evaluation of the grand partition function can often be done with relative ease. The formalism of the grand canonical ensemble is of no value if we can evaluate the partition function explicitly. Therefore, assuming that the grand partition function $Z_G(t)$ is easier to evaluate in closed form than Z_N, we can use the method similar to that used in physics to approximate the partition function Z_N from $Z_G(t)$ and then calculate the performance averages. We call this **Canonical Approximation**. The method is valid for product form stochastic models, not necessarily described by an interference graph (e.g. closed queueing networks). It is presented in detail in Chapter 3.