University of London
Imperial College of Science, Technology and Medicine
Department of Computing

Models of Circuit-Switched Interconnection Networks

Nareshkumar Maganlal Patel

Submitted in part fulfilment of the requirements for the Doctor of Philosophy degree in Engineering of the University of London, and the Diploma of Imperial College of Science, Technology and Medicine, May 1989.

Abstract

Interconnection networks (INs) provide concurrent communication between processing elements in parallel machines. A single crossbar can be used for this purpose but it is difficult to fabricate in large dimensions, and so it is common to use a multi-stage IN (MIN) which is itself constructed by linking a number of crossbars together. For all MIN protocols this causes contention for internal links as well as destination links. Moreover, when the network traffic becomes concentrated on one particular destination address, the effect of the hot-spot propagates back over all shared internal links causing throughput to drop sharply. This fundamental design problem is investigated for the circuit switching protocol in which partial paths are held. For this protocol, the switching network is a passive resource and so queueing networks which include them do not have product-form solutions. However, under mild approximating assumptions, hierarchical decomposition techniques may be applied to obtain a flow-equivalent server for a parallel bank of devices transmitting through a switching network. Validation of this model with respect to simulation indicates a high degree of accuracy. Simulation models of MINs with more complex protocols are also described. In particular, a novel circuit switching protocol specifically for optical networks is proposed and this gives better performance than the more basic protocols.

Acknowledgements

First and foremost, I would like to thank my supervisor Peter Harrison who, through his careful guidance, sound advice and encouragement has sustained me throughout my research and made this thesis possible. I am also indebted to all past and present members of the Functional Programming Group at Imperial College for providing such an interesting and stimulating environment to work in. In particular, I owe special thanks to the head of the group, John Darlington, for his support and encouragement. Many thanks are also due to Graham Cope of British Telecom for the many fruitful discussions about network modelling.

Finally, I would like to thank my wife, Hina, who has been very supportive and understanding throughout the course, and my parents to whom I dedicate this thesis.

The research was conducted under both the Flagship Project (Alvey/SERC) and the Performance Modelling of Parallel Architectures Project (SERC).

Contents

Introduction 9

Chapter 1: Survey of Switching Networks and Modelling Techniques 12

1.1 Interconnection Networks for Parallel Processing Systems 13
1.1.1 Introduction to Parallel Computer Systems 13
1.1.2 Interconnection Networks for Multiprocessors 15
1.2 Multi-stage Interconnection Networks 18
1.2.1 Topology of Multi-stage Interconnection Networks 18
1.2.2 Banyan Networks 18
1.2.3 Delta Networks 20
1.2.4 Fault Tolerant MINs 24
1.2.5 Operational Characteristics of Switching Networks 24
1.3 Models of MINs 25
1.3.1 Blocking in MINs 25
1.3.2 Synchronous MIN Models 26
Synchronous Crossbars 26
Synchronous, Circuit-switched MINs 27
Synchronous, Packet-switched MINs 28
1.3.3 Asynchronous Multi-stage Interconnection Network Models 28
Asynchronous, Packet-switched MINs 29
Asynchronous, Circuit-switched MINs 29
1.4 Modelling Circuit-Switched Networks 29
1.4.1 Markov Model of Circuit-switched Networks 30
1.4.2 Limiting Regimes 31
1.4.3 Erlang's Fixed-Point Approximation 31
1.4.4 Circuit-switched MIN Model using Erlang's Fixed-Point 32

Chapter 2: Representation of Interconnection Networks in Queueing Models 33

2.1 Flow Equivalent Server for the Interconnection Network 34
2.2 Underlying Markov Process 35
2.3 Underlying Birth and Death Process 38
2.4 Virtual Bank of Infinite Servers 42

Chapter 3: Models of a Uniform Crossbar Switch 46

3.1 Markov Model for a Full Crossbar 47
3.2 Markov Model for a Uniform Crossbar 49
3.2.1 Transition Probabilities 49
3.2.2 Global Balance Equations 50
3.2.3 Constrained Markov Process 51
3.2.4 Exact Result for a 2 by 2 Crossbar 52

3.3 Probability Model for a Uniform Crossbar 52
3.4 Throughput Function for a Uniform Crossbar 54
3.5 Static Probabilities for a Uniform Crossbar 55

Chapter 4: Models of the Non-uniform Crossbar Switch 59

4.1 Markov Model for a Biased Crossbar 60
4.1.1 Transition Probabilities 61
Notation for Transition Probabilities 62
Transition Probability Functions 63
4.1.2 Global Balance Equations 65
4.1.3 Derivation of the Birth-Death Process 68
4.1.4 Constrained Process 69
4.2 Approximate Probability Model for a Non-uniform Crossbar 69
4.2.1 Static Probabilities for a Non-uniform Crossbar 70
4.2.2 Transformed Static Probabilities 70
4.3 Non-uniform Routing with Different Pin Release Times 71
4.3.1 Constrained Process 72
4.3.2 Analysis of a 2 by 2 Crossbar 73
4.3.3 Output Pin Utilisation in a Biased Crossbar 74

Chapter 5: Uniform Delta Networks 77

5.1 Exact Analysis of Delta Networks 78
5.1.1 Definitions 79
5.1.2 A Crossbar in Equilibrium 79
5.1.3 A Delta Network in Equilibrium 80
5.2 Analysis of Delta-2 Networks 81
5.2.1 The Exponential Assumption and the Arriving Observer Property 81
5.2.2 A 2 by 2 Crossbar in Equilibrium 82
5.2.3 Recursive Analysis of Delta-2 Networks 83
5.2.4 Saturated Delta Networks 86
5.3 Approximate Analysis of General Delta Networks 87
5.3.1 Definitions and Assumptions 87
5.3.2 Recursive Analysis of General Delta Networks 88
5.3.3 Saturated Delta Networks 91

Chapter 6: Hot-spot Contention in Delta Networks 95

6.1 Terminology 96
6.2 Reachability 98
6.3 Non-uniform Traffic Model 99
6.3.1 Network Routing Probabilities 100
6.3.2 Switch Routing Probabilities 101
6.3.3 Relation between Network and Switch Routing Probabilities 101
6.3.4 Induced Routing Probabilities 102
6.4 Path Blocking Probabilities 102
6.5 Exact Analysis of Biased Delta Networks 104
6.5.1 Definitions 105
6.5.2 Biased Crossbars 105

6.5.3 Biased Delta Networks 106
6.6 Analysis of Biased Delta-2 Networks 107
6.6.1 Simplifying Assumptions 107
6.6.2 Biased 2 by 2 Crossbars 108
6.6.3 Recursive Analysis of Biased Delta-2 Networks 109
6.7 Approximate Analysis of Biased Delta Networks 111
6.7.1 Definitions and Assumptions 111
6.7.2 Approximate Recursive Analysis 112
6.8 Fixed-point Equation for Release Time Ratios 113
6.8.1 Fixed-point Equation 114
6.8.2 Iterative Method 115

Chapter 7: Towards a better Protocol for Switching Networks 117

7.1 Circuit Switching Protocols 118
7.2 The Delta Network Topology 119
Definitions 120
Adjacent Input Address 120
Clashing of Two Paths 121
7.3 The Combined Release-Hold Protocol 121
7.4 Choosing the Final Releasing Stage 123
7.5 Unblocking Time of Blocked Tasks 124
7.5.1 Worst Case Analysis 125
7.5.2 Average Case Analysis 126
7.5.3 Saturated Networks 127
7.6 Implementation of the Release-Hold Protocol 127

Chapter 8: Validation and Numerical Results 129

8.1 Standard Traffic Schemes 130
8.2 Validation of the Analytical MIN Models 131
8.2.1 Validation of Model A against Simulation 131
8.2.2 Validation of Analytical Model B 133
8.3 Numerical Results of the Analytical MIN Models 134
8.3.1 16-way Interconnection Networks 134
8.3.2 Delta Networks with a Hot-spot 135
8.4 Simulation Results for the NSD Release-Hold Protocol 137
8.4.1 NSD Protocol with Standard Traffic Schemes 137
8.4.2 Varying the Last Releasing Stage 140

Chapter 9: Conclusion and Future Work 142

Appendix A: Exact Analysis of a 2-stage Delta-2 Network 148

A.1 State Space 148
A.1.1 Definitions 148
A.1.2 Symmetric States 149
A.1.3 Aggregated States 150
A.2 Markov Process 150
A.2.1 Balance Equations 151
A.2.2 Steady State Distribution 151

A.2.3 Derived Distributions 152
A.3 Analysis of an Arriving Observer 154
A.3.1 Notation 154
A.3.2 State seen by an Arriving Observer 155
A.4 Summary of Results 156

Appendix B: Exact Results for a 2 by 2 Crossbar 158

B.1 Definitions 158
B.2 Two by Two Crossbar Results 158

Appendix C: Conversion of Decimals into Rationals 160

References 162

List of Illustrations

Figure 1.1 IN with a Static Topology 15
Figure 1.2 IN with a Dynamic Topology 16
Figure 1.3 The 4-way crossbar switch 18
Figure 1.4 The SW Banyan Structure 21
Figure 1.5 Labelling a Bidelta Network 22
Figure 1.6 Delta Network Topologies 24
Figure 1.7 An Example of a Circuit-switched Network 31
Figure 2.1 The Short-circuited Sub-network defining the FES Rate 37
Figure 2.2 VBIS Servers with an Interconnection Network 44
Figure 3.1 The Full Crossbar 49
Figure 4.1 State of a Biased Crossbar 61
Figure 5.1 Pin Utilisations in a b by a Crossbar 81
Figure 5.2 Recursive Structure of s-stage Delta-2 Networks 85
Figure 6.1 Pin Classes in a 8-way Delta-2 Network 98
Figure 6.2 Decode Tree of Topmost Network Input 101
Figure 7.1 Pin Addresses in a Delta Network 121
Figure 7.2 Task States in the NSD Release-Hold Protocol 124
Figure 7.3 Fan-in Fan-out Structure of the Delta Network 127
Figure 8.1 A Performance Comparison of 16-way Networks 136
Figure 8.2 Graph of Throughput vs Hot-spot Routing Probability 137
Figure A.1 Symmetric States in a 2-stage Network 150
Figure A.2 Pin Utilisations in a 2-stage network 157

List of Tables

Table 7.1 Unblocking Times for Blocked Tasks 128
Table 8.1 Saturated, Uniform Delta Network Results 132
Table 8.2 Saturated, Non-uniform Delta Network Results 133
Table 8.3 Non-saturated, Uniform Delta Network Results 133
Table 8.4 Non-saturated, Non-uniform Delta Network Results 133
Table 8.5 Throughput Results for Models A and B 134
Table 8.6 Saturated 2-stage Network Results 139
Table 8.7 Non-saturated 2-stage Network Results 139
Table 8.8 Saturated 6-stage Network Results 139
Table 8.9 Non-saturated 6-stage Network Results 140
Table 8.10 Effect of the Last Releasing Stage 141

Introduction

The complexity of large-scale parallel computer architectures is such that their effective design requires guidance from some form of predictive modelling prior to construction, and analytical models based on queueing network analysis provide a powerful tool for this purpose. In many parallel machines, for example ALICE [DaRe81, HaRe86], interconnection networks (INs) are required by data transmission devices (DMA processors) to provide concurrent communication between the many processing and memory elements. It is normally necessary for any element of one type to be able to communicate with any of the other type, e.g. each processor must have access to any memory. In principle, the interconnection can be provided by a crossbar switch in which the contention for destination addresses is relatively easy to model as a degradation of the transmission service rates. However, in practice, large crossbars cannot be fabricated cost-effectively and a multi-stage interconnection network (MIN) is normally used, comprising a number of stages of parallel banks of small crossbars. This provides the required connectivity, but the sharing of internal links by different paths across the network introduces extra contention.

Typically, the memory modules in parallel machines are not uniformly utilised by the processing elements (PEs). For example, a data structure that is frequently accessed by all the PEs may be stored entirely in one part of the distributed memory. Even if such cases are avoided, the use of shared locks have a similar effect. When the network traffic becomes concentrated on one particular destination link, the effect of this *hot-spot* propagates back over all shared internal links, thereby adversely affecting tasks which are not requesting the hot-spot. Consequently, the network throughput drops sharply for all MIN protocols.

There are two basic transmission protocols which require entirely different model types - circuit switching and packet switching. In the case of circuit switching, the network is modelled as a *passive* resource in a queueing network; in order to operate a data-transmission server must hold a path comprising internal links and crossbars at each stage through to the destination addressed by the task at the front of its queue. A passive resource (such as a set of links across the network) is one that must be held by a task in order to permit some associated *active* resource (such as a data transmission

device) to provide service to that task, executing units of applied workload at its particular service rate; a more common example is computer memory. In the case of packet switching, such paths need not be held throughout a message's transmission, and buffered internal crossbars can be modelled as conventional servers since their transmission times are non-negligible because of the intelligence required to provide the buffering. However, similar problems arise if the buffers become exhausted and cannot be reasonably assumed infinite. Assuming infinite buffers, if the departure processes from the output pins of each stage can be determined in terms of the stage's arrival processes, the arrival processes to the next stage can be derived by simply permuting the outputs according to the interconnection topology. Shuffle-exchange is therefore a particularly convenient representation of this topology, since the permutation is then the same between every pair of stages, [Si85]. Certain approximations regarding the internal arrival processes at successive stages simplify the analysis, providing an efficient algorithm, and such an analysis is given by Mitra and Cieslak [MiCi87]. An analysis of a packet-switched crossbar in discrete time is given in [PoHa88] and an approximate analysis of a buffered MIN with non-uniform routing is given in [GaHu88].

However, we consider the alternative protocol - circuit switching - in which the interconnection network is a passive resource, providing no direct service in the sense of advancing tasks holding network links towards their completion. Our approach to this problem relies on the *recursive structure* of the MINs we consider which is in contrast to analogous research reported by Kelly [Ke86] on telephone networks. In either case, we could in principle consider all routes through the network individually and solve the associated Markov process directly to obtain the Erlang loss formula - assuming, that is, that partial paths are not held but released on a collision, resulting in lost transmissions. Because of the large number of routes, such a direct solution is computationally intractable. Kelly's analysis relies on the network behaving as if the steady state probabilities that each link is blocked were independent. This property is shown to hold asymptotically as link capacities and network traffic approach infinity jointly so that the traffic offered per link remains constant. In fact the approximation is very accurate for telephone networks because of the large capacity links and large number of nodes. Our analysis differs from this in that links have *small* capacities (viz. one) and also that partial paths are held during path building. Moreover, partly as a result of the former difference, there is a strong dependence between individual links which do not behave remotely as if they were independent, and it is our recursive analysis which captures this dependence in a simple way. In this sense, our analysis

addresses the opposite end of the spectrum of switching networks and their control protocols.

In the first chapter, we survey some of the switching networks commonly used in multiprocessor systems, describe their operational characteristics, and discuss the problems encountered in modelling such networks. Our general approach to modelling a queueing sub-network that represents a collection of DMA servers together with a circuit-switched MIN is presented in Chapter 2. First we apply decomposition techniques to obtain the service rate of a *flow equivalent server* (FES) by determining the throughput of the short-circuited sub-network as a function of its population, according to the method of [CHW75]. This throughput is obtained by defining a simple Markov process, under appropriate assumptions, in which some function, μ_n, for throughput is assumed to be known when the number of active *input* pins, n, is known - i.e. the number of DMA servers currently wishing to transmit. By considering a Markov process in Chapter 3, we derive an expression for μ_n when the interconnection network is a *full crossbar switch* with uniform routing. The uniform crossbar model is extended in Chapter 4 so that non-uniform traffic patterns can be investigated. In Chapters 5 and 6, we describe respectively, analytical models of uniform and non-uniform delta networks by exploiting their recursive structure and using crossbar models as the base case. A novel switching protocol for MINs is modelled in Chapter 7 and the degree of degradation introduced by such networks on the DMA servers' throughput is illustrated by numerical predictions in Chapter 8. Our conclusions and suggestions for future work are laid out in the final chapter.

Chapter 1

Survey of Switching Networks and Modelling Techniques

Outline

1.1 Interconnection Networks for Parallel Processing Systems
- 1.1.1 Introduction to Parallel Computer Systems
- 1.1.2 Interconnection Networks for Multiprocessors

1.2 Multi-stage Interconnection Networks
- 1.2.1 Topology of Multi-stage Interconnection Networks
- 1.2.2 Banyan Networks
- 1.2.3 Delta Networks
- 1.2.4 Fault Tolerant MINs
- 1.2.5 Operational Characteristics of Switching Networks

1.3 Models of MINs
- 1.3.1 Blocking in MINs
- 1.3.2 Synchronous MIN Models
  - Synchronous Crossbars
  - Synchronous, Circuit-switched MINs
  - Synchronous, Packet-switched MINs
- 1.3.3 Asynchronous Multi-stage Interconnection Network Models
  - Asynchronous, Packet-switched MINs
  - Asynchronous, Circuit-switched MINs

1.4 Modelling Circuit-Switched Networks
- 1.4.1 Markov Model of Circuit-switched Networks
- 1.4.2 Limiting Regimes
- 1.4.3 Erlang's Fixed-Point Approximation
- 1.4.4 Circuit-switched MIN Model using Erlang's Fixed-Point

Switching networks have long been of interest to telephone engineers, computer communication network designers and, more recently, architects of parallel processing systems. This chapter describes various types of interconnection networks for parallel machines. For large-scale parallel computer systems, multi-stage interconnection networks (MINs) appear to provide the most cost-effective solution. The structure of the most well-known MINs is described in some detail before the results of selected MIN models with various operational characteristics are presented. The final section of the chapter describes a modelling technique for circuit-switched networks. However, for MINs, this approach is impractical because the underlying assumptions are not valid. Nevertheless, we include it as an example of a general modelling technique which is very accurate for many large circuit-switched networks.

1.1 Interconnection Networks for Parallel Processing Systems

At the heart of any parallel processing system lies some form of interconnection network (IN) which provides interprocessor communication. We present a popular classification scheme for computing systems and describe various types of INs associated with parallel systems. This brief introduction to parallel systems attempts to show the importance of IN design and to outline some of the choices available.

1.1.1 Introduction to Parallel Computer Systems

Parallel computer architectures may be classified according to the number of instructions and data streams that are accepted at the same time. In this classification, due to Flynn [Fl66], there are four categories: SISD (Single Instruction stream, Single Data stream), SIMD (Single Instruction stream, Multiple Data stream), MISD (Multiple Instruction stream, Single Data stream), and MIMD (Multiple Instruction stream, Multiple Data stream). Traditional uniprocessor systems are SISD machines and many current systems exploit pipelining techniques to increase execution speed by overlapping execution of instructions. SIMD machines or array processors have a number of data streams which can be simultaneously read by a number of processors when a single instruction is issued by the control unit. Each processor has a tag which can be set to inhibit reading of the next data item so that selected processors may be turned on depending on the program being executed. For certain applications, SIMD machines are significantly faster than uniprocessors. Examples of SIMD machines include the Massively Parallel Processor (MPP) [Po85] and Illiac IV [Bo72]. The third category, MISD, is only of theoretical interest and has been dubbed as impractical by many computer architects since it is unnatural for different instructions to simultaneously act on the same data. In a MIMD architecture, there are a number of

processors, each with its own instruction stream and data stream. The processors behave independently until one processor needs to communicate or cooperate with another. The interprocessor communication is provided by some kind of IN.

Parallel architectures can also be classified according to the structure or topology of the IN. An IN is said to have a *static topology* if links between processors are static, i.e. cannot be reconfigured to connect to another processor. The structure of the IN is normally some regular pattern, the simplest example being the 1-D linear array. More interesting static topologies include 2-D structures like the ring, star, tree, near-neighbour mesh, and systolic array. Many of these INs provide the optimal structure for mapping certain application program structures onto real processors. Although this gives orders of magnitude speed-ups in specific application areas such as image processing, matrix multiplication, LU decomposition (for solving simultaneous linear equations), the fixed IN structure is not always appropriate in general. Thus another important requirement of an IN topology is the ability to simulate other structures. For this reason, higher dimensional structures such as the hypercube have also become popular. Thinking machines Inc.'s Connection machine [Hi85] is a well-known example of a SIMD architecture with a hypercube structure. The class of homogeneous parallel computer architectures with a large number of processing elements (PEs) and a static topology is shown in Figure 1.1 where processors are labelled P and memories are labelled M. These parallel architectures are sometimes also called *multicomputers*.

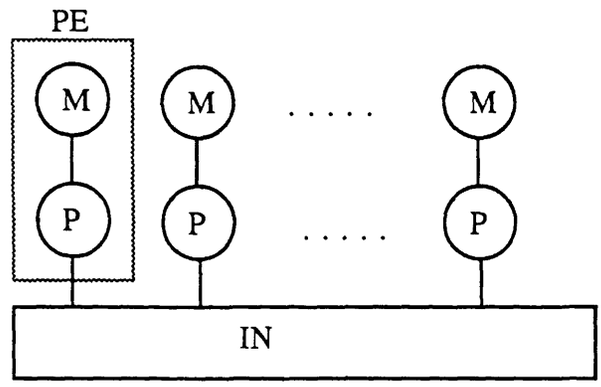


Figure 1.1 IN with a Static Topology

In contrast, the class of homogeneous parallel computer architectures with a large number of PEs and memory modules (MM) and a dynamic IN topology is shown in Figure 1.2. The dynamic IN can make connections from any PE to any MM and, in principle, no MM is favoured by any PE. An example of this type of MIMD computer is the NYU Ultracomputer [Go83].

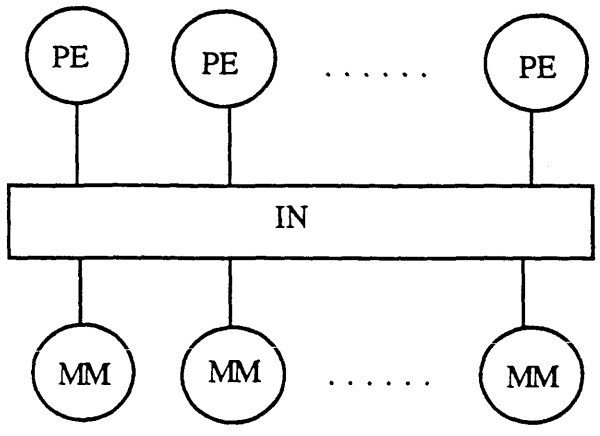


Figure 1.2 IN with a Dynamic Topology

Generally in multiprocessing systems, the processors can communicate and cooperate at different levels in solving a given problem, i.e. by sending messages or by sharing memory. It is generally agreed that for interprocessor communication, message passing is slower than sharing memory. If there is relatively little processor interaction via shared memory the parallel system is said to be *loosely coupled.* Thus in loosely coupled multiprocessors, the processors have large local memories from which most of its instructions and data are obtained and communication between processors is achieved by sending messages across an IN. In this case, full connectivity is not essential as the processors spend little time communicating with each other and an IN with a static topology is suitable.

Parallel systems are said to be *tightly coupled* if there is a lot of processor interaction via shared memory. Thus in tightly coupled multiprocessors, the processors communicate through shared main memory and the IN provides full connectivity between processors and memory. Usually a small local memory or cache is attached to each processor to improve overall performance. Essentially, the speed of the machine is restricted by the memory bandwidth and hence the IN topology. An IN with a dynamic topology is most suitable. ALICE [HaRe86] is an example of a tightly coupled multiprocessor system that is designed to execute functional programming languages [FiHa88].

1.1.2 Interconnection Networks for Multiprocessors

Both static and dynamic IN topologies can be used for interprocessor communication in multiprocessors. Here we present some dynamic INs specifically for shared-memory multiprocessors. In these architectures the sharing of main memory naturally causes contention and so the IN design must be efficient, otherwise the processors spend most of their time communicating and doing little real work. We assess the relative merits of

each type of IN and conclude that the multi-stage IN appears to provide the most cost-effective means of implementing interprocessor communication for large-scale shared-memory multiprocessors.

Shared-memory multiprocessor systems require some kind of switch to connect their processing elements and memory modules, and being a unique shared resource in the parallel machine, the switch is a potential source of contention. The simplest type of switch for this purpose is the common bus which connects to every element. However, the bus derives its simplicity at the expense of increased complexity in each component that it connects. For example, bus request and receive pins are needed by each element, together with several others for control and synchronisation, and non-trivial logic is necessary to implement the bus's protocol. This increases the overall cost of a large-scale multiprocessor and, moreover, each additional processor that is added to the bus is faced with more contention and so delivers a diminishing increment in performance.

In order to accommodate more processors without increasing contention, the common bus can be extended by using multiple common buses. Multiple bus INs are simple yet fault tolerant and suitable for all algorithms. If there are N processors and M memories a B-bus IN has cost which varies as $O(B(N+M))$. Despite the inherent disadvantages of the common bus, multiple bus INs are good for medium-scale parallel systems because the number of buses can be increased until reasonable performance is attained.

At the other end of the spectrum, the switch might be a full crossbar which provides parallel communication between any number of distinct pairs of processors connected through it. For example, a crossbar may have two types of links – one for processors and the other for memory elements – and a way of dynamically connecting any pair of the two types. This allows parallel access to the shared memory provided that there are no *memory conflicts*, i.e. two or more processors attempting to access the same memory module.

An $a \times b$ crossbar can switch any of its a input pins to any of its b output pins on the appropriate clock cycle, so that the only contention possible is for the destination processors, i.e. is internal in the switch for the output pins. We therefore have the configuration shown below in Figure 1.3 when $a = b = 4$.

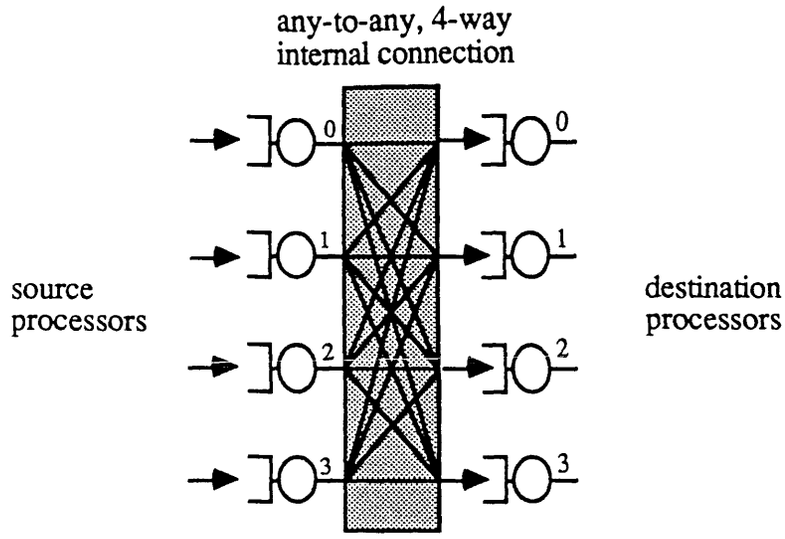


Figure 1.3 The 4-way crossbar switch

Each additional processor that is added to the crossbar requires a new parallel link to each of the output pins. Thus for a $N \times N$ or N-way crossbar the complexity and cost grow as $O(N^2)$ and the very large crossbars required for large-scale multiprocessors would become hopelessly complex and expensive. Moreover, large crossbars are physically infeasible to construct using contemporary electronic devices.

We can interconnect a matrix of crossbars to form a *multi-stage interconnection network* (MIN) which has a number of levels (or stages) such that the output pins of crossbars in one stage are connected to the input pins of crossbars in the next stage. This provides the same connectivity as a full crossbar (any input pin can be dynamically connected to any output pin if there is a free path) but introduces contention for internal links (*path conflicts*) on top of the memory conflicts. The cost of an N-way MIN grows almost linearly, $O(N \log N)$, compared with $O(N^2)$ for the equivalent crossbar. There are several types of MINs, many with similar structure and properties, but here we will consider *banyan networks* which are also called blocking networks because of their path conflicts. However, there are also MINs like the Benes network [Be65] in which all permutations can be realised by re-arranging some of the existing connections. The Clos network [Cl53] is nonblocking without requiring any rearranging. Both the Benes and Clos networks require more complex control than banyans and are therefore unsuitable for large-scale multiprocessors.

A *single stage* or *recirculating* IN is so called because requests circulate over a single stage of crossbars until they reach their destination output port. Although complexity is

only O(N) for an $N \times N$ single stage IN, the disadvantages of low bandwidth and little fault tolerance make them unsuitable in practice.

1.2 Multi-stage Interconnection Networks

MINs provide a cost-effective means of implementing a full connection from processors to memory with higher performance than the common bus and lower cost than the equivalent full crossbar. Many types of MINs have been proposed and we consider a sub-class of the banyan network, called a *delta network*. More recently, banyan networks have been extended in various ways to improve fault tolerance.

The purpose of this section is to introduce various kinds of MINs: banyan, delta and fault tolerant networks in particular, and discuss some of the design issues. The types of MINs considered are by no means exhaustive but are presented here merely to indicate broadly the most important classes.

1.2.1 Topology of Multi-stage Interconnection Networks

A MIN consists of a number of switching elements connected together via communication links to form a network. We may interpret each switching element as a *node* (or vertex) and each link as an *edge* (or arc). Ignoring the network input and output pins, the MIN topology may be defined as a directed graph with a number of *input nodes* (i.e. bases or nodes with no incident edges), a number of *output nodes* (i.e. apexes or nodes with no outgoing edges) and *internal nodes* (i.e. nodes with at least one incident edge and at least one outgoing edge). For an internal node A representing an $a \times b$ crossbar, we write indegree(A)=a and outdegree(A)=b.

Suppose a link connects two crossbars A and B, then the corresponding edge in the graph has two labels: the first one corresponding to the output pin number of A and the second to the input pin number of B. The edges are directed in the graph representation for convenience only, and actual links may be bi-directional. For each graph G representing a particular MIN, there is a reverse graph G^R which is formed by reversing all the edges and swapping the input and output nodes.

1.2.2 Banyan Networks

A banyan network, named after a type of Indian fig tree with a similar structure, is a MIN with a unique path from each input pin to each output pin. This criterion implies that the network has a tree-like fan-in and fan-out structure; a formal definition of

banyan networks in terms of graph theory is given by Goke and Lipovski, [GoLi73]. A banyan network is said to be *layered* if its switches can be arranged as a number of distinct stages. A layered banyan comprising J stages of crossbars in which outputs of one stage connect directly to the inputs of the next stage, is called a *J-level banyan.* There are two main types of J-level banyans: *regular banyans* and *irregular banyans.* Irregular banyans connect any N inputs to M outputs through J stages of crossbars where the ith stage employs c_i $n_i \times m_i$ crossbars. Since no crossbar pin is left unconnected, the number of outputs from one stage have to be the same as the number of inputs to the next stage, i.e.

$$c_i m_i = c_{i+1} n_{i+1} \qquad (i = 1,2,\ldots,J-1)$$

so that:

$$c_i = \prod_{j=1}^{i-1} m_j \prod_{k=i+1}^{J} n_k \qquad (i = 1,2,\ldots,J)$$

Thus $N = n_1 \times c_1 = n_1 \times n_2 \times \ldots n_J$ and $M = c_J \times m_J = m_1 \times m_2 \times \ldots m_J$

Regular banyans are constructed from a single type of basic crossbar switch (i.e. for i=2,3,...,J: n_i=n_1 and m_i=m_1) and examples of specific structures include CC banyans and SW banyans which are defined in [GoLi73].

In this section we consider SW banyans which can be defined recursively as follows for networks with 2×2 crossbars (the definition can be generalised easily to networks constructed from $a \times b$ crossbars with $a,b \geq 2$):

(i) A one-stage network, Π_1, is the single 2×2 crossbar

(ii) An s-stage network, Π_s, (s>0) has 2^s inputs and outputs, i.e. 2^{s-1} switches in each stage, and is constructed by connecting a *head stage* of 2^{s-1} switches to the right of 2 *tail networks* of (s–1) stages according to the partial shuffle topology shown in Figure 1.4. The ith switch in stage s takes its top input and bottom input from the ith pin of the upper sub-network and the ith pin of the lower sub-network, respectively. This property of the topology is particularly useful in obtaining the recurrence formulae in Chapters 4 and 5.

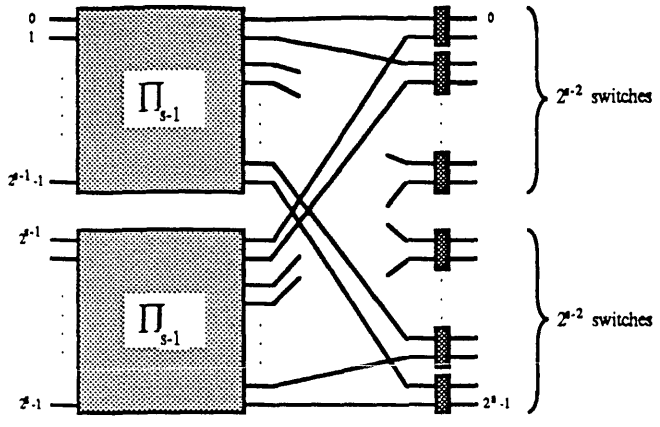


Figure 1.4 The SW Banyan Structure

1.2.3 Delta Networks

A well-known example of a regular banyan network is the delta network defined by Patel, [Pa81]. The definition of delta networks includes the additional property that routing in the network is *digit-controlled*, i.e. the choice of which output pin to select at a particular crossbar in the network can be determined by a single digit in the destination address of the packet. A rectangular delta network is constructed from *b*-way crossbars with the same number of input pins as output pins and is often called a *delta-b network*. There are a number of different ways of connecting the outputs of one stage of crossbars to the inputs of the next so as to obtain the desired connectivity. The corresponding permutation functions define the network's topology. All these topologies have been shown to be equivalent by Wu and Feng, [WuFe80]. Each topology can be obtained from any other by permuting the switches in each stage of the network and so all have the same performance characteristics, such as equilibrium throughput.

In general, the path from a given input node to some output node can be described by a *path descriptor* which is a string of digits where each digit indicates the output pin number of some intermediate switch. If this path descriptor is independent of the input node then we could associate it with the output node address and no longer require to store path descriptors at each input node. In delta networks, this independence of source node is maintained for all nodes and usually the address or the reversed address of the output node is the path descriptor. Hence messages can be routed through the network by consuming the destination address string from one end.

It is useful to have this delta property in the reverse direction as well as the forwards direction because messages sent in one direction need acknowledgements in the other

direction. Thus a delta network with the self-routing property in both directions is called a *bidelta network*. In graph terms, a delta network G is a bidelta network if G^R represents a delta network.

Each node A in a bidelta network is now associated with two types of path descriptors: forward (fpd(A)) describing the normal routing from an input node and backward (bpd(A)) describing the reverse route from an output node to node A. The way in which node A is labelled using fpd(A) and bpd(A) determines the topology of the network. Each descriptor can be reversed and the two descriptors can be concatenated in two ways. Hence there are eight possible topologies given the operators reverse (^{R}) and concatenation (<>):

fpd(A)<>bpd(A) which gives the *baseline network* (Figure 1.6(a))
bpd(A)<>fpd(A) which gives the *reverse baseline network* (Figure 1.6(b))
fpd^R(A)<>bpd(A) which gives the *flip network* (Figure 1.6(c))
bpd^R(A)<>fpd(A) which gives the *omega network* (Figure 1.6(d))
fpd(A)<>bpd^R(A) which gives the *modified data manipulator* (Figure 1.6(e))
bpd(A)<>fpd^R(A) which gives the *indirect binary cube* (Figure 1.6(f))
fpd^R(A)<>bpd^R(A) and bpd^R(A)<>fpd^R(A) which remain nameless (Figures 1.6(g) and 1.6(h) respectively)

Suppose an edge labelled (a,b) connects node A in stage j to node B in stage j+1 with labels $\alpha_1,\alpha_2,...,\alpha_{J-1}$ and $\beta_1,\beta_2,...,\beta_{J-1}$ in a J-stage network. In any of these topologies:

$$\text{fpd}(B) = \text{fpd}(A)<>[a] \text{ and } \text{bpd}(A) = \text{bpd}(B)<>[b].$$

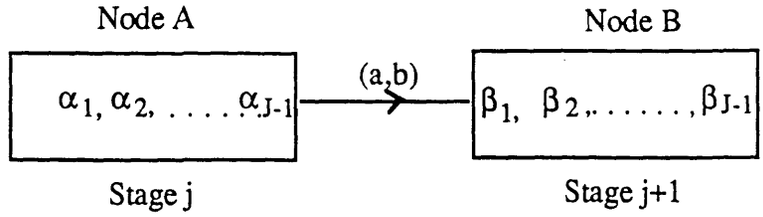


Figure 1.5 Labelling a Bidelta Network

Consider the baseline network (i.e. nodes labelled by fpd(A)<>bpd(A)) so that

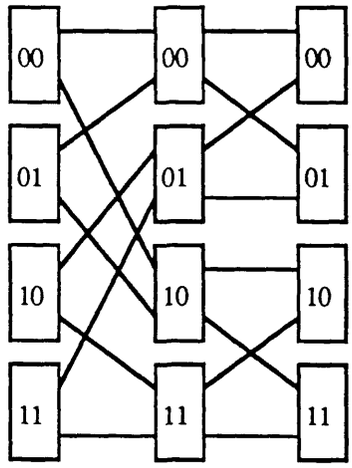
$$\text{fpd}(A) = \alpha_1,\alpha_2,...,\alpha_{j-1} \text{ and } \text{bpd}(A) = \alpha_j,\alpha_{j+1},...,\alpha_{J-1}$$

and likewise $\text{fpd}(B) = \beta_1,\beta_2,...,\beta_j$ and $\text{bpd}(B) = \beta_{j+1},\beta_{j+2},...,\beta_{J-1}$.

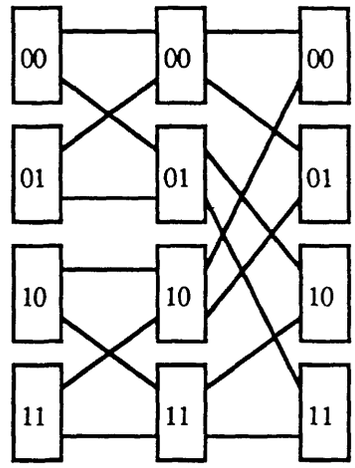
From the first equality fpd(B) = fpd(A)<>[a] we have: $\beta_i = \alpha_i$ ($1 \le i \le j-1$) and $\beta_j = a$ and from bpd(A) = bpd(B)<>[b] we have: $\beta_i = \alpha_{i-1}$ ($j+1 \le i \le J-1$) and $\alpha_{J-1} = b$.

Thus any node with label $\alpha_1,\alpha_2,...,\alpha_{J-2},b$ in stage j (j=1,2,...,J–1) connects to nodes in stage j+1 with labels $\alpha_1,\alpha_2,.\alpha_{j-1},a,\alpha_j,..,\alpha_{J-1}$ where $a \in$ {0,1,...,outdegree(A)}.

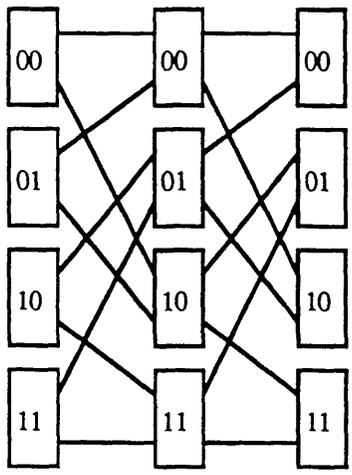
Connections for other topologies follow using a similar argument based on forward and backward path descriptors. It can be shown (as in [WuFe80]) that the baseline, reverse baseline, flip, omega, modified data manipulator, indirect binary cube and the remaining two topologies are isomorphic.



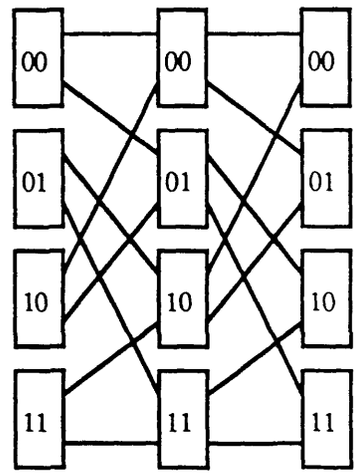
(a) **fpd(A)<>bpd(A) (Baseline)**



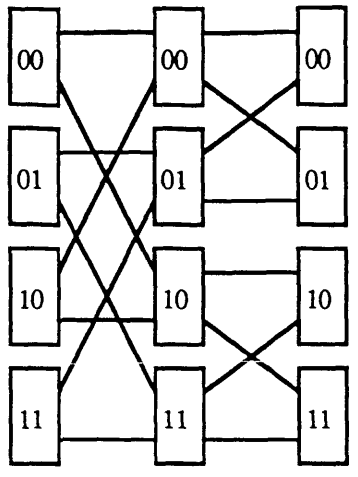
(b) **bpd(A)<>fpd(A) (Reverse Baseline)**



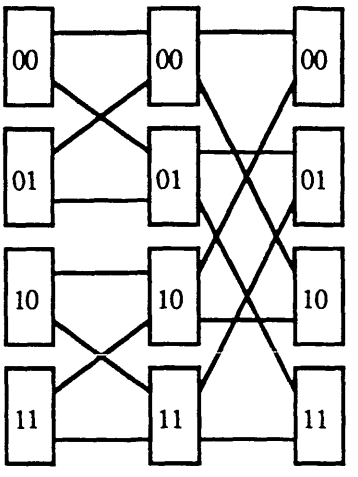
(c) **fpd^R(A)<>bpd(A) (Flip)**



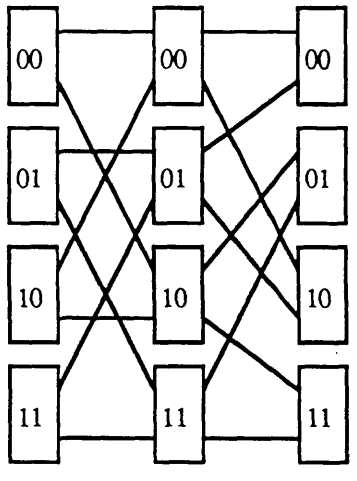
(d) **bpd^R(A)<>fpd(A) (Omega)**



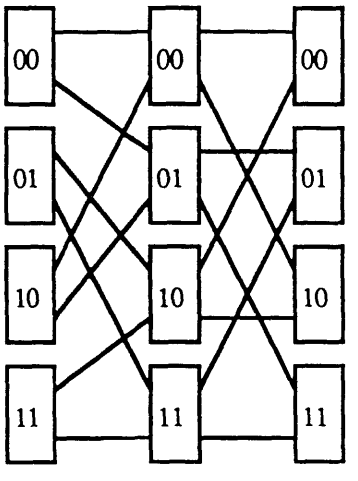
(e) $\mathbf{fpd}(A) <> \mathbf{bpd}^R(A)$
(Indirect Binary Cube)



(f) $\mathbf{bpd}(A) <> \mathbf{fpd}^R(A)$
(Modified Data Manipulator)



(g) $\mathbf{fpd}^R(A) <> \mathbf{bpd}^R(A)$



(h) $\mathbf{bpd}^R(A) <> \mathbf{fpd}^R(A)$

Figure 1.6 Delta Network Topologies

Notice that the mirror image topology is formed by swapping the fpd and the bpd, and that the topologies labelled with fpd^R (i.e. a reversed forward path descriptor) require the destination address to be consumed from the least significant bit whereas in the other four topologies the address is consumed from the other (most significant) end.

In the remaining chapters, we will be using the following simplified terminology for MIN topologies. The baseline and reverse baseline networks have a *partial shuffle* topology. The flip and omega networks have a *perfect shuffle* topology. The modified data manipulator and indirect binary cube have a *n-cube* topology. We will be primarily concerned with the partial shuffle topology.

1.2.4 Fault Tolerant MINs

As more and more parallel processing projects were constructed using banyan-type networks it soon became apparent that some form of fault tolerance was needed. There have been several studies to classify fault tolerant MINs [AAS87]. Rather than attempt to compare all types of fault tolerant MINs, and assess the degree of fault tolerance with respect to some fault tolerance model, we briefly describe a small selection of MINs to highlight the sort of extensions that can be made to delta networks.

A simple extension to the delta network is to add an extra stage of switches. This then gives multiple paths between input and output nodes so that single faults on one path may be overcome by choosing an alternative route. In addition to providing fault tolerance, multipath MINs give better throughput than unique path MINs because requests that get blocked can try another path. The BBN Butterfly network [ReTh86] is an example of an extra stage MIN.

An *r-replicated, d-dilated MIN* is a switching network which comprises r basic MINs connected to a stage of crossbars and has each link increased in capacity by a factor d. As well as this simple extension to the unique path MIN there have been some more complex extensions aimed at improved throughputs and fault tolerance but with minimal additional cost. For example, Augmented Shuffle Exchange Networks (ASENs) [KuRe87] are MINs in which the crossbars have auxiliary input and output pins that allow intra-stage connections. These auxiliary links form loops within stages so that blocked requests may be redirected via another crossbar in the same stage.

1.2.5 Operational Characteristics of Switching Networks

MINs are distinguished by three main operational characteristics: *control*, *timing* and *switching protocol*. Typically, MINs have decentralised control with implicit digit-controlled routing as in delta networks. However, timing characteristics depend on the type of parallel machine that uses the MIN. For SIMD machines, the MIN is normally used in synchronous mode whereas in MIMD machines the MIN is used in asynchronous mode. As mentioned in the Introduction, there are two main types of switching protocols – circuit switching and packet switching – as well as hybrid protocols. In circuit switching, a complete path has to be established across the network before data can be transferred from the input buffer to the destination buffer. In the process of setting up this path there may be a required link already in use which causes the path to be blocked. The partial path already established may then be held until the link becomes free or may be released whereupon the source processor may

have to retry after some appropriately chosen time. In a packet-switched MIN, there are buffers at each crossbar switch and packets of data are transferred from one buffer to the next in a single hop fashion. This protocol incurs higher transfer delays than circuit switching but introduces greater potential for parallelism and higher throughput. Hybrid protocols can also be used in buffered MINs in which packets can by-pass many buffers by forming circuits across any number of stages when possible.

In the ALICE machine, the interconnection is provided by a 64-way delta-4 network. This network operates in asynchronous mode with a circuit switching protocol in which partial paths are held. Similar protocols have been proposed for a new generation of INs constructed from optical crossbar switches [JO86] which can be fabricated with larger dimensions than electronic crossbars and so deliver higher throughput. At present, switching times in optical switches are non-negligible in contrast to the corresponding electronic devices and so it is preferable to hold a partial path rather than release the links when a request is blocked.

1.3 Models of MINs

Models of MINs with various operational characteristics have been presented in the literature. Here we briefly describe a selection of these models classified by the timing philosophy: synchronous or asynchronous mode of operation. We further classify these categories by the switching mode: packet or circuit switching and explain why asynchronous, circuit-switched MINs are harder to model than MINs with other operational characteristics. Each of the modelling techniques we consider has to overcome the problem of blocking in MINs, i.e. requests being blocked because of unavailable shared resources (links). We find that different types of blocking are involved in MINs with different operational characteristics and so different modelling techniques are sometimes required for each type.

1.3.1 Blocking in MINs

Three types of blocking have been identified for general queueing networks: *transfer*, *service* and *rejection blocking*. An approximate algorithm based on Mean Value Analysis (MVA) has been developed in [Ak87] which solves queueing networks with all three types of blocking involved. The maximum entropy approach can also be used to solve such networks [KoXe88]. Transfer blocking occurs at server i when a task completes its service and wishes to join server j but is unable to do so because queue j is full. Service blocking occurs at server i when a task is denied service at i because it wishes next to join server j which is running at full capacity or not available. Rejection

blocking occurs when a task at server i completes but can not go to queue j which is full and so instead rejoins queue i for another service.

One type of blocking is present in each type of operation mode for a given MIN. A form of rejection blocking where rejected tasks receive no service is present in many models of synchronous MINs. In asynchronous, packet-switched MINs with finite capacity buffers there is transfer blocking. Similar approximations as in [Ak87] can be applied here because the MIN is basically a queueing network with blocking. Service blocking occurs in the DMA (Direct Memory Access) servers attached to the inputs to the asynchronous, circuit-switched MIN because the data transfer (i.e. DMA service) cannot take place until a complete path across the MIN is established. Furthermore, all links in the path across the MIN need to be held at the same time and since the links are shared resources, we have the problem of *simultaneous resource possession* [Wh85].

1.3.2 Synchronous MIN Models

Over the past ten years numerous synchronous MIN models have been proposed primarily as a result of interest in SIMD computers. Much of this work involves the *permutation capability* of MINs, i.e. the number of total, 1-1 functions mapping input ports to output ports that can be realised by the MIN without path conflicts. Other work in this area has concentrated on determining the effective bandwidth of the MIN when requests arrive at an input with a given probability at the beginning of the time slice. Here we present an overview of [Pa81] which considers a synchronous circuit-switched network and the results of [Je83] which considers a synchronous packet-switched network.

Synchronous Crossbars

We start by considering a model of an $n \times b$ crossbar operating in synchronous mode. Bhandarkar, [Bh75], presents a discrete Markov chain model in which the state $(k_1,k_2,\ldots,k_b)$ with $\sum k_i=n$ represents the number of requests at each output pin. At the beginning of a cycle all free inputs request any output with equal probability. If at the end of a cycle there are k non-empty queues, then k requests need re-assigning because after the first request in each queue is served it effectively recycles to join again. By adding the recycled requests one by one to the queues, the terminal states can be determined and hence the transition probability matrix and steady state probabilities can be calculated. However, this is computationally complex and difficult to extend to arbitrary memory selection probabilities.

The complexity of the above Markov model can be reduced by assuming that all blocked requests in a given cycle, retry independently on the next cycle. This assumption has come to be known as *Strecker's approximation* [St70]. Let p be the probability of sending a request in a particular cycle (which is the same for all inputs). Thus the probability that the uppermost output is not connected via any of the n inputs is $\left[1 - \frac{p}{b}\right]^n$. Since all outputs are identical, the bandwidth (B) or expected number of busy output ports and the probability of acceptance (P_A) is given by, respectively:

$$B = b\left[1 - \left[1 - \frac{p}{b}\right]^n\right] \quad \text{and} \quad P_A = \frac{B}{np}$$

The error is about six to eight percent for $b/n>0.75$ when $p=1$. The closed form solution is widely used despite its underestimation of bandwidth at low request rates (p) and overestimation for higher p.

In [YPD82] the rate p is increased to take account of the rejected requests that are resubmitted. The new rate p' is given by:

$$p' = \frac{1}{1-P_A(1-1/p)}$$

This overestimates the bandwidth because it assumes that the *queued* requests are distributed uniformly over the outputs. By restricting the uniformity of the distribution, the model is refined to give very accurate results.

A queueing system approach assuming binomially distributed arrivals to the crossbar is investigated in [BaSm76]. This is asymtotically exact as either n and/or b tend to infinity but also give fairly accurate results for small values of n and b.

Synchronous, Circuit-switched MINs

Consider a J-stage delta network constructed from $a \times b$ crossbars. The following analysis of synchronous circuit-switched delta networks can be found in [Pa81]. In a delta network with uniform request patterns, the transmission rate of requests from each output in a given stage is the same. Thus we only need to consider one switch of each stage where the rate of requests at a switch output is a function of the rate of requests at the switch input. Let p_j ($0<j\leq J$) be the rate of requests at an output pin of a crossbar in stage j and p_0 be the rate of requests at each input port of the network. Applying Strecker's approximation to a crossbar in stage j we have:

for j=1,2,...,J, $\quad p_j = \left[1 - \left(1 - \frac{p_{j-1}}{b}\right)^a\right]$ and $\quad B = b^J p_J$

In [KrSn83], an asymptotic expression for p_J (with a=b) is derived which approximates to:

$$p_J = \frac{2b}{(b-1)J+2b/p_0}$$

The analysis of [Pa81] is extended to include favourite memory references in [Bh85].

Synchronous, Packet-switched MINs

The model in [Pa81] also applies to synchronous packet-switched delta networks provided that switches are unbuffered and packets are generated at the beginning of each cycle. The other models presented here can be classified by the size of the buffers used at each node: single, finite or infinite buffers.

The first paper on modelling buffered packed-switched MINs is by Dias and Jump, [DiJu81] which considers single buffers at each input pin. The switch is modelled using petri-nets and results in complicated expressions. Jenq, [Je83], has a similar analysis but without using petri-nets. Kruskal and Snir, [KrSn83], present a model for a MIN with infinite-sized buffers which implies that there is no blocking and the request rate p is the same at all stages in the steady state. More recently, there have been papers concerning finite buffered MINs, e.g. [HaJu87] and the analysis has been extended to include non-uniform routing [KiGa88].

1.3.3 Asynchronous Multi-stage Interconnection Network Models

Models of asynchronous MINs have been sparse in the literature. However, in recent years there has been considerable interest in developing such models mainly because of developments in two distinct areas: MIMD parallel computer architectures and communication networks. The need to design novel parallel architectures has encouraged the development of new modelling techniques and new applications for more established ones. In particular, the MINs in shared-memory architectures need to be modelled to aid the design process. The other impetus for modelling asynchronous MINs has come from the new integrated services digital communications network in which MINs provide a switching function for data packets.

Asynchronous, Packet-switched MINs

Here, we briefly describe the analysis reported by Mitra and Cieslak, [MiCi87], which assumes that buffers at links have infinite capacity. This assumption removes the problem of blocking and effectively makes the MIN a complex queueing network. Much of this complexity can be reduced because there is no feedback (i.e. no task can ever return after leaving a node) and each node in a particular stage has statistically identical behaviour. Thus, the analysis considers only one crossbar in each stage and approximates the superposition of arrival processes and the splitting of the departure processes by stationary renewal processes described by the traffic descriptors: the traffic intensity and the coefficient of variation of queueing time at the node. Assuming that the sources at the first stage are Poisson, the link queues are M/G/1 and so the Pollaczek-Khinchine formula gives the mean delay and other well-known results give the coefficient of variation of the departing process. In other stages the queues are GI/GI/1 and so further approximations need to be made. Finally, an expression for the mean time from source to destination is derived. The problem of blocking in asynchronous MINs with finite buffer capacity has been difficult to solve. An MVA algorithm for blocking in general queueing networks is reported in [Ak88] and this is extended for MINs in [JiBh88].

Asynchronous, Circuit-switched MINs

We will be considering asynchronous, circuit-switched MINs in the following chapters. Here we note that the problem of blocking still remains in this case and that it is further compounded by simultaneous resource possession. Similar problems arise in modelling telephone networks and in the next section we shall see that an elegant modelling technique has been developed for this case.

1.4 Modelling Circuit-Switched Networks

The modelling of large circuit-switched networks has long been of interest to telephone engineers, designers of large computer communication networks and more recently architects of parallel computers. Here we summarise some of the work presented by Kelly, [Ke86], and attempt to apply the results in MINs. We start by considering a Markov model for general circuit-switched networks which is exact but computationally impractical for large networks. However, in limiting cases links behave *as if* they were independent and this leads to Erlang's fixed-point approximation which is computationally tractable and gives very accurate results. We apply this approximation

to MINs below and discuss why the technique does not work well for this type of network.

1.4.1 Markov Model of Circuit-switched Networks

As an example, consider the following network in Figure 1.7.

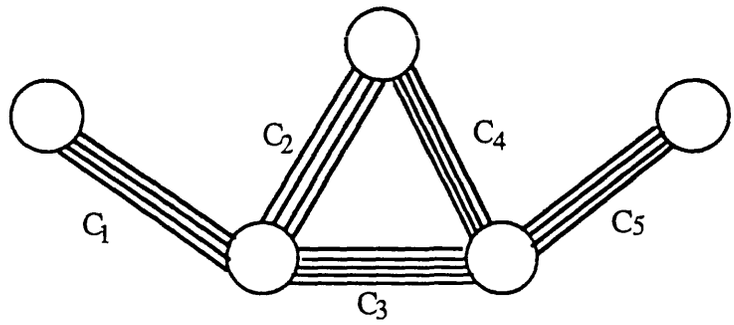


Figure 1.7 An Example of a Circuit-switched Network

In general the network may have an arbitrary topology. Suppose that there are J links labelled $j=1,2,...,J$ with link j comprising C_j circuits. A call on route r simultaneously holds $A_{jr} \in \{0,1\}$ circuits from link j. (More generally, a call may use more than one circuit from a link but here we will only consider the simpler case to simplify the notation.) Thus route r can be identified by $\{j \mid A_{jr}=1\}$ and the set of routes R is a subset of the power set of $\{1,2,...,J\}$. Calls requesting route r arrive as an independent Poisson process with rate υ_r and are lost if a required link has no free circuits. Successful calls on route r hold all links $j \in r$ for a random period of time with unit mean.

Let n_r be the number of calls in progress on route r,

$\boldsymbol{n} = (n_r \mid r \in R)$ describe the state of the network,

$\boldsymbol{C} = (C_j \mid j=1,2,\ldots,J)$ and $\boldsymbol{A} = (A_{jr} \mid j=1,2,\ldots,J;\ r \in R)$

The state space of the network can then be expressed by $S(\boldsymbol{C})=\{\boldsymbol{n} \in Z_+^R \mid \boldsymbol{A}\boldsymbol{n} \leq \boldsymbol{C}\}$

If $\pi(\boldsymbol{n})$ is the steady state probability of the process being in state $\boldsymbol{n}$, the detailed balance equations are:

$$\pi(\boldsymbol{n})\upsilon_r = \pi(\boldsymbol{n}^{(r)})\ (n_r+1) \quad \text{where } \boldsymbol{n}, \boldsymbol{n}^{(r)} \in S(\boldsymbol{C})$$

and $\boldsymbol{n}^{(r)}$ is the vector $\boldsymbol{n}$ with n_r incremented by one.

These equations are satisfied by (See [BHJ48] or [Ke79]):

$$\pi(\boldsymbol{n}) = G(\boldsymbol{C})\prod_{r\in R}\frac{\upsilon_r^{n_r}}{n_r!} \qquad \text{where } G(\boldsymbol{C})^{-1} = \sum_{\boldsymbol{n}\in S(\boldsymbol{C})}\prod_{r\in R}\frac{\upsilon_r^{n_r}}{n_r!}$$

The steady state probability of acceptance of a call on route r is given by:

$$\sum_{\boldsymbol{n}\in S(\boldsymbol{C}-\boldsymbol{A}\boldsymbol{e}_r)}\pi(\boldsymbol{n}) \quad = \quad \frac{G(\boldsymbol{C})}{G(\boldsymbol{C}-\boldsymbol{A}\boldsymbol{e}_r)}$$

where $\boldsymbol{e}_r\in S(\boldsymbol{C})$ is the unit vector describing just one call or route r.
These simple forms of equations may appear at first to give the complete solution. However, the number of routes can grow exponentially with the number of nodes J (since $|R|\leq 2^J$). So the normalising constant is only computable for small networks.

1.4.2 Limiting Regimes

Suppose the capacities C_j and offered traffic υ_r are increased together such that the ratios C_j/υ_r are kept fixed. It has been shown in [Ke86] that in the limit, parameters $B_j\in[0,1)$ associated with link j emerge such that the probability of losing a call on route r is given by:

$$L_r = 1 - \prod_{j\in r}(1-B_j)$$

Thus in the limit, it appears *as if* links block independently, link j blocking with probability B_j.

For star networks, where all calls pass through the central node, it has been shown in [KeZi85] that in the limit when the number of nodes approaches infinity, the same result holds. These limiting regimes suggest that links behave as if they block independently when link capacities are high and/or when the number of nodes is large.

1.4.3 Erlang's Fixed-Point Approximation

Suppose a link of capacity C is offered Poisson traffic with rate υ and is held for some random time with mean one. This forms a M/G/C/C queue and in the steady state the probability that a call is lost is the same as the probability that all circuits are busy since Poisson arrivals "see" the link in equilibrium. This probability can be determined from Erlang's Loss formula:

$$E(\upsilon,C) = \frac{\upsilon^C}{C!}\left[\sum_{n=0}^{C}\frac{\upsilon^n}{n!}\right]^{-1}$$

The throughput of this link is $\upsilon(1-E(\upsilon,C))$.

In the network we make the assumptions that before traffic υ_r is offered to link j, it is thinned by a factor $1-E_i$, $i \in r-\{j\}$ (where E_i is the blocking probability for link i) and all thinnings are independent. Thus it can be shown that:

$$E_j = E(\sum_{r \in R} A_{jr} \upsilon_r \prod_{i \in r-\{j\}} (1-E_i),\ C_j) \qquad \text{where } E_j \in (0,1)$$

This is Erlang's fixed-point equation and in [Ke86] it is shown that the fixed-point exists and is unique for fixed routing. For dynamic routing, where alternative routes are attempted when the first route is unavailable, there may be more than one fixed-point. The work in [KeZi85] indicates that for more complex networks Erlang's fixed-point approximation gives more accurate results.

1.4.4 Circuit-switched MIN Model using Erlang's Fixed-Point

Consider an s-stage delta-b network as an open network with arrival rate υ per input pin and uniform routing across all output pins. There are b^s routes from each input pin, all of length $s+1$ links and all links have unit capacity. Since blocking is uniform across all stages (since partial paths are released) let the link blocking probability be B. The arrival rate per route is υ/b^s and is thinned by a factor $(1-B)^s$. Thus Erlang's fixed-point equation is $B = E(\upsilon(1-B)^s,1)$ and hence throughput is $b^s\upsilon(1-B)^{s+1}$. This looks suspect because the first equation is independent of the degree of the crossbars in the delta network, but depends only on the number of stages. For small networks (e.g. 3-stage delta-2 network) the exact state space probabilities of the Markov model can be determined. Cope, [Co87], shows that the Erlang model for a MIN gives unacceptably large errors in throughput compared to the exact model.

The inadequacy of Erlang's fixed-point approximation for MINs indicates that the links do not block as if they were independent. From the limiting regimes (Section 1.4.2), we can see that Erlang's model for MINs is inappropriate because MINs have unit capacity links and small number of links per node. Cope, [Co87], extends the Erlang model to other fixed-point methods that achieve more accurate results.

Note that all these models deal with networks which release partial paths whereas the main body of the thesis will be concerned with holding partial paths.

Chapter 2

Representation of Interconnection Networks in Queueing Models

Outline

2.1 Flow Equivalent Server for the Interconnection Network

2.2 Underlying Markov Process

2.3 Underlying Birth and Death Process

2.4 Virtual Bank of Infinite Servers

Queueing networks provide a powerful tool for modelling computer and communication systems. In the cases of parallel computer architectures and INs (connecting processing elements to shared memory modules) queueing models are crucial to the design process. However, the IN is a passive resource in that links in it need to be held by active resources for them to provide their service. Thus queueing network models of the parallel machines have blocking and this rules out the possibility of obtaining an exact product-form solution.

We consider a circuit-switched delta network in which partially established paths are held. Assuming that the remaining components of the parallel machine are easy to model, the queueing network model of the entire machine will have service blocking at servers attached to the IN. Even without blocking this queueing network may have a large number of servers which may preclude an efficient solution. Hence, we need to apply aggregation techniques which replace a collection of servers by a single server. Bearing this in mind, we extend these techniques so that INs can be represented in a queueing network.

2.1 Flow Equivalent Server for the Interconnection Network

Consider a M server queueing network whose state is described by $\underline{n}=(n_0,n_1,\ldots,n_{M-1})$ where $n_i \geq 0$ $(i=0,1,\ldots,M-1)$ is the queue length at server i. In the steady state we can equate the flow into state $\underline{n}$ with the flow out of state $\underline{n}$ to obtain the global balance equations. In *product-form networks*, these equations can be partitioned into *local balance equations*. If there exist steady state probabilities which satisfy these equations then the network is said to have the *local balance property*. In such networks, the queues may be considered in isolation and the steady state probabilities of the entire network will be a product of the steady state probabilities of each queue (up to a normalising constant).

In a locally balanced queueing network, it is a standard result [CHW75] that any sub-network can be replaced by a flow equivalent server (FES). The service rate of the FES with queue length N is equal to the throughput of the *short-circuited* sub-network with population N. This result is an application of Norton's theorem from electrical circuit theory. In general, it cannot be applied to queueing networks which do not have the local balance property. However, in many such cases when it is applied it gives a good approximation [Co77].

We apply Norton's theorem to sub-networks which comprise a bank of direct memory access (DMA) servers together with some circuit-switched IN, the remainder of the enclosing network satisfying the local balance property. Service blocking at the DMA servers implies that this decomposition is not exact but at least it isolates the problem. Under appropriate assumptions, the behaviour of the short-circuited sub-network can be represented by a stationary Markov process. For each valid population of this closed system, we find its throughput and hence FES-rate by solving the balance equations of this process.

The switching network is represented by a function which determines the effective service rate of the bank of DMA devices when they compete for network paths. The appropriate functions for full crossbar and delta networks are derived in Chapter 3 and 5 respectively; in the form of a recurrence relation that exploits the recursive structure of the network in the latter case. In Section 2.3, we consider the limiting case in which there is *no* degradation, i.e. every active input port of a network is always connected to a destination. In this case, we obtain the well-known expression for the throughput of a multiple server. Later in Chapter 3, we also derive a closed formula for FES-rate when the switching network is a crossbar.

2.2 Underlying Markov Process

The flow equivalent server for b parallel DMA channels connected through an interconnection network is defined by the short-circuited network shown in Figure 2.1 for populations $N>1$. The switching network may be a full crossbar or a delta network. When $N=1$ the customer experiences no contention for links and so the throughput is the same as the DMA service rate. The shaded box represents the switching network which limits the number of active outputs, m, to some value between one and the number of active inputs, n, according to its internal connection topology.

We make the following assumptions about this closed system with a circuit-switched IN in which partial paths are held:

- All queues have a FCFS discipline.
- All tasks are of the same class, i.e. we do not distinguish between read and write requests.
- The IN is used in one direction only, i.e. requests arrive only at DMA servers and the network does not set up paths in the reverse direction.

- Service time of DMA servers has negative exponential distribution with rate μ when a connection is made, i.e. message lengths have this distribution.
- Path set up time and release times are negligible compared to service durations, i.e. the path across the network or partial path to the first blockage will be established instantly. In practice, this is a reasonable assumption because the DMA service times are many orders of magnitude higher that switching times in contemporary electronic switches.
- Recycling task chooses any of the b queues with equal probability.

For the present, we will not need to know the memory reference pattern of the tasks in the network.

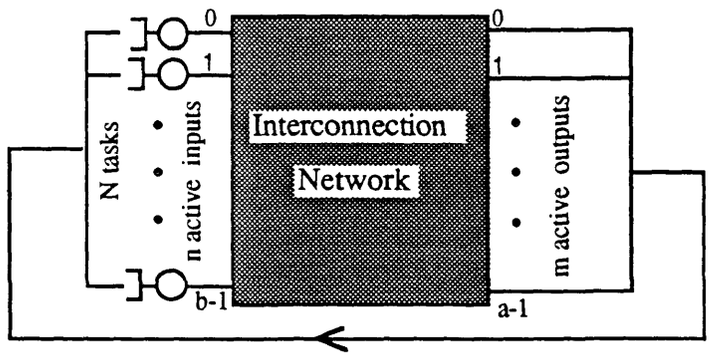


Figure 2.1 The Short-circuited Sub-network defining the FES Rate

For this closed system, with population $N>1$, we define the following:

- $\underline{n} = (n_0,...,n_{b-1})$ is a vector of queue lengths at the source servers ($n_i \geq 0$, $i=0,1,\ldots,b-1$).
- m is the corresponding number of active output pins.
- $\#(\underline{n})$ is the number of non-zero components of $\underline{n}$; (we will use n to denote $\#(\underline{n})$ where there is no confusion).

We also define the following random variables:

- $\underline{N}(t)$ in which the component $N_i(t)$ denotes the number of tasks at server i ($i=0,1,\ldots,b-1$) at time $t \geq 0$.
- $M(t)$ which denotes the number of active outputs at time $t \geq 0$.

Thus we define the following process:

$$W(t) = \{(\underline{N}(t),M(t)) \mid t \geq 0\}$$

with state space $$\Omega_w = \{(\underline{n},m) \mid \sum_{i=1}^{b-1} n_i = N,\ m=1,2,\ldots,\min(a,n)\}$$

This is a non-null recurrent Markov process (with generators defined by appropriate state transition probabilities) under the assumption that all servers have negative exponentially distributed service times; this assumption can be relaxed for various queueing disciplines in standard ways, as described by Baskett et al, [BCMP75].

We could now obtain the balance equations for the state space probabilities of $W(t)$ in the steady state by considering every possible state transition which may occur on a service completion. Suppose a path from server i is released and the recycled task joins server j. Then there are three types of free output port acquisitions:

(a) by task blocked on path from i,

(b) by next task in queue i, and

(c) by recycled task if queue j was empty.

In an s-stage IN, there may be up to s tasks blocked on the path from server i and so there may be s type (a) acquisitions. At the same time, there may be two other acquisitions, one of type (b) and the other of type (c). Altogether there may be up to $s+2$ pin acquisitions on a single service completion. Thus, state transitions $(\underline{n},m)$ to $(\underline{n}',m')$ can occur such that

$$m' = m-1,\ m,\ m+1,\ m+2,\ldots,\ m+s+1$$

This results in a large number of equations and the approach is impractical even in the simple case of a full crossbar. Moreover, in a similar analysis of a delta network, the state space Ω_w must be extended to represent every path and partial path established, vastly increasing computational complexity.

The number of states of the process $W(t)$ can be reduced considerably by aggregating states which represent the same number of active input ports. The aggregation results in the following process:

$$X(t) = \{(\#(\underline{N}(t)),\ M(t)) \mid t \geq 0\}$$

with state space $$\Omega_x = \{(n,m) \mid n=1,2,\ldots,b;\ m=1,2,\ldots,\min(a,n)\}$$

Next we will make assumptions about the arrangements of the N tasks on the n queues. In the next chapter, we show that all such arrangements are equally likely in the case of full crossbars.

2.3 Underlying Birth and Death Process

Suppose the switching network can be represented by an expression υ_n, the expected number of active outputs when there are n active inputs in the steady state. We assume (approximately) that this relationship holds at all times. In the case of a direct connection (no switching network) every active 'input' will always be connected to an 'output' and we have $\upsilon_n=n$ which does always hold exactly. We will also use $\mu_n = \upsilon_n\mu$ to denote the *effective service rate function* of a bank of servers, each with rate μ, connected through such a network when n servers have non-empty queues. Thus it is no longer necessary to include the number of active outputs in the state space, since we can approximate this by υ_n when there are n active inputs. This approximate aggregation then leads to the following process when the population of the short-circuited sub-network is N :

$$Y(t) = \{\underline{N}(t) \mid t\geq 0\} \text{ with state space } \Omega_y = \{\underline{n} \mid \Sigma n_i = N\}$$

Let $\pi(\underline{n})$ be the stationary probability of the process being in state $\underline{n}\in\Omega_y$.

We now aggregate all states $\underline{n}$ with the same number of non-zero components and consider the following integer-valued, stationary Markov process:

$$Z(t) = \{\#(\underline{N}(t)) \mid t\geq 0\} \text{ with state space } \Omega_z = \{n \mid 1\leq n\leq \min(b,N)\}$$

This process describes the number of active inputs to the switching network at time t and has stationary probability p_n for $n\in\Omega_z$, i.e. p_n is the steady state probability that n input pins are active. The aggregate state n therefore represents the subset of states $S_n = \{\underline{n} \mid \#(\underline{n})=n, \Sigma n_i=N\}$ of the process $Y(t)$, where the sets S_n form a partition of Ω_y. In fact, for a given n, each state in S_n has the same steady state probability with respect to the process $Y(t)$, i.e. each arrangement of customers in any n non-empty queues is equally likely. This follows because the generators of the Markov process $Y(t)$ depend only on n and not directly on the component values n_i. The balance equations for $Z(t)$ then follow immediately by expressing $\pi(\underline{n})$ in terms of p_n for each $\underline{n}\in\Omega_y$. Of course a stronger result holds in a simple Markovian queueing network (with no switching network) when the visitation rate to service rate ratio is the same for all servers, namely that *all* states are equi-probable (cf. the multiple server result discussed below).

We define the combinatorial function (binomial coefficient), C, by ${}^nC_m = \dfrac{n!}{m!(n-m)!}$ and use the following elementary combinatorial results:

(i) ${}^nC_m = \dfrac{n}{m}{}^{n-1}C_{m-1}$ for integers $n\geq m>0$.

(ii) ${}^nC_m = {}^{n-1}C_{m-1} + {}^{n-1}C_m$ for integers $n\geq m>0$.

(iii) The number of ways of arranging n balls in m urns is ${}^{n+m-1}C_{m-1}$. Suppose there are $n+m-1$ items in a line, then the result follows since every choice of $m-1$ items (each as a partition) creates m containers (urns) with a total of n remaining items (balls).

(iv) The number of ways of arranging n balls in m *non-empty* urns is ${}^{n-1}C_{m-1}$. This follows from (iii) since there is at least one ball in each of the m urns and so the result is the same as the number of ways of arranging $n-m$ balls in m urns (m having already been accounted for).

Thus we deduce from (iv) that for $n \geq m$, $\sum_{k=1}^{m} {}^{m}C_k \, {}^{n-1}C_{k-1} = {}^{m+n-1}C_{m-1}$ since if n balls are arranged in m urns, they must occupy k non-empty urns for some k ($1 \leq k \leq m$) and there are ${}^{m}C_k$ ways of selecting k urns from m. This result follows formally from Lemma 3.1 in the next chapter.

Now by analysing process $Z(t)$ in the steady state and using the above combinatorial results we obtain the following theorem:

THEOREM 2.1

Consider a b-input IN with effective service rate function μ_n in a cyclic network comprising a parallel bank of DMA servers with population N. Under the assumption that all arrangements of customers on n non-empty queues are equally likely, the throughput is given by:–

$$T(N) = \sum_{n=1}^{b} \mu_n p_n \tag{2.1}$$

where

$$p_n = \frac{p_1 \mu_1 \prod_{j=1}^{n-1} \{(b-j)(N-j)\}}{\mu_n (n-1)!^2} \tag{2.2}$$

and

$$p_1^{-1} = \sum_{n=1}^{b} \frac{\mu_1 \prod_{j=1}^{n-1} \{(b-j)(N-j)\}}{\mu_n (n-1)!^2} \tag{2.3}$$

PROOF

State transitions of the process $Z(t)$ can only occur on completion of service by a task which then recycles to join one of the b queues. Thus there are only two possible types

of state transitions: $n \rightarrow n+1$ and $n+1 \rightarrow n$ $(1 \leq n < b)$ since transitions $\underline{n} \rightarrow \underline{n}'$ of the process $Y(t)$ with $\#(\underline{n})=\#(\underline{n}')$ do not cause a change of state in $Z(t)$. For the first type of transition to be possible, the completing task must depart from a queue in which there is at least one other task (length > 1), and must then join one of the $b-n$ empty queues. Let q_{ij} be the probability that the state changes from i to j on a service completion. Since states (of $Y(t)$) with the same number of non-empty queues are equi-probable, the transition $n \rightarrow n+1$ occurs with probability $q_{n,n+1}$ given by:–

$$q_{n,n+1} = \frac{(b-n)}{b} \text{ Pr(queue length at a given active input} > 1 \text{ when } n \text{ are active)}$$

$$= \frac{(b-n)}{b} \frac{\text{\# arrangements of } N-1 \text{ tasks in } n \text{ non-empty queues}}{\text{\# arrangements of } N \text{ tasks in } n \text{ non-empty queues}}$$

$$= \frac{(b-n)}{b} \frac{{}^{N-2}C_{n-1}}{{}^{N-1}C_{n-1}} \text{ by standard combinatorial results as above}$$

$$= \frac{(b-n)}{b} \frac{(N-n)}{(N-1)} \qquad (2.4)$$

Conversely, for the second type of transition, the completing task must leave from a queue in which there are no other tasks and must then join one of the n non-empty queues. The transition $n+1 \rightarrow n$ therefore has probability $q_{n+1,n}$ given by:–

$$q_{n+1,n} = \frac{n}{b} \text{ Pr(queue length at a given active input} = 1 \text{ when } n+1 \text{ are active)}$$

$$= \frac{n}{b} \frac{\text{\# arrangements of } N-1 \text{ tasks in } n \text{ non-empty queues}}{\text{\# arrangements of } N \text{ tasks in } n+1 \text{ non-empty queues}}$$

$$= \frac{n}{b} \frac{{}^{N-2}C_{n-1}}{{}^{N-1}C_{n}} = \frac{n^2}{b(N-1)} \qquad (2.5)$$

Now from (2.4) and (2.5), given that the effective service rate function of the switching network is μ_n, we have the following balance equations for the process $Z(t)$ with $n=1,2,\ldots,b-1$:

$$\mu_n p_n (b-n)(N-n) = \mu_{n+1} p_{n+1} n^2 \qquad (2.6)$$

Solving this for p_n gives (2.2),with normalising constant given by (2.3). The expression for the throughput $T(N)$ follows from this distribution and the effective service rate function. □

When the effective service rate function μ_n is a simple expression, it is not necessary to compute the distribution p_n directly to obtain the throughput $T(N)$ by Theorem 2.1. In such cases an expression for throughput may be obtained by the following theorem:

THEOREM 2.2

Consider a b-input IN with effective service rate function μ_n, in a closed system of a parallel bank of DMA servers with population N. Under the assumption that all arrangements of customers on n non-empty queues are equally likely, the throughput is given by:–

$$T(N) = \frac{bN}{b+N-1} H(1) \tag{2.7}$$

where
$$H(z) = \sum_{n=1}^{\infty} \frac{t_n z^n}{n} \tag{2.8}$$

and t_n is given by the following recurrence formula and boundary condition, respectively:–

$$(b-n)(N-n)t_n = n^2 t_{n+1} \qquad n=1,2,\ldots,b-1 \tag{2.9}$$

and
$$\sum_{n=1}^{b} t_n/\mu_n = 1 \tag{2.10}$$

PROOF

Substituting $t_n = \mu_n p_n$ in the balance equations (2.6) and the normalising condition for p_n we obtain the recurrence formula (2.9) and the boundary condition, (2.10). We can solve this by defining the generating function

$$G(z) = \sum_{n=1}^{\infty} t_n z^n \qquad (t_n = 0 \text{ for } n>b) \tag{2.11}$$

$G(z)$ has first derivative $G'(z) = \sum_{n=1}^{\infty} n t_n z^{n-1}$ and in fact $H(z) = \int_0^z \frac{G(u)}{u}\,du$

We now have $T(N) = G(1)$ and we can rewrite the recurrence formula (2.9) as

$$Nb(t_n/n) - (N+b)t_n + nt_n + t_{n+1} - (n+1)t_{n+1} = 0$$

Multiplying by z^n and summing yields

$$NbH(z) - (N+b)G(z) + zG'(z) + z^{-1}(G(z)-t_1 z) - G'(z) + t_1 = 0$$

i.e. $$(1-z)G'(z) + (N+b-z^{-1})G(z) - NbH(z) = 0$$

Thus, setting $z=1$, we obtain the required expression for the throughput $T(N)$. □

In Theorem 2.2, $H(1)$ is derived from the boundary condition (2.10) which depends on the specific form of μ_n given by the characteristics of the switching network.

In the simplest exact case where we have direct connection (no switching network), $\mu_n=n\mu$ and the boundary condition (2.10) becomes $\sum_{n=1}^{b} \frac{t_n}{n\mu} = 1$, i.e. $H(1) = \mu$, giving the well-known 'multiple server' formula for $T(N) = \frac{bN}{b+N-1}\mu$ derived by a number of authors, e.g. [HaFi86]. There is a similarity between the expression for $T(N)$ when there is a direct connection and the expression for μ_n when there is a full crossbar (given in the next chapter). In fact, the two expressions can be unified by the substitution, $N=n$. As we will see shortly, this is to be expected because both expressions can be derived by purely probabilistic arguments which use the fact that each arrangement of tasks over a fixed number of non-empty queues occurs with equal probability in the steady state.

2.4 Virtual Bank of Infinite Servers

In parallel computer architectures, the processors may be idle for some time waiting for communication and so they are usually multiprogrammed to increase utilisation, i.e. one real processor (agent) behaves like a number of virtual processors. This is particularly important in architectures with relatively small grain parallelism such as ALICE. In many cases the agents are synchronised with some other agent across the network. For example, in the ALICE machine a virtual agent sends a request to a packet pool agent (memory module) for more work and awaits a reply. Hence, for any new arrival (the reply), there has to be an idle virtual agent that initiated the task even though the actual agent may be busy. Such agents are effectively infinite servers.

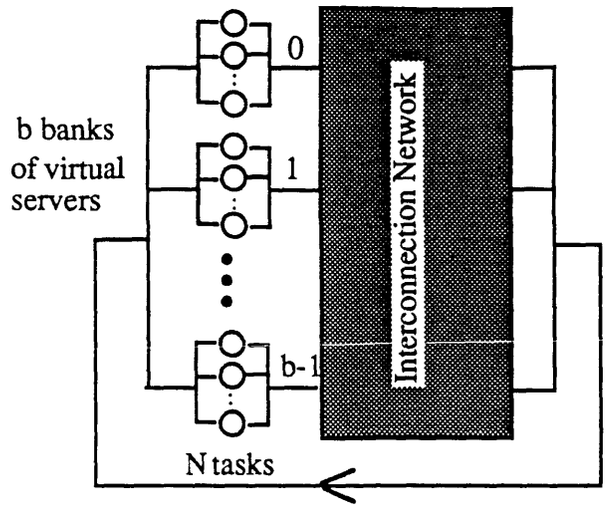


Figure 2.2 VBIS Servers with an Interconnection Network

Suppose the b DMA servers connected to the MIN in Figure 2.2 are multiprogrammed as v virtual agents and form a virtual bank of infinite servers (VBIS). The short-circuited sub-network with population $N \leq bv$ can be analysed by a similar approach to that adopted in Section 2.3. Now the analysis of process $Z(t)$ (assuming the effective service rate function is μ_n) leads to Theorem 2.3. First we require the following combinatorial results:

Let $R(N,M,v)$ be the number of ways of distributing N balls on M urns under the v-occupancy restriction (i.e. maximum of v balls per urn). It is shown in [GaCh87],

$$R(N,M,v) = \sum_{j=0}^{M} (-1)^j \, {}^{M}C_j \, {}^{N+M-j(v+1)-1}C_{M-1} \qquad (2.12)$$

Let $A(M,N,v)$ be the number of ways of distributing N balls on M non-empty urns under the v-occupancy restriction. So $A(N,M,v) = R(N-M,M,v-1)$ since one ball is fixed in each of the M non-empty queues (thereby reducing the number of vacant slots in each queue by one) and the remaining balls are arranged on top. Thus,

$$A(N,n,v) = \sum_{j=0}^{M} (-1)^j \, {}^{M}C_j \, {}^{N-jv-1}C_{M-1} \qquad (2.13)$$

THEOREM 2.3

Consider a b-input IN with effective service rate function μ_n in a cyclic network comprising a parallel bank of DMA servers with multiprogramming level v and total

population N. Under the assumption that all arrangements of customers on n non-empty queues (banks of virtual servers) are equally likely, the throughput is given by:–

$$T(N) = \sum_{n=1}^{b} \mu_n p_n \qquad N=1,2,\ldots,bv \qquad (2.14)$$

where

$$p_n = p_1 \frac{\mu_1}{\mu_n} \prod_{r=1}^{n-1} \frac{(1-\gamma(r))\delta(r)}{\gamma(r+1)(1-\delta(r))} \qquad n=1,2,\ldots,b \qquad (2.15)$$

p_1 normalises these probabilities

$$\delta(n) = \frac{(b-n)v}{bv-N+1}, \qquad \gamma(n) = \frac{A(N-1,n-1,v)}{A(N,n,v)} \qquad (2.16),\ (2.17)$$

PROOF

First we observe that $\gamma(n)$ is the steady state probability that a particular server has a singleton queue when n servers are busy. To see this, note that the numerator for expression (2.17) is the number of ways of arranging N–1 tasks on n–1 non-empty queues; one task being chosen to be in the singleton queue. The denominator is the number of ways of arranging N tasks on n non-empty queues. Hence, (2.17) follows from (2.13).

Similarly, $\delta(n)$ is the steady state probability that a recycling task joins an idle server when n are busy. Any idle virtual agent (or slot) is chosen by an arrival since each configuration of N customers is equally likely. The numerator for expression (2.16) is the number of empty slots on idle servers (i.e. $(b-n)v$) and the denominator is the total number of empty slots (i.e. $bv-N+1$) seen by an arrival.

p_n is the steady state probability that the process $Z(t)$ is in state n, i.e. has n active inputs. Equating the flow from state n to state $n+1$ with the reverse transition we obtain the following balance equations:

$$\mu_n p_n (1-\gamma(n))\delta(n) = \mu_{n+1} p_{n+1} \gamma(n+1)(1-\delta(n+1)) \qquad n=1,2,\ldots,b-1$$

which are satisfied by (2.15).
The expression for throughput (2.14) directly follows from (2.15). □

The analysis of a VBIS server without a MIN has been presented in [HaFi86] in which they derive a simple recurrence for the expected number of busy servers in the parallel

bank when there are N customers in the queues. In our approach, the throughput for this special case can be obtained by Theorem 2.3 with $\mu_n=n\mu$.

The results of the early part of this chapter forms the basis of much work later in the thesis. In the following chapters, we will consider different types of interconnection networks (such as crossbars, delta networks, etc.) in the closed system with parallel DMA servers. We begin in the next chapter with an analysis of the uniform crossbar.

Chapter 3

Models of a Uniform Crossbar Switch

Outline

3.1 Markov Model for a Full Crossbar
    Notation
3.2 Markov Model for a Uniform Crossbar
    3.2.1 Transition Probabilities
    3.2.2 Global Balance Equations
    3.2.3 Constrained Markov Process
    3.2.4 Exact Result for a 2 by 2 Crossbar
3.3 Probability Model for a Uniform Crossbar
3.4 Throughput Function for a Uniform Crossbar
3.5 Static Probabilities for a Uniform Crossbar

Since the crossbar forms the basis of the MIN analysis in later chapters it is important to model it as accurately as possible. The crossbar model also gives us an upper bound for the throughput of the equivalent MIN. This chapter describes a number of Markov models and probability models of full crossbars under a uniform traffic model. The Markov models provide the exact analysis with respect to certain assumptions whereas the probability models are derived more intuitively but are more computationally tractable. Our primary interest in this chapter is to derive expressions for the throughput of a uniform crossbar in a closed system with a fixed population. This allows comparisons between crossbars and delta networks providing the same connectivity. Furthermore, we derive expressions for the throughput conditional on the number of active inputs. These expressions can be used not only in the birth and death process in the previous chapter but also as the base case for analysing uniform delta networks in Chapter 5.

3.1 Markov Model for a Full Crossbar

In this chapter we consider a closed system with population N comprising a $b \times a$ crossbar connecting b DMA servers to a output ports (as in Figure 3.1). This is the special case of Section 2.2 where the IN is a crossbar and we make exactly the same assumptions here. The closed system is effectively a conventional queueing network but with service blocking at the DMA servers. Tasks at the head of the DMA queues are simultaneously in an output port queue. When a task reaches the head of the output port queue the DMA service begins and lasts for a random time interval with negative exponential distribution and mean $1/\mu$. Without loss of generality we let $\mu=1$.

Let $N_i(t)$ be the random variable which describes the number of tasks at DMA server i at time t and $M_j(t)$ be the random variable which describes the number of tasks queued on output port j at time t.
Now consider the process $V(t) = \{\underline{N}(t),\underline{M}(t) \mid t \geq 0\}$ with state space given by:

$$\Omega_V = \{(\underline{n},\underline{m}) \mid (n_i \geq 0, i=0,1,\ldots,b-1), \sum_{i=0}^{b-1} n_i = N,\ (m_j \geq 0, j=0,1,\ldots,a-1), \sum_{j=0}^{a-1} m_i = \#(\underline{n})\ \}.$$

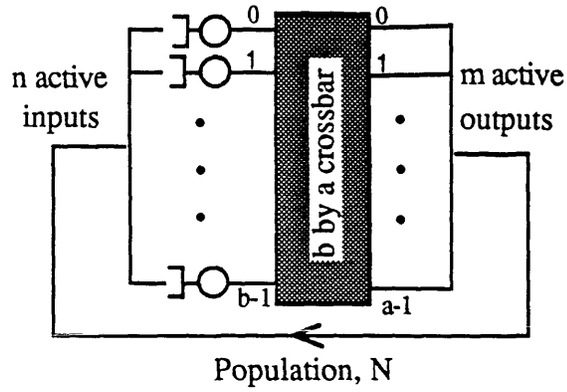


Figure 3.1 The Full Crossbar

Notation

We will be using the following notation in this and subsequent chapters:

- The components of a vector describing queue lengths are labelled starting at zero so that they correspond with the pin numbers of the crossbar.
- The number of non-zero components in a vector $\underline{n}$ is denoted by $\#(\underline{n})$.
- $\underline{n}^i$ denotes the vector $\underline{n}$ with component i incremented by one.
- $\underline{n}_j$ denotes the vector $\underline{n}$ with component j decreased by one.
- we will use the following abbreviations where there is no confusion: $n=\#(\underline{n})$, $m=\#(\underline{m})$ and $q=m_0$.

For the uniform crossbar, each DMA server is chosen by a recycled task with equal probability ($1/b$) and each output port is also chosen with equal probability ($1/a$). By symmetry when m outputs are active, any of the ${}^aC_m$ arrangements is equally likely. Furthermore, if n inputs and m outputs are active then any of the ${}^{n-1}C_{m-1}$ arrangements of n tasks on the m non-empty queues is equally likely (see Section 3.2.2). Hence we only need to consider the number of active outputs m rather than the output queue lengths $\underline{m}$ in the full state space.

Later, in the next chapter, we consider the non-uniform crossbar in which one DMA server is chosen with probability ω and each of the other output ports is chosen with probability $(1-\omega)/(a-1)$. In this case, each arrangement of active outputs is not always equally likely but for a given value of the number of tasks q at the special output port, the arrangements of the other $n-q$ tasks on the m (if $q=0$) or $m-1$ (if $q>0$) non-empty queues is equally likely. Hence we include m and q in the state space.

We could also consider an alternative process for a crossbar with the following state space:

$$\Omega' = \{ (\underline{n},\underline{d}) \mid (n_i \geq 0, i=0,1,\ldots,b-1), \sum_{i=0}^{b-1} n_i = N,\ d_i=0,1,\ldots,a-1 \text{ if } n_i>0,$$
$$d_i=\text{undefined if } n_i=0\}$$

where d_i describes the destination of the head task in DMA queue i. Note that unlike state space Ω_v, Ω' holds explicit information about the dynamic connections in the crossbar and this complicates the expressions for the transition rates. On a service completion at output port j any DMA i with $d_i=j$ may have completed. Since information about the connections is not present in Ω_v, any one of the n active DMA servers may have completed (with appropriate probability). The sets Ω_v and Ω' are related by

$$m_j = \text{number of } d_i \text{ such that } d_i=j,\ i=0,1,\ldots,b-1.$$

Hence the number of active outputs $m=\#(\underline{m})$ is a long expression. This in turn gives long expressions for the state transition rates if we consider the process with state space Ω' and so we consider process $V(t)$ with state space Ω_v.

3.2 Markov Model for a Uniform Crossbar

In this section we consider process $W(t)$ with state space Ω_w as in Section 2.2 but for the case when the switching network is a full crossbar. For a $b\times b$ crossbar, this work can be found in [Ha87]. Here we extend this work to the $b\times a$ crossbar case and state some of the main results.

3.2.1 Transition Probabilities

Suppose server i completes and the recycled task joins server j ($0\leq i,j\leq b-1$), then we can distinguish three types of transitions when the number of active inputs n: increases by one, decreases by one, and remains the same. Correspondingly, we have three domains for (i,j):

$D(\theta) = \{i,j \mid n_i>0,\ n_j=1, 1\leq i\neq j\leq b\}$

$D(\phi) = \{i,j \mid n_i=0,\ n_j>1, 1\leq i\neq j\leq b\}$

$D(\psi) = \{i,j \mid n_i>0,\ n_j>1, 1\leq i,j\leq b\} \cup \{i,j \mid n_i=0,\ n_j=1, 1\leq i,j\leq b\} \cup \{i,i \mid n_i=1, 1\leq i\leq b\}$

Transitions are of the form $(\underline{n}^i_j,m') \rightarrow (\underline{n},m)$ where $m'= m-2,m-1,m,m+1$ and so we define the following:

$\theta_k(n,m)$ is the transition rate from (n,m) to $(n+1,m+k-2)$, $k=1,2,3,4$

$\phi_k(n,m)$ is the transition rate from (n,m) to $(n-1,m+k-2)$, $k=1,2$

$\psi_k(n,m)$ is the transition rate from (n,m) to $(n,m+k-2)$, k=1,2,3

where for $n>1$ and $n=1$, $m=1$ respectively

$\theta_1(n,m) = m(m-1)^3 / n(n-1)a^2$	0
$\theta_2(n,m) = m\{(n-m)m^2+(m-1)(m-1)(a-m+1)\} / n(n-1)a^2$	$1/a$
$\theta_3(n,m) = m(a-m)\{(n-m)(2m+1)+(m-1)(a-m+1)\} / n(n-1)a^2$	$(a-1)/a$
$\theta_4(n,m) = m(a-m)(a-m-1)(n-m) / n(n-1)a^2$	0
$\phi_1(n,m) = m(m-1) / n(n-1)$	undefined
$\phi_2(n,m) = m(n-m) / n(n-1)$	undefined
$\psi_1(n,m) = m(m-1)^2 / n(n-1)a$	0
$\psi_2(n,m) = m\{m(n-m)+(m-1)(a-m+1)\} / n(n-1)a$	1
$\psi_3(n,m) = m(n-m)(a-m) / n(n-1)a$	0 (3.1)

The factor m/n in all the transition rate functions is the probability that server i is connected to an active output.

3.2.2 Global Balance Equations

By analysing process $W(t)$ for a uniform $b\times a$ crossbar, we obtain the following theorem.

THEOREM 3.1

The steady state probabilities $P(\underline{n},m)$ of process $W(t)$ with state space Ω_w are given by:

$$P(\underline{n},m)m = \frac{1}{b}\sum_{(i,j)\ \in D(\theta)} \{P(\underline{n}^i_j,m+1)\theta_1(n-1,m+1) + P(\underline{n}^i_j,m)\theta_2(n-1,m) + P(\underline{n}^i_j,m-1)\theta_3(n-1,m-1) + P(\underline{n}^i_j,m+2)\theta_4(n-1,m+2)\}$$
$$+ \frac{1}{b}\sum_{(i,j)\ \in D(\phi)} \{P(\underline{n}^i_j,m+1)\phi_1(n+1,m+1) + P(\underline{n}^i_j,m)\phi_2(n+1,m)\}$$
$$+ \frac{1}{b}\sum_{(i,j)\ \in D(\psi)} \{P(\underline{n}^i_j,m+1)\psi_1(n,m+1) + P(\underline{n}^i_j,m)\psi_2(n,m) + P(\underline{n}^i_j,m-1)\psi_3(n,m-1)\} \quad (3.2)$$

PROOF By analysis of assumed Markov process $W(t)$. □

The factor $1/b$ in the above balance equations is the probability that the recycled task joins server j=0,1,...,b–1.

Since the transition rate functions are independent of i and j (they only depend on n and m), we can reduce the state space of process $W(t)$ to obtain process $X(t)$ with state space $\Omega_x = \{(n,m) \mid n=1,2,\ldots,b \text{ and } m=1,2,\ldots,\min(n,a)\}$ and steady state probabilities

$P(n,m)$. In equilibrium if all arrangements of N tasks on n non-empty queues are equally likely then the following equation must hold:

$$P(\underline{n},m) = \frac{P(n,m)}{{}^{N-1}C_{n-1}\ {}^{b}C_{n}} \qquad \text{where } n=\#(\underline{n}) \qquad (3.3)$$

By substituting this expression for $P(\underline{n},m)$ in Theorem 3.1 we obtain the following corollary.

COROLLARY 1 TO THEOREM 3.1

The steady state probabilities $P(n,m)$ of process $X(t)$ with state space Ω_x satisfy the following balance equations:

$$\begin{aligned}
P(n,m)m(N-1)b \;=\; & \\
& (n-1)(N-n+1)\ \{P(n-1,m+1)\theta_1(n-1,m+1) + P(n-1,m)\theta_2(n-1,m) \\
& \qquad + P(n-1,m-1)\theta_3(n-1,m-1) + P(n-1,m+2)\theta_4(n-1,m+2)\ \} \\
& + n^2(n+1)\ \{P(n+1,m+1)\phi_1(n+1,m+1) + P(n+1,m)\phi_2(n+1,m)\} \\
& + n\{n(N-n)+(b-n+1)(n-1)\}\{P(n,m+1)\psi_1(n,m+1) \\
& \qquad + P(n,m)\psi_2(n,m) + P(n,m-1)\psi_3(n,m-1)\}
\end{aligned} \qquad (3.4)$$

PROOF By analysis of process $X(t)$ or by substitution of equation (3.3) in Theorem 3.1. □

3.2.3 Constrained Markov Process

Suppose the transitions of process $X(t)$ were constrained in such a way that the number of active inputs n remains the same. This constrained process represents an open model with n infinite queues at the crossbar inputs. When $n=b$, this corresponds to a closed model with infinite population and so will give an upper bound on the throughput. This approximate approach is particularly useful as it gives a simple solution to the steady state probabilities:

COROLLARY 2 TO THEOREM 3.1

The steady state probabilities of the constrained process $X(t)$ with $n=1,2,\ldots,b$ active inputs is given by

$$P(m|n) = \frac{{}^{n-1}C_{m-1}\ {}^{a}C_{m}}{{}^{a+n-1}C_{n}} \qquad m=1,2,\ldots,\min(n,a) \qquad (3.5)$$

PROOF

For the constrained process, only the third summation of balance equations in Theorem 3.1 is required since n is fixed, and so we obtain:

$$P(m|n)m = P(m+1|n)\psi_1(n,m+1) + P(m|n)\psi_2(n,m) + P(m-1|n)\psi_3(n,m-1) \tag{3.6}$$

This is satisfied by (3.5) and later we show that the probabilities do sum to one (3.13). □

3.2.4 Exact Result for a 2 by 2 Crossbar

In the case of a 2×2 crossbar switch, $b=a=2$, with $N>1$, the values of the transition rate functions are $\theta_2(1,1)=\theta_3(1,1)=1/2$, $\phi_1(2,2)=1$, $\phi_2(2,1)=1/2$, $\psi_1(2,2)=1/2$, $\psi_2(1,1)=1$, $\psi_2(2,1)=1/4$, $\psi_2(2,2)=1/2$, $\psi_3(2,1)=1/4$, and the (dependent) equations of Corollary 1 to Theorem 3.1 become:

$$\begin{aligned}
(N-1)P(1,1) &= 2P(2,2) + P(2,1) \\
(2N-1)P(2,1) &= (N-1)P(1,1) + 2(2N-3)P(2,2) \\
2(2N-1)P(2,2) &= (N-1)P(1,1) + (2N-3)P(2,1)
\end{aligned} \tag{3.7}$$

These equations have normalised solution:

$$P(1,1) = \frac{4}{(3N+1)}, \qquad P(2,1) = \frac{2(N-1)}{(3N+1)}, \qquad P(2,2) = \frac{(N-1)}{(3N+1)} \tag{3.8}$$

giving a throughput $T(N) = 1.P(1,1) + 1.P(2,1) + 2.P(2,2) = \dfrac{4N}{(3N+1)}$ (3.9)

Thus throughput is always less than 4/3, and approaches this value as $N\to\infty$. This is as expected for a full 2×2 crossbar with both of its inputs active. For finite N, there is always a non-zero probability that an input is inactive, giving lower throughput.

3.3 Probability Model for a Uniform Crossbar

Rather than the Markov model representing the dynamic behaviour of the crossbar, the probability model described here uses the properties of the crossbar in equilibrium and simple combinatorial arguments to determine steady state probabilities. The purpose of this section is to derive an expression for the effective service rate function for a $b\times a$ crossbar (i.e. the throughput conditional on the number of active outputs). From this, the overall throughput function can be determined using Theorem 2.2.

For a $b \times a$ crossbar, let the random variables N, M denote respectively, the number of active inputs and the number of active outputs. We define the following steady state probability: $p_{m|n} = \Pr(M=m \mid N=n)$. Now by symmetry, all valid arrangements of the n requests on a queues are equally likely in equilibrium. Hence, to find the probability that m of those n requests will be successful we need to find the number of arrangements of n requests on m non-empty queues and divide by the total number of arrangements. The number of arrangements can be determined by urn model results and simple combinatorial arguments as in the previous chapter. This gives the following result:

$$p_{m|n} = \frac{{}^{n-1}C_{m-1}\ {}^{a}C_{m}}{{}^{a+n-1}C_{n}} \qquad m=1,2,\ldots,\min(n,a) \qquad (3.10)$$

This expression agrees with the stationary probability (3.5) for the constrained Markov process $X(t)$ with n active inputs. Hence the following results will be valid for both the probability model and the constrained process.

Now we wish to derive the throughput of the crossbar given the number of active inputs. First we will require the following lemma and corollary which can also be found in [Kn68] and [HaPa87]:

LEMMA 3.1

$$\text{For } n \geq 1,\ 0 \leq h \leq X-n+1, \quad \sum_{k=1}^{n} {}^{n-1}C_{k-1}\ {}^{X}C_{k+h-1} = {}^{n+X-1}C_{X-h} \qquad (3.11)$$

PROOF

For $n=1$, the left hand side, lhs $= {}^{X}C_{h} = {}^{X}C_{X-h}$ = rhs, the right hand side if $0 \leq h \leq X$. Now assume inductively that the result is true for $1 \leq n \leq N$. Then,

$$\sum_{k=1}^{N+1} {}^{N}C_{k-1}\ {}^{X}C_{k+h-1} = \sum_{k=2}^{N} ({}^{N-1}C_{k-2} + {}^{N-1}C_{k-1})\ {}^{X}C_{k+h-1} + {}^{X}C_{N+h} + {}^{X}C_{h}$$

$$= \sum_{k=1}^{N} ({}^{N-1}C_{k-1}\,{}^{X}C_{k+h} + {}^{N-1}C_{k-1}\,{}^{X}C_{k+h-1})$$

(we have replaced the dummy summation variable, k, by $k+1$ in the first sum, and extended each summation domain to include the two loose terms.)

Therefore, using the inductive hypothesis twice, for $0 \leq h+1 \leq X-N+1$ and $0 \leq h \leq X-N+1$, i.e. for $0 \leq h \leq X-N$,

$$\text{lhs} = {}^{N+X-1}C_{X-h-1} + {}^{N+X-1}C_{X-h}$$

$$= {}^{N+X}C_{X-h} \ \square$$

COROLLARY

For $n \geq 1$, $X \geq 1$, $0 \leq h \leq X-n$, $\sum_{k=1}^{n} {}^{n-1}C_{k-1} \, {}^{X}C_{k+h}.(k+h) = X.{}^{X+n-2}C_{X-h-1}$ (3.12)

PROOF

Since ${}^{X}C_{k+h}.(k+h) = X.{}^{X-1}C_{k+h-1}$, lhs $= X \, . \sum_{k=1}^{n} {}^{n-1}C_{k-1} \, {}^{X-1}C_{k+h-1}$ = rhs by the lemma. $\square$

Now by Lemma 3.1 with $h=1$ and $X=a$, $\sum_{m=1}^{\min(n,a)} p_{m|n} = \frac{{}^{n+a-1}C_{a-1}}{{}^{a+n-1}C_{n}} = 1$ (3.13)

By the corollary (3.12) with $h=0$ and $X=a$, the expected number of active outputs when there are n active inputs is given by:–

$$\sum_{m=1}^{\min(n,a)} p_{m|n} m = a \frac{{}^{a+n-2}C_{a-1}}{{}^{a+n-1}C_{a-1}} = \frac{an}{(a+n-1)}$$

Since each DMA server has rate μ, the degraded service rate function is given by:–

$$\mu_n = \frac{an}{(a+n-1)} \mu \qquad n=1,2,\ldots,b \qquad (3.14)$$

3.4 Throughput Function for a Uniform Crossbar

Now that we have a simple expression for μ_n in the case of a uniform crossbar, we can use Theorem 2.2 to obtain a simple expression for the throughput $T(N)$.

COROLLARY TO THEOREM 2.2

For a closed system with population N, consisting of b parallel DMA servers transmitting across a $b \times a$ crossbar network the throughput is given by:–

$$T(N) = \frac{abN\mu}{(a+b-1)N + (a-1)(b-1)} \qquad (3.15)$$

PROOF

For a $b \times a$ crossbar network, we obtained in the previous section an expression for μ_n (3.14). Substituting this in the boundary condition (2.10) gives:

$$G(1) + (a-1)H(1) = a\mu.$$

Substituting $T(N) = G(1)$ and using (2.7) in Theorem 2.2 we obtain

$$\left\{ \frac{bN}{b+N-1} + (a-1) \right\} H(1) = a\mu$$

which gives the required expression for throughput. □

Since a and b are interchangeable in (3.15), the corollary shows that a $b \times a$ crossbar has exactly the same throughput as an $a \times b$ crossbar. As an example, consider the extreme cases of a $b \times 1$ and $1 \times b$ crossbar. In the first case, the throughput is μ because there is only one output and the other case has the same throughput because there is only one input. For more general cases, the result is not so obvious.

For a 2×2 crossbar (3.15) becomes $T(N) = \frac{4N}{3N+1}\mu$ which agrees with our more rigorously derived expression (3.9). Hence our simpler analysis of IN throughput using process $Z(t)$ and effective service rate function μ_n is exact with respect to the more precise analysis in Section 3.2 for a 2×2 crossbar.

In the case of a multi-stage delta network, the expression for μ_n is given by a recurrence formula (as we will see in the following chapters), and so a compact expression for throughput would appear impossible to derive. We have obtained our numerical predictions (see Chapter 8) of throughput $T(N)$ in this case from the degraded rates μ_n by directly computing the set of probabilities p_n (using (2.2) and (2.3)) and hence $T(N)$ from (2.1), from Theorem 2.1.

3.5 Static Probabilities for a Uniform Crossbar

Consider a uniform $b \times a$ crossbar represented by process $X(t)$ with state space Ω_x (as in Section 3.2). Suppose we constrain the process so that $n=1,2,\ldots,b$ remains fixed and consider the embedded Markov chain of $X(t)$. We make the further assumption that when the crossbar state changes the next state can be any one in the state space with a probability that is a function of the next state only. In effect, we have a zeroth order stochastic process in that the next state is independent of the current state as well as all previous states. With this gross simplification of the process we can readily derive the

steady state probabilities which we call *static probabilities*. They can be derived by considering all the possible transitions from an arbitrary state.

We will require the following lemma.

LEMMA 3.2

For $n \geq 1$, $k=0,1,\ldots,n-1$

$$\sum_{\underline{r}} \prod_{j=1}^{k} (r_j - j) = S(n,k)$$

$$2 \leq r_i \leq n,\ i=1,2,\ldots,k$$

$$r_i < r_{i+1},\ i=1,2,\ldots,k-1$$

where

$\underline{r}$ is an integer vector with distinct elements

$S(n,0) = 1$

$S(n,k) = S(n-1,k) + (n-k)\, S(n-1,k-1) \qquad k=1,2,\ldots,n-2$

$S(n,n-1) = 1$

PROOF

We prove this lemma by case analysis on $k=0,1,\ldots,n-1$.

Case $k=0$: Empty product is one by definition, so $S(n,0)=1$

Case $k=1$: $S(n,1) = \sum_{r_1=2}^{n} (r_1 - 1) = \sum_{r_1=1}^{n-1} r_1 = \frac{n(n-1)}{2}$

(This is an optional base case which reduces the number of recursive calls when k=1.)

Case $k=1,2,\ldots,n-2$: As all elements of $\underline{r}$ are distinct and sorted in increasing order, we can separate it as follows.

$$\sum_{\underline{r}} \prod_{j=1}^{k} (r_j - j) = \sum_{\underline{r}} \prod_{j=1}^{k} (r_j - j) + \sum_{\underline{r}} \prod_{j=1}^{k} (r_j - j)$$

$2 \leq r_i \leq n,\ i=1,2,\ldots,k$ $\quad$ $2 \leq r_i \leq n-1,\ i=1,2,\ldots,k$ $\quad$ $2 \leq r_i \leq n-1,\ i=1,2,\ldots,k-1$

$r_k = n$

$S(n,k) = S(n-1,k) + (n-k)\, S(n-1,k-1)$

Case $k=n-1$: Since all elements of $\underline{r}$ are distinct and sorted, for this case $\underline{r}=(2,3,\ldots,n)$ and $S(n,n-1) = \prod_{j=1}^{n-1} (r_j - j) = 1$ $\square$

THEOREM 3.2

The static probability that m outputs are active in a uniform $b \times a$ crossbar with n active inputs is given by:

$$P_a(m|n) = \frac{a!\ S(n,n-m)}{a^n(a-m)!} \qquad n=1,2\ldots,b;\ m=1,2,\ldots,\min(n,a)$$

PROOF

Since the static probabilities correspond to a zeroth order process, its short term behaviour is identical to its long term behaviour. Thus the static probability of the process being in state m is the probability that the initial state is m. The latter probability can be determined by the arriving sequence of n requests as follows.

Consider n requests arriving in sequence and choosing any given output pin with probability $1/a$. The i^{th} request in the sequence is given a label $i=1,2,\ldots,n$ which we call the *request number*. Let $k=n-m$ be the number of blocked requests. We can describe the blocking by the vector $\underline{r} = (r_1,r_2,\ldots,r_k)$ which represent the request numbers that are blocked, i.e. r_j is the request number of the j^{th} request to be blocked. Request number zero is never blocked as all output pins are free on its arrival.

Now the probability that the blocking is $\underline{r}$ is given by:

$$\left[\prod_{i=1}^{r_1-2}\frac{a-i}{a}\right]\left(\frac{r_1-1}{a}\right)\left[\prod_{i=r_1-1}^{r_2-3}\frac{a-i}{a}\right]\left(\frac{r_2-2}{a}\right)\cdots\cdots\left[\prod_{i=r_k-k}^{n-(k+1)}\frac{a-i}{a}\right]$$

$$= \frac{1}{a^{n-1}}\left[\prod_{j=1}^{k}(r_j-j)\right]\left[\prod_{i=1}^{n-(k+1)}(a-i)\right]$$

$$= \frac{a!}{a^n(a-m)!}\prod_{j=1}^{n-m}(r_j-j)$$

Hence the probability that m outputs are active is given by summing these probabilities over all blockings $\underline{r}$ that give rise to m active outputs:

$$P_a(m \mid n) = \frac{a!}{a^n(a-m)!}\sum_{\underline{r}}\prod_{j=1}^{n-m}(r_j-j)$$

$$2 \le r_i \le n,\ i=1,2,\ldots,n-m$$

$$r_i < r_{i+1},\ i=1,2,\ldots,n-m-1$$

$$= \frac{a!\ S(n,n-m)}{a^n(a-m)!} \qquad \text{by Lemma 3.2 } \square$$

Static probabilities actually relate to a synchronous crossbar in which n tasks arrive simultaneously at the beginning of a cycle and request any output with equal probability. Although they are not directly useful for the asynchronous protocol which we consider, they will prove useful in the analysis of non-uniform crossbars in the following chapter.

Chapter 4

Models of the Non-uniform Crossbar Switch

Outline

4.1 Markov Model for a Biased Crossbar
- 4.1.1 Transition Probabilities
  - Notation for Transition Probabilities
  - Transition Probability Functions
- 4.1.2 Global Balance Equations
- 4.1.3 Derivation of the Birth-Death Process
- 4.1.4 Constrained Process

4.2 Approximate Probability Model for a Non-uniform Crossbar
- 4.2.1 Static Probabilities for a Non-uniform Crossbar
- 4.2.2 Transformed Static Probabilities

4.3 Non-uniform Routing with Different Pin Release Times
- 4.3.1 Constrained Process
- 4.3.2 Analysis of a 2 by 2 Crossbar
- 4.3.3 Output Pin Utilisation in a Biased Crossbar

So far we have only considered a crossbar with uniform routing to its output pins. However, in a shared-memory multiprocessor where the crossbar provides the interconnection between processors and memories, the memory references are not always uniform. Here, we analyse a particular memory reference pattern for a $b \times a$ crossbar in which the top output pin (numbered zero) is chosen with some given probability and all the other pins are chosen with equal probability. The purpose of the first part of this chapter is to provide an exact analysis of a $b \times a$ crossbar with this reference pattern using a Markov model. Later in the chapter, we also present some approximate models of biased crossbars.

4.1 Markov Model for a Biased Crossbar

As before we consider a $b \times a$ crossbar in a closed system with population N (as shown in Figure 4.1). The b DMA servers have service times which are negative exponentially distributed with rate μ when connected to an output port. In general, the routing probability to each of the a outputs is arbitrary in a non-uniform crossbar. In a *biased* crossbar any new arrival chooses the top pin with probability ω and each other output pin with probability $(1-\omega)/(a-1)$.

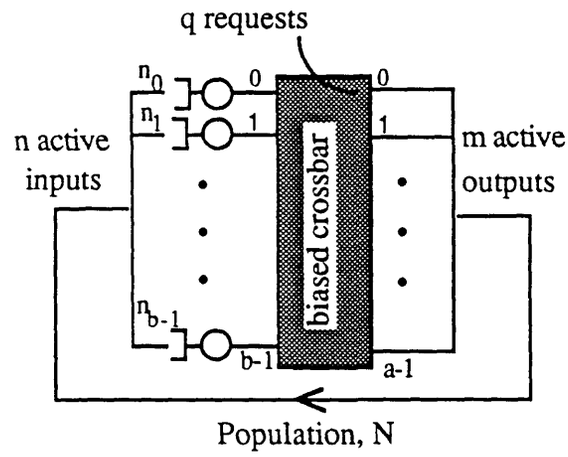


Figure 4.1 State of a Biased Crossbar

We can model a general non-uniform crossbar by the process $V(t)$ with state space Ω_V given in Section 3.1. However, for a biased crossbar, output pins 1 to $a-1$ have identical properties such as equilibrium queue length distribution. So we consider the following process:

$W'(t) = \{(\underline{N}(t), \#(\underline{M}(t)), M_0(t)) \mid t \geq 0\}$ which has state space $\Omega_{W'}$ given by:

$$\Omega_{w'} = \{ (\underline{n},m,q) \mid \sum_{i=0}^{b-1} n_i = N, [\ (q=0,\ 1 \le m \le \min(a-1,n))$$
$$\text{or } (1 \le q \le n-1,\ 2 \le m \le \min(a,n-q+1)) \text{ or } (q=n,\ m=1)\ ]\ \}$$

The uniform crossbar is a special case of the biased crossbar in two ways. Of course when $\omega=1/a$, the biased crossbar has equal routing to all its outputs as in the uniform crossbar. The Markov process describing the biased crossbar is still more complicated because of the extra parameter q in the state space. We can eliminate this parameter by considering the $b \times a$ crossbar with $\omega=0$. This implies $q=0$ and so the process describes a $b \times (a-1)$ uniform crossbar.

4.1.1 Transition Probabilities

On a service completion at DMA i ($i=0,1,\ldots,b-1$), three types of output pin acquisitions can simultaneously occur:

Event A : The freed output pin is acquired by a task in the remaining queues (previously blocked on that output)

Event B : A free output pin is acquired by the next task at DMA i (given that there is one)

Event C : A free output pin is acquired by the recycled task (given that it joins an idle server)

Although these events occur instantaneously, they are ordered A, B, C as we would expect in the physical crossbar. Suppose the recycled task joins server j, then we have the following micro-transitions which make up the transition from the initial state $(\underline{n},m,q)$:

$$(\underline{n},m,q) \rightarrow (\underline{n}_i,m^{(A)},q^{(A)}) \rightarrow (\underline{n}_i,m^{(B)},q^{(B)}) \rightarrow (\underline{n}_i,m^{(C)},q^{(C)}) \rightarrow (\underline{n}^j,m',q')$$

DMA i completes — Event A — Event B — Event C

where $m^{(X)}$ is the number of active outputs prior to event $X \in \{A,B,C\}$
and $q^{(X)}$ is the number of tasks queued on pin 0 prior to event $X \in \{A,B,C\}$
Note that:

$m^{(A)} = m-1$ (i.e. on completion there is one less output)

and $q^{(A)} = q$ with probability $(m-1)/m$ (i.e. if a non-top pin releases, $i \neq 0$)
$= q-1$ with probability $1/m$ (i.e. if the top pin releases, $i=0$, $q>0$)

Suppose output pin 0 is freed, then $\Pr(A) = 0$ if $q=1$ and $\Pr(A) = 1$ if $q>1$
Suppose another output pin is freed (1 to $a-1$),
then Pr(Remaining queues do *not* require the freed pin | $q=1,2,\ldots,n-2$)

$$= \frac{\text{number of arrangements of } (n-q)-1 \text{ tasks on } (m-1)-1 \text{ non-empty queues}}{\text{number of arrangements of } n-q \text{ tasks on } m-1 \text{ non-empty queues}}$$

$$= \frac{{}^{n-q-2}C_{m-3}}{{}^{n-q-1}C_{m-2}} \quad \text{by combinatorial arguments (Section 2.3)}$$

$$= \frac{m-2}{n-q-1}$$

Hence we have the following probabilities for the event A when a non-top pin is freed:

$$\Pr(A) = \frac{n-m}{n-1} \qquad \text{if } q=0$$

$$\Pr(A) = \frac{n-q-m+1}{n-q-1} \qquad \text{if } q=1,2,\ldots,n-2$$

$$\Pr(A) = 0 \qquad \text{if } q=n-1$$

For the other two events we have the following probabilities:

$$\Pr(B) = \frac{a-(m^{(B)}-1)}{a} \qquad \text{if } q^{(B)}=q^{(C)},\ q^{(B)}>0$$

$$\Pr(B) = \frac{a-m^{(B)}}{a} \qquad \text{if } q^{(B)}=q^{(C)}=0$$

$$\Pr(B) = 1 \qquad \text{if } q^{(B)}=0,\ q^{(C)}=1$$

$$\Pr(B) = 0 \qquad \text{if } q^{(B)}=q^{(C)}-1,\ q^{(B)}\geq 1$$

Likewise,

$$\Pr(C) = \frac{a-(m^{(C)}-1)}{a} \qquad \text{if } q^{(C)}=q',\ q^{(C)}>0$$

$$\Pr(C) = \frac{a-m^{(C)}}{a} \qquad \text{if } q^{(C)}=q'=0$$

$$\Pr(C) = 1 \qquad \text{if } q^{(C)}=0,\ q'=1$$

$$\Pr(C) = 0 \qquad \text{if } q^{(C)}=q'-1,\ q^{(C)}\geq 1$$

Notation for Transition Probabilities

The pair (i,j) corresponds to a customer leaving pin i and joining pin j. We define the following domains for (i,j) when the crossbar is in state $(\underline{n},m,q)$:

$D(\phi) = \{i,j \mid n_i=0,\ n_j>1, 1\leq i\neq j\leq b\}$

$D(\psi) = \{i,j \mid n_i>0,\ n_j>1, 1\leq i,j\leq b\} \cup \{i,j \mid n_i=0,\ n_j=1, 1\leq i,j\leq b\} \cup \{i,i \mid n_i=1, 1\leq i\leq b\}$

$D(\theta) = \{i,j \mid n_i>0,\ n_j=1, 1\leq i\neq j\leq b\}$

which correspond respectively to n decreasing by one, n remaining the same and n increasing by one after the transition. The transition probabilities in the given domains for (i,j) are denoted by:

$$\phi_{k,l}(n,m,q) : (n,m,q) \rightarrow (n-1,m+k-2,q+l-2) \qquad \text{if } (i,j)\in D(\phi)$$

$\psi_{k,l}(n,m,q) : (n,m,q) \rightarrow (n,m+k-2,q+l-2)$ if $(i,j) \in D(\psi)$

$\theta_{k,l}(n,m,q) : (n,m,q) \rightarrow (n+1,m+k-2,q+l-2)$ if $(i,j) \in D(\theta)$

where $k,l \in \{1,2,3,4\}$

We further subdivide the transition probabilities according to a condition regarding the parameter q.

$\gamma_{h,k,l}(n,m,q)$: for $q=0$

$\delta_{h,k,l}(n,m,q)$: for $q=1,2,\ldots,n$ given the top output pin is freed

$\alpha_{h,k,l}(n,m,q)$: for $q=1,2,\ldots,n-2$ given a non-top output pin is freed

$\beta_{h,k,l}(n,m,q)$: for $q=n-1$, $n \geq 2$ given a non-top output pin is freed

where $h \in \{1,2,3\}$ indicates the effect of the transition on the parameter n: $n \rightarrow n+h-2$. The subscripts h,k,l correspond to the transition $(n,m,q) \rightarrow (n+h-2,m+k-2,q+l-2)$ For brevity, we omit the parameter list when the state is (n,m,q) and there is no confusion from the context in which it appears.

Let $\sigma(m,q)$ be the probability that the top output pin is released when a server completes in state $(\underline{n},m,q)$ for arbitrary $\underline{n}$. For the biased crossbar described here $\sigma(m,q) = 1/m$ if $q>0$ and 0 otherwise. Later in Section 4.3, we will consider a more general biased crossbar.

Also we define the function $\eta_a^b(q)=1$ if $a \leq q \leq b$ and 0 otherwise.

For clarity, we abbreviate $\eta_a^a(q)$ to $\eta_a(q)$.

Transition Probability Functions

When $(i,j) \in D(\phi)$ only event A is possible.

$$\phi_{k,l}(n,m,q) = \eta_0(q)\gamma_{1,k,l} + \sigma(m,q)\eta_1^n(q)\delta_{1,k,l} + (1-\sigma(m,q))\{\eta_1^{n-2}(q)\alpha_{1,k,l} + \eta_{n-1}(q)\beta_{1,k,l}\}$$

where $k, l \in \{1,2\}$,

$$\gamma_{1,1,2} = \frac{m-1}{n-1}, \qquad \gamma_{1,2,2} = \frac{n-m}{n-1},$$

$\delta_{1,2,1} = 1$ if $q=2,3,\ldots,n$, $\qquad \delta_{1,1,1} = 1$ if $q=1$,

$$\alpha_{1,1,2} = \frac{m-2}{n-q-1}, \qquad \alpha_{1,2,2} = \frac{n-q-m+1}{n-q-1},$$

$\beta_{1,1,2} = 1$

When $(i,j)\in D(\psi)$ event A and either event B or event C can occur.

$$\psi_{k,l}(n,m,q) = \eta_0(q)\gamma_{2,k,l} + \sigma(m,q)\eta_1^n(q)\delta_{2,k,l} + (1-\sigma(m,q))\{\eta_1^{n-2}(q)\alpha_{2,k,l} + \eta_{n-1}(q)\beta_{2,k,l}\}$$

where $k, l \in \{1,2,3\}$,

$$\gamma_{2,1,2} = \frac{(1-\omega)(m-1)^2}{(n-1)(a-1)} \text{ if } n\geq 2$$

$$\gamma_{2,2,2} = \frac{(1-\omega)}{(n-1)(a-1)}\{(m-1)(a-m)+(n-m)m\} \text{ if } n\geq 2 \quad = (1-\omega) \text{ if } n=1$$

$$\gamma_{2,3,2} = \frac{(1-\omega)}{(n-1)(a-1)}(n-m)(a-m-1) \text{ if } n\geq 2$$

$$\gamma_{2,2,3} = \frac{\omega(m-1)}{n-1} \text{ if } n\geq 2 \qquad = \omega \text{ if } n=1$$

$$\gamma_{2,3,3} = \frac{\omega(n-m)}{n-1} \text{ if } n\geq 2$$

$$\delta_{2,1,1} = \frac{(1-\omega)(m-1)}{(a-1)} \text{ if } q=1$$

$$\delta_{2,2,1} = \frac{(1-\omega)(m-1)}{(a-1)} \text{ if } q=2,3,\ldots,n \quad = \frac{(1-\omega)(a-m)}{(a-1)} \text{ if } q=1$$

$$\delta_{2,3,1} = \frac{(1-\omega)(a-m)}{(a-1)} \text{ if } q=2,3,\ldots,n$$

$$\delta_{2,2,2} = \omega$$

$$\alpha_{2,1,2} = \frac{(1-\omega)(m-2)^2}{(n-q-1)(a-1)}$$

$$\alpha_{2,2,2} = \frac{(1-\omega)}{(n-q-1)(a-1)}\{(n-q-m+1)(m-1)+(m-2)(a-m+1)\}$$

$$\alpha_{2,3,2} = \frac{(1-\omega)}{(n-q-1)(a-1)}(n-q-m+1)(a-m)$$

$$\alpha_{2,1,3} = \frac{\omega(m-2)}{n-q-1}, \qquad \alpha_{2,2,3} = \frac{\omega(n-q-m+1)}{n-q-1}$$

$$\beta_{2,2,2} = 1-\omega, \ \beta_{2,1,3} = \omega$$

When $(i,j)\in D(\theta)$ events A, B and C can occur.

$$\theta_{k,l}(n,m,q) = \eta_0(q)\gamma_{3,k,l} + \sigma(m,q)\eta_1^n(q)\delta_{3,k,l} + (1-\sigma(m,q))\{\eta_1^{n-2}(q)\alpha_{3,k,l} + \eta_{n-1}(q)\beta_{3,k,l}\}$$

where $k, l \in \{1,2,3,4\}$,

$$\gamma_{3,1,2} = \frac{(1-\omega)^2(m-1)^3}{(n-1)(a-1)^2}$$

$$\gamma_{3,2,2} = \frac{(1-\omega)^2}{(n-1)(a-1)^2}\{(n-m)m^2+m(m-1)(a-m)+(m-1)^2(a-m)\}$$

$$\gamma_{3,3,2} = \frac{(1-\omega)^2(a-m-1)}{(n-1)(a-1)^2}\{(n-m)(m+1)+(n-m)m+(m-1)(a-m)\}$$

$$\gamma_{3,4,2} = \frac{(1-\omega)^2(n-m)(a-m-1)(a-m-2)}{(n-1)(a-1)^2}$$

$$\gamma_{3,2,3} = \frac{\omega(1-\omega)(2m-1)}{a-1}$$

$$\gamma_{3,3,3} = \frac{\omega(1-\omega)(2a-2m-1)}{a-1}$$

$$\gamma_{3,2,4} = \omega^2$$

$$\delta_{3,1,1} = \frac{(1-\omega)^2(m-1)^2}{(a-1)^2} \text{ if } q=1$$

$$\delta_{3,2,1} = \frac{(1-\omega)^2(m-1)^2}{(a-1)^2} \text{ if } q=2,3,\ldots,n \qquad = \frac{(1-\omega)^2(a-m)(2m-1)}{(a-1)^2} \text{ if } q=1$$

$$\delta_{3,3,1} = \frac{(1-\omega)^2(a-m)(2m-1)}{(a-1)^2} \text{ if } q=2,3,\ldots,n \quad = \frac{(1-\omega)^2(a-m)(a-m-1)}{(a-1)^2} \text{ if } q=1$$

$$\delta_{3,4,1} = \frac{(1-\omega)^2(a-m)(a-m-1)}{(a-1)^2} \text{ if } q=2,3,\ldots,n$$

$$\delta_{3,2,2} = 2\omega(1-\omega)\frac{m-1}{a-1}, \qquad \delta_{3,3,2} = 2\omega(1-\omega)\frac{a-m}{a-1}$$

$$\delta_{3,2,3} = \omega^2$$

$$\alpha_{3,1,2} = \frac{(1-\omega)^2(m-2)^3}{(n-q-1)(a-1)^2}$$

$$\alpha_{3,2,2} = \frac{(1-\omega)^2}{(n-q-1)(a-1)^2}\{(n-q-m+1)(m-1)^2 + (m-1)(m-2)(a-m+1)+(m-2)^2(a-m+1)\}$$

$$\alpha_{3,3,2} = \frac{(1-\omega)^2(a-m)}{(n-q-1)(a-1)^2}\{(n-q-m+1)m+(n-q-m+1)(m-1)+(m-2)(a-m+1)\}$$

$$\alpha_{3,4,2} = \frac{(1-\omega)^2(n-q-m+1)(a-m)(a-m-1)}{(n-q-1)(a-1)^2}$$

$$\alpha_{3,1,3} = \frac{2\omega(1-\omega)(m-2)^2}{(n-q-1)(a-1)},$$

$$\alpha_{3,2,3} = \frac{2\omega(1-\omega)}{(n-q-1)(a-1)}\{(n-q-m+1)(m-1)+(m-2)(a-m+1)\}$$

$$\alpha_{3,3,3} = \frac{2\omega(1-\omega)}{(n-q-1)(a-1)}(n-q-m+1)(a-m)$$

$$\alpha_{3,1,4} = \frac{\omega^2(m-2)}{n-q-1}, \qquad \alpha_{3,2,4} = \frac{\omega^2(n-q-m+1)}{n-q-1}$$

$$\beta_{3,2,2} = \frac{(1-\omega)^2}{a-1}, \qquad \beta_{3,3,2} = \frac{(1-\omega)^2(a-2)}{a-1}$$

$$\beta_{3,1,3} = \omega^2, \qquad \beta_{3,2,3} = 2\omega(1-\omega)$$

4.1.2 Global Balance Equations

By analysing the process $W'(t)$ for a biased $b \times a$ crossbar with routing probability ω to pin zero, we obtain the following theorem.

THEOREM 4.1

The steady state probabilities $P(\underline{n},m,q)$ for a biased $b\times a$ crossbar described by the process $W'(t)$ with state space $\Omega_{W'}$ is given by the following balance equations:

$$P(\underline{n},m,q)m =$$

$$\frac{1}{b(n+1)} \sum_{(i,j)\ \in D(\phi)} \{ \sum_{k=1}^{2} \sum_{l=1}^{2} (m+2-k)\ P(\underline{n}^i_j,m+2-k,q+2-l)\ \phi_{k,l}(n+1,m+2-k,q+2-l) \}$$

$$+ \frac{1}{bn} \sum_{(i,j)\ \in D(\psi)} \{ \sum_{k=1}^{3} \sum_{l=1}^{3} (m+2-k)\ P(\underline{n}^i_j,m+2-k,q+2-l)\ \psi_{k,l}(n,m+2-k,q+2-l) \}$$

$$+ \frac{1}{b(n-1)} \sum_{(i,j)\ \in D(\theta)} \{ \sum_{k=1}^{4} \sum_{l=1}^{4} (m+2-k)\ P(\underline{n}^i_j,m+2-k,q+2-l)\ \theta_{k,l}(n-1,m+2-k,q+2-l) \}$$

PROOF

These balance equations follow from the transition probabilities given above in Section 4.1.1. The factor $1/b$ is the probability that the recycled task joins server j and $1/n'$ factor (where $n'=\#(\underline{n}^i_j)$) is the probability that server i completes given that there are n' active servers. □

Again we observe that the transition rates do not depend on the exact value of (i,j) but only on the domain to which (i,j) belongs. Thus we can partition the state space $\Omega_{W'}$ such that each partition has the same $\#(\underline{n})$. This now gives the process $X'(t)$ with state space given by:

$$\Omega_{X'} = \{(n,m,q) \mid n=1,2,\ldots,\min(b,N),\ [\ (q=0,\ 1\le m\le \min(a-1,n))$$
$$\text{or } (1\le q\le n-1,\ 2\le m\le \min(a,n-q+1)) \text{ or } (q=n,\ m=1)\ ]\ \}$$

By the same argument in Section 3.2.2, each state in a given partition is equally likely to occur. So if $P(n,m,q)$ is the steady state probability for the process $X'(t)$ we have:

$$P(\underline{n},m,q) = \frac{P(n,m,q)}{{}^{N-1}C_{n-1}\ {}^{b}C_{n}} \qquad \text{where } n=\#(\underline{n})$$

COROLLARY 1 TO THEOREM 4.1

For the process $X'(t)$ with state space $\Omega_{X'}$ which represents a biased $b\times a$ crossbar (routing probability ω to the top pin), the steady state probabilities $P(n,m,q)$ are given by:

$$P(n,m,q)m =$$

$$\frac{n^2}{b(N-1)} \sum_{k=1}^{2} \sum_{l=1}^{2} (m+2-k)\, P(n+1,m+2-k,q+2-l)\, \phi_{k,l}(n+1,m+2-k,q+2-l)$$

$$+ \frac{[n(N-n)+(b-n+1)(n-1)]}{b(N-1)} \sum_{k=1}^{3} \sum_{l=1}^{3} (m+2-k)\, P(n,m+2-k,q+2-l)\, \psi_{k,l}(n,m+2-k,q+2-l)$$

$$+ \frac{(N-n+1)(b-n+1)}{b(N-1)} \sum_{k=1}^{4} \sum_{l=1}^{4} (m+2-k)\, P(n-1,m+2-k,q+2-l)\, \theta_{k,l}(n-1,m+2-k,q+2-l)$$

PROOF

By substituting the above equation for $P(\underline{n},m,q)$ in the balance equations of Theorem 4.1 each summation on the rhs sums to a constant w.r.t. i and j. Hence we can replace the summation signs by the size of the summation domains in each given state $(\underline{n}_j^i,m',q')$ for any m' and q'.

First, let $E_1(n)$ be the expected number of singleton queues at the n active inputs when there are N tasks altogether.

$$E_1(n) = n \Pr(\text{an active server } i \text{ has a singleton queue})$$

$$= n \frac{\text{number of arrangements of } N-1 \text{ tasks on } n-1 \text{ non-empty queues}}{\text{number of arrangements of } N \text{ tasks on } n \text{ non-empty queues}}$$

$$= n \frac{{}^{N-2}C_{n-2}}{{}^{N-1}C_{n-1}} = \frac{n(n-1)}{N-1}$$

Now for a state with $n=\#(\underline{n})$ active inputs the size of the summation domains are derived as follows:

$$D(\phi) = \{i,j \mid n_i=0,\ n_j>1, 1\le i\ne j\le b\}$$

$$|D(\phi)| = (n-E_1(n))(b-n) = \frac{n(N-n)(b-n)}{N-1}$$

$$D(\psi) = \{i,j \mid n_i>0,\ n_j>1, 1\le i,j\le b\} \cup \{i,j \mid n_i=0,\ n_j=1, 1\le i,j\le b\} \cup \{i,i \mid n_i=1, 1\le i\le b\}$$

$$|D(\psi)| = (n-E_1(n))n + E_1(n)(b-n) + E_1(n)$$

$$= \frac{n}{N-1}\{n(N-n)+(b-n+1)(n-1)\}$$

$$D(\theta) = \{i,j \mid n_i>0,\ n_j=1, 1\le i\ne j\le b\}$$

$$|D(\theta)| = E_1(n)(n-1) = \frac{n(n-1)^2}{N-1}$$

The result follows by substituting these domain sizes (after appropriate adjustment to n depending on the transition type) into the balance equations of Theorem 4.1. □

4.1.3 Derivation of the Birth-Death Process

We apply the following probability generating function to the balance equations of process $X'(t)$:

$$G_{n,q}(y) = \sum_{m=1}^{\infty} P(n,m,q)y^m \text{ with first derivative } G_{n,q}'(y) = \sum_{m=1}^{\infty} m\,P(n,m,q)y^{m-1}$$

$$H_n(y,z) = \sum_{q=0}^{\infty} G_{n,q}(y)z^q \text{ and } H_n'(y,z) = \frac{\partial H_n(y,z)}{\partial y}$$

When $y=z=1$, all higher order partial derivatives cancel out and for $n=2,3,\ldots,b$ we obtain:

$$(N-n+1)(b-n+1)H_{n-1}'(1,1)$$

$$= (n-1)^2\{2G_{n,n-1}(1)+G_{n,n}(1)-G_{n,n-1}'(1)-G_{n,n}'(1)+H_n'(1,1)\}$$

But $q=n-1$ implies $m=2$, so $G_{n,n-1}'(1) = 2G_{n,n-1}(1)$
and $q=n$ implies $m=1$, so $G_{n,n}'(1) = G_{n,n}(1)$.

Hence the equation becomes:

$$(N-n+1)(b-n+1)H_{n-1}'(1,1) = (n-1)^2 H_n'(1,1) \qquad n=2,3,\ldots,b$$

This is exactly the same recurrence as (2.9) with $t_n = \mu_n p_n = H_n'(1,1)$ but we do not have the corresponding boundary condition (2.10).

$$H_n'(1,1) = \sum_{m=1}^{\infty}\sum_{q=0}^{\infty} mP(n,m,q) = P(n)\sum_{m=1}^{\infty}\sum_{q=0}^{\infty} mP(m,q \mid n)$$

and the throughput is given by $T(N) = \sum_{n=1}^{b} H_n'(1,1)$.

Our analysis of the simple birth-death process $Z(t)$ in Chapter 2 appears to be exact in the case of a biased crossbar. There must be a function μ_n which can be used with Theorem 2.1 or 2.2 to give exact throughput results (from the boundary condition). However, the presence of fourth order partial derivatives in the generating function balance equations means that complex analysis methods are hard to apply. For the $b \times a$

uniform crossbar, we hoped to solve the generating function and derive the boundary condition:

$$\sum_{n=1}^{b} \frac{t_n}{\mu_n} = 1 \quad \text{where } \mu_n = \frac{an}{a+n-1}$$

By expanding the generating function explicitly it is possible to show that this is true for a 2×2 crossbar, but for larger crossbars this approach is impractical.

4.1.4 Constrained Process

If the process $X'(t)$ is constrained in such a way that the number of active inputs remain fixed then we have b simpler processes to consider n=1,2,...,b. The n=b case represents a saturated crossbar (N=∞) with infinite queues at all inputs. Likewise, each constrained process represents an open model of the crossbar with n infinite queues at its inputs.

COROLLARY 2 TO THEOREM 4.1

The steady state probabilities $P(m,q \mid n)$ for the process $X'(t)$ constrained to keep n=1,2,...,b fixed is given by:

$$P(m,q \mid n)m = \sum_{k=1}^{3} \sum_{l=1}^{3} (m+2-k)\, P(m+2-k,q+2-l \mid n)\, \psi_{k,l}(n,m+2-k,q+2-l)$$

PROOF

Since n is fixed, only the middle summation in Theorem 4.1 is non-zero. The probability that an active server i completes is $1/n$ and the recycled task joins server j with probability $1/n$ (as it always joins one of the n non-empty queues). Since $|D(\psi)|=n^2$ (as there are no singleton queues), the coefficient to the summation is one and the above result follows. □

Unlike the uniform crossbar case, there appears to be no compact solution for the probabilities in general. In the next section we develop an approximate method to derive closed form probabilities for a biased crossbar.

4.2 Approximate Probability Model for a Non-uniform Crossbar

Since a compact solution for the constrained process of a biased crossbar was difficult to obtain, we present an approximate probability model instead. The approach uses the properties of the uniform crossbar in equilibrium (Section 3.5) to derive results for the

non-uniform case. Thus, the results are exact when the biased crossbar has $\omega=1/a$ and very accurate when it is almost uniform (ω close to $1/a$). We start by considering the static probabilities of the non-uniform crossbar.

4.2.1 Static Probabilities for a Non-uniform Crossbar

We can analyse the biased crossbar in a similar way to Section 3.5 by extending the state space to $\Omega_{X'}$ and considering the zeroth order Markov chain for the constrained process $X'(t)$. Alternatively, we can make use of the results for the uniform crossbar and extend them to the biased case. We adopt the latter method and obtain the following theorem.

THEOREM 4.2

The static probability that m outputs are active and q of the n tasks request the top pin in a biased $b\times a$ crossbar with top pin routing probability ω is given by:

$$P_a(m,q \mid n) = {}^nC_q\ \omega^q(1-\omega)^{n-q}\ P_{a-1}(m-\eta_1^n(q) \mid n-q)$$

$(n=1,2,\ldots,b)$,

$(q=0,\ 1\le m\le \min(a-1,n))$ or $(1\le q\le n-1,\ 2\le m\le \min(a,n-q+1))$ or $(q=n,\ m=1)$

where $P_a(m \mid n)$ is the static probability for the uniform crossbar.

PROOF

Since $a-1$ outputs are uniform, we can use Theorem 4.2 probabilities and incorporate the following:

- probability ω^q that q of the n tasks select the top pin;
- probability $(1-\omega)^{n-q}$ that the remaining requests select other outputs;
- the number of ways ${}^nC_q$ that q pin zero requests are chosen from the n requests.

Given that the static probabilities for the uniform crossbar sum to one, the above probabilities also sum to one since the coefficients are from the binomial expansion of $(\omega+(1-\omega))^n$. □

4.2.2 Transformed Static Probabilities

So far in this analysis we have ignored state holding times and assumed that all state transitions are independent of the initial state. These omissions can be corrected by

multiplying the static probabilities for a uniform crossbar by a state dependent factor R_m (with R_1=1) and re-normalising the new transformed probabilities by $E(R)$. We already know the steady state probabilities of the constrained process $X(t)$ for a uniform crossbar and so we can use this to determine the factors R_m. Using (3.5) we obtain:

$$P_a(m \mid n)\frac{R_m}{E(R)} = \frac{{}^{n-1}C_{m-1}\,{}^{a}C_m}{{}^{a+n-1}C_n} \qquad m=1,2,\ldots,\min(n,a)$$

Substituting in Theorem 3.2, we have:

$$E(R) = \frac{1}{a^n}\,{}^{a+n-1}C_n \text{ and } R_m = \frac{{}^{n-1}C_{m-1}}{m!\,S(n,n-m)} \qquad m=1,2,\ldots,\min(n,a)$$

We conjecture that the factors R_m are the same for both the uniform and biased crossbar and not dependent on q. This is clearly approximate but should give accurate results when ω is close to $1/a$. The intuitive argument that leads to this postulate is as follows:

- The mean state holding time for both the uniform and non-uniform crossbar is the same when there are m active outputs.
- The non-uniformity in routing is accounted for in the static probabilities by the factor $\omega^q(1-\omega)^{n-q}$.
- The transitions have a similar form in both the biased and uniform crossbar

The transformed static probabilities for a biased $b\times a$ crossbar with top pin routing probability ω is given by:

$$P_a^*(m,q \mid n) = \frac{{}^{n}C_q\,\omega^q(1-\omega)^{n-q}}{E^*(R)} P_{a-1}(m-\eta_1^n(q) \mid n-q)\frac{{}^{n-1}C_{m-1}}{m!\,S(n,n-m)}$$

where $E^*(R)$ is the normalising constant.

The conditional throughput function is then given by:

$$\mu_n = \sum_{m=1}^{n}\sum_{q=0}^{n-m+1} m\,P_a^*(m,q \mid n)$$

4.3 Non-uniform Routing with Different Pin Release Times

So far we have assumed that the holding time for all crossbar output pins has the identical negative exponential distribution with rate μ. However, there are cases (such as the hot-spot analysis in Chapter 6) where certain output pins may have different

mean holding times (*release* times). Here we consider the case in which all pins but the top output pin of the crossbar have the same release time. All holding times are still negative exponential and the crossbar is biased so that ρ is the top pin routing probability. The actual values of the release times are not required to model this crossbar – only their ratio. Let the top output pin have a release time of one and all other pins a release time of r so that r is the release time ratio.

4.3.1 Constrained Process

We consider the embedded Markov chain of the process $X'(t)$ for a biased $b \times a$ crossbar. Since state holding times are not relevant to the Markov chain, this makes the analysis easier but the release times are required to calculate $\sigma(m,q)$: the probability that the top pin is released when a transition occurs.

$$\text{For } q>0,\ \sigma(m,q) = \frac{\text{release rate of top pin}}{\text{total release rate}} = \frac{1}{1+(m-1)/r} = \frac{r}{r+m-1}$$

$$\text{and} \qquad \sigma(m,0) = 0$$

From the stationary Markov chain probabilities $P^{(C)}(m,q \mid n)$, the stationary Markov process probabilities $P(m,q \mid n)$ can be determined by multiplying by the release time.

$$P(m,q \mid n) \propto \frac{r}{m} P^{(C)}(m,q \mid n) \qquad \text{if } q=0$$

$$\propto \frac{r}{r+m-1} P^{(C)}(m,q \mid n) \qquad \text{if } q>0$$

This analysis leads to the following theorem:

THEOREM 4.3

For a biased $b \times a$ crossbar with top pin routing probability ρ and release time ratio r, the steady state constrained probabilities $P(m,q \mid n)$ for the process $X'(t)$ with state space $\Omega_{x'}$ are given by the following balance equations:

$$\frac{m}{r} P(m,0 \mid n) =$$

$$P(m+1,1 \mid n) \frac{(1-\rho)m}{a-1} + \frac{m+1}{r} P(m+1,0 \mid n) \frac{(1-\rho)m^2}{(n-1)(a-1)}$$

$$+ P(m,1 \mid n) \frac{(a-m)(1-\rho)}{a-1} + \frac{m}{r} P(m,0 \mid n) \frac{(1-\rho)}{(n-1)(a-1)} \{(m-1)(a-m)+(n-m)m\}$$

$$+ \frac{m-1}{r} P(m-1,0 \mid n) \frac{(1-\rho)(n-m+1)(a-m)}{(n-1)(a-1)}$$

For $q>0$,

$$\frac{r+m-1}{r}P(m,q \mid n) =$$

$$\frac{m}{r}P(m+1,q \mid n)\,\eta_1^{n-2}(q)\,\frac{(1-\rho)(m-1)^2}{(n-q-1)(a-1)}$$

$$+\frac{m}{r}P(m+1,q-1 \mid n)\,\{\eta_1^{n-2}(q-1)\,\frac{m-1}{a-1}+\eta_{n-1}(q-1)\}$$

$$+P(m,q+1 \mid n)\,\eta_2^{n}(q+1)\,\frac{(1-\rho)(m-1)}{a-1}$$

$$+P(m,q \mid n)\{\eta_1^{n}(q)\rho + \frac{m-1}{r}[\frac{\eta_1^{n-2}(q)(1-\rho)}{(n-q-1)(a-1)}\{(n-q-m+1)(m-1)+(m-2)(a-m+1)\}$$

$$+\eta_{n-1}(q)(1-\rho)]\}$$

$$+\frac{m}{r}P(m,q-1 \mid n)\,\eta_0(q-1)\,\frac{\rho(m-1)}{a-1} + \frac{m-1}{r}P(m,q-1 \mid n)\,\eta_1^{n-2}(q-1)\rho\frac{n-q+m+1}{n-q}$$

$$+P(m-1,q-1 \mid n)\,\eta_2^{n}(q-1)\,\frac{(1-\rho)(a-m+1)}{a-1}$$

$$+\frac{m-1}{r}P(m-1,q \mid n)\,\eta_1^{n-2}(q)(1-\rho)\frac{(n-q-m+2)(a-m+1)}{(n-q-1)(a-1)}$$

$$+\frac{m}{r}P(m-1,q-1 \mid n)\,\eta_0(q-1)\rho\frac{(n-m+1)}{n-1}$$

PROOF

These balance equations can be derived from Corollary 2 of Theorem 4.1 by converting to the embedded Markov chain balance equations; multiplying the coefficients of the probabilities by the state release rates and substituting for $\sigma(m.q)$ in the transition probabilities $\psi_{k,l}$. □

4.3.2 Analysis of a 2 by 2 Crossbar

When $a=b=2$ in Theorem 4.3 we obtain the following.

COROLLARY TO THEOREM 4.3

For a 2×2 biased crossbar with top pin routing probability ρ and release time ratio r, the steady state constrained probabilities $P(m,q \mid n)$ are given by the following equations:

n=1:	$(1-\rho)r\,P(1,1 \mid 1)$	$=\rho\,P(1,0 \mid 1)$
n=2:	$\rho\,P(1,0 \mid 2)$	$=r(1-\rho)\,P(2,1 \mid 2)$
	$r(1-\rho)\,P(1,2 \mid 2)$	$=\rho\,P(2,1 \mid 2)$
	$(r(1-\rho)+\rho)\,P(2,1 \mid 2)$	$=\rho\,P(1,0 \mid 2)+r(1-\rho)\,P(1,2 \mid 2)$

PROOF

By direct substitution into Theorem 4.3 and simplification. □

Note that all coefficients of the probabilities have factors ρ, $r(1-\rho)$ or their sum. This implies that we can replace these factors by:

$$\omega = \frac{\rho}{\rho+r(1-\rho)} \qquad \text{and} \qquad 1-\omega = \frac{r(1-\rho)}{\rho+r(1-\rho)}$$

without changing the steady state probabilities. Thus for every pair (ρ,r) of top pin routing probability and release time ratio of a 2×2 crossbar there exists a unique top pin routing probability ω in an equivalent crossbar with unit release time ratio which gives the same steady state probabilities. This simplification is not possible for larger crossbars, as can be seen from Theorem 4.3, because the factors ρ and $r(1-\rho)$ do not appear in *all* terms in the balance equations.

4.3.3 Output Pin Utilisation in a Biased Crossbar

As with the uniform case, we consider the crossbar only at times when there are a given number of customers at the inputs. Hence we require the following conditional probability distribution:

$$W_i(z \mid n,\rho,r) = \Pr(Z_i=z \mid I=n) \qquad i=0,1,\ldots,a-1$$

which is the steady state probability that output pin i of the switch is in state z given the following:

- There are n active inputs to this switch.
- The routing probability to the top switch output pin is ρ.
- The release time for non-top output pins is r times that of the top output pin of the switch.

We can derive expressions W_i by analysing a closed embedded sub-chain of the Markov process $X'(t)$ in which one state parameter, representing the number of active inputs, is kept constant. The balance equations of this constrained process are given in Theorem 4.3. The corresponding steady state probabilities lead to the following theorem.

THEOREM 4.4

For a constrained $b \times a$ crossbar with $n=0,1,\ldots,b$ active inputs, top pin routing probability ρ, and release time ratio r, the probability that output pin i is active is given by W_i:

$$W_0(1 \mid n,\rho,r) = 1 - \sum_{m=1}^{n} P(m,0 \mid n)$$

for $i=1,2,\ldots,a-1$

$$W_i(1 \mid n,\rho,r) = \frac{1}{a-1}\left[\sum_{m,q} mP(m,q \mid n) + \sum_{m=1}^{n} P(m,0 \mid n) - 1\right]$$

where $P(m,q \mid n)$ is given by Theorem 4.3.

PROOF

The first expression follows from the marginal probability that $q>0$, and the second follows from the expected number of active output pins:

$$\sum_{m,q} mP(m,q \mid n) = W_0(1 \mid n,\rho,r) + (a-1)\, W_1(1 \mid n,\rho,r) \square$$

COROLLARY TO THEOREM 4.4

For a 2×2 crossbar with $n \in \{0,1,2\}$ active inputs, top pin routing probability ρ, and release time ratio r, the probability that output pin i is active is given by W_i:

$$W_i(1|0,\rho,r) = 0$$

$$W_0(1|1,\rho,r) = \frac{\rho}{\rho+(1-\rho)r}$$

$$W_1(1|1,\rho,r) = \frac{(1-\rho)r}{\rho+(1-\rho)r}$$

$$W_0(1|2,\rho,r) = \frac{\rho^2 + \rho(1-\rho)r}{\rho^2 + \rho(1-\rho)r + (1-\rho)^2r^2}$$

$$W_1(1|2,\rho,r) = \frac{(1-\rho)^2r^2 + \rho(1-\rho)r}{\rho^2 + \rho(1-\rho)r + (1-\rho)^2r^2}$$

PROOF

For n=1, the solution of the balance equations in the Corollary to Theorem 4.3 gives the following:

$$P(1,1 \mid 1) = \frac{\rho}{\rho+(1-\rho)r} = W_0(1|1,\rho,r)$$

and

$$P(1,0 \mid 1) = \frac{(1-\rho)r}{\rho+(1-\rho)r} = W_1(1|1,\rho,r)$$

For n=2, the solution to the balance equations in the same Corollary gives the following:

$$P(1,0 \mid 2) = \frac{(1-\rho)^2r^2}{\rho^2 + \rho(1-\rho)r + (1-\rho)^2r^2}$$

$$P(1,2 \mid 2) = \frac{\rho^2}{\rho^2 + \rho(1-\rho)r + (1-\rho)^2r^2}$$

$$P(2,1 \mid 2) = \frac{\rho(1-\rho)r}{\rho^2 + \rho(1-\rho)r + (1-\rho)^2r^2}$$

Thus by Theorem 4.4,

$$W_0(1|2,\rho,r) = P(1,2 \mid 2) + P(2,1 \mid 2)$$

$$W_1(1|2,\rho,r) = P(1,0 \mid 2) + P(2,1 \mid 2) \square$$

The analysis of crossbars with non-unit release time ratio will be particularly useful in analysing hot-spots in delta-2 networks in Chapter 6. But first let us consider models of delta networks without hot-spots.

Chapter 5

Uniform Delta Networks

Outline

5.1 Exact Analysis of Delta Networks
- 5.1.1 Definitions
- 5.1.2 A Crossbar in Equilibrium
- 5.1.3 A Delta Network in Equilibrium

5.2 Analysis of Delta-2 Networks
- 5.2.1 The Exponential Assumption and the Arriving Observer Property
- 5.2.2 A 2 by 2 Crossbar in Equilibrium
- 5.2.3 Recursive Analysis of Delta-2 Networks
- 5.2.4 Saturated Delta Networks

5.3 Approximate Analysis of General Delta Networks
- 5.3.1 Definitions and Assumptions
- 5.3.2 Recursive Analysis of General Delta Networks
- 5.3.3 Saturated Delta Networks

Thus far we have considered closed systems of DMA servers connected to a switching network which is a simple crossbar. In this chapter, we extend this analysis to a delta network with uniform routing to the output ports. Our aim is to derive an expression for the conditional throughput of the delta network, i.e. throughput given the number of active inputs. This expression describes the network characteristics and can be used to determine throughput as a function of the closed system population (using Theorem 2.1) and hence the FES rate of the composite server.

By exploiting the recursive structure of the network we obtain a recurrence for the conditional throughput of a J-stage delta network constructed from $b \times a$ crossbars. The result holds for asynchronous, circuit-switched delta networks in which partial paths are held (as in the ALICE machine). For this operational mode, we also determine an upper bound on throughput of the network which is useful for comparison with the throughput of models describing other protocols.

We begin with a general analysis of delta networks by applying Little's result [Li61], and so initially require no assumptions about distributions of task service times, only that the network is in stochastic equilibrium. In order to obtain a recursive solution, however, we need to make certain approximating assumptions to determine the mean queueing time for tasks arriving at the crossbars. These assumptions are very mild, especially in the case of a saturated network, but only applicable to delta-2 networks. In order to analyse more general networks, we require a further assumption, viz. that the sub-networks in the delta network behave independently. This is a much coarser approximation, although it becomes more accurate as the size of the sub-networks increases.

5.1 Exact Analysis of Delta Networks

In this section we define some terminology for $b \times a$ crossbars and determine the output pin utilisations in terms of the input pin utilisations (by using Little's result [Li61]). By applying these results to crossbars in each stage, we can determine the network output pin utilisations (and hence throughput) in terms of the network input pin utilisations. However, two parameters for each crossbar in the network are unknown and need to be estimated. This will be the subject of Section 5.2 for 2×2 crossbars, but for now we require the following definitions.

5.1.1 Definitions

Let the state of a $b \times a$ crossbar be the binary $(b+a)$-tuple $(I_0,I_1,\ldots,I_{b-1}; U_0,U_1,\ldots,U_{a-1})$ describing the states (active or inactive) of its input pins $I_0,I_1,\ldots,I_{b-1}$ and output pins $U_0,U_1,\ldots,U_{a-1}$. In the steady state, we define the following:

- A pin's *mean holding time* (MHT) is the expected elapsed time between the pin becoming active and the departure of the task holding it after its service completion (i.e. the pin next becoming inactive, even if instantaneously).
- An active output pin's *mean residual holding time* (MRHT) is the expected elapsed time between an arrival at the crossbar and the instant that the arrival acquires that pin.
- The *blocking probability*, B_i, for input pin i (i=0,1,...,b-1) is the steady state probability that, at the instant a task arrives on that pin, the output pin it requires is already held by a task currently holding another input pin.

5.1.2 A Crossbar in Equilibrium

LEMMA 5.1

For a $b \times a$ crossbar with output pins that

(a) are selected with the same probability by incoming traffic

(b) have the same MHT, m, and MRHT, d,

the probability that the top output pin is active is

$$p = \frac{m}{a} \sum_{i=0}^{b-1} \frac{\pi_i}{m+B_i d}$$

where π_i is the utilisation of input pin i (i=0,1,...,b–1).

PROOF

Consider Figure 5.1 below.

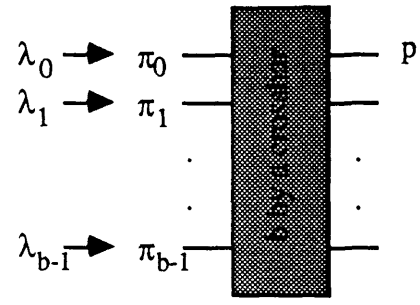


Figure 5.1 Pin Utilisations in a *b* by *a* Crossbar

Let the arrival rates (i.e. reciprocals of the mean inter-arrival times) on input pin i be λ_i. Applications of Little's result then yield:

(i) For an output pin, $p = \left[\frac{\lambda_0+\lambda_1+ \ldots + \lambda_{b-1}}{a}\right]m$, since the probability that the pin is active is the same as its mean queue length;

(ii) For input pin i, $\pi_i = \lambda_i(m+B_i d)$, since the mean queueing time for a task is the sum of the delay d in acquiring the output (which occurs with probability B_i) and the mean pin holding time m.

The result follows by eliminating the λ_i's. □

5.1.3 A Delta Network in Equilibrium

The steady state probability that a given output pin in any stage of a delta network satisfies a recurrence relation which can be derived by considering a single crossbar, using Lemma 5.1, and the network's recursive structure.

Let m_s, d_s be the MHT and MRHT respectively for output pins at stage s in a uniform delta network with J stages ($1 \le s \le J$). Thus $m_J = d_J = \mu^{-1}$ if all servers have exponential service times with parameter μ. Let β_s be the blocking probability for an arrival on any input pin at stage s. Then we have the following:

THEOREM 5.1

The equilibrium probability π_s that an output pin (of any crossbar) in stage s of a J-stage delta network is active satisfies

$$\pi_s = \frac{b\pi_{s-1}}{a(1+\beta_s\alpha_s)} \qquad s=1,2,\ldots,J$$

where $\alpha_s = \frac{d_s}{m_s}$ and π_0 is the equilibrium probability that an input pin in stage 1 is active.

PROOF

Any crossbar in stage s (s=1,2,...,J) satisfies the conditions of Lemma 5.1 since its output pins are stochastically identical and clearly a steady state exists since the Markov process representing the state of *every* pin in the network is Ergodic. □

5.2 Analysis of Delta-2 Networks

In order to proceed further in the analysis we will need to make some approximating assumptions so that the blocking probabilities for new arrivals and mean residual holding times can be determined. We restrict the analysis to delta-2 networks in which case we believe the approximations are very mild. The 2×2 crossbar is analysed first so that it can be used as the base case in the recursive analysis of the delta-2 network. This leads to a recurrence for the conditional throughput function which can then be used in the birth and death process of Chapter 2 to determine the throughput of the closed system enclosing the network.

5.2.1 The Exponential Assumption and the Arriving Observer Property

First of all we define the arriving observer property: a 2×2 crossbar is said to possess the *arriving observer property* (AOP) if the steady state probability that an input pin is active at the instant a task arrives on the other input pin is equal to the equilibrium (marginal) probability that the former input pin is active.

We make the following simplifying assumptions:

(i) For all pins in the network, MHT = MRHT, i.e. that $m_s = d_s$. This assumption holds if *all* pin holding time distributions are exponential, which is the case in the final stage of a network connected to exponential servers but not true in general.

(ii) The AOP holds for all crossbars in the network. So the probability that an arrival at an input pin in stage s is blocked is the product of the probability that the other crossbar input is active (i.e. π_{s-1}) and the conditional probability that the new arrival selects the active output (i.e. 1/2). Thus, $\beta_s = \frac{1}{2}\pi_{s-1}$

Under these intuitively reasonable assumptions, the recurrence in Proposition 5.1 becomes solvable in closed form for the saturated, uniform case. In general, these assumptions are both approximations in that they do not hold at every stage of the network.

If we assume that all DMA servers have exponential service times, then all holding time distributions are mixtures of Erlang distributions with the same parameter. Thus if arrivals at any crossbar's input pin are *random* with respect to the holding times of the other input pin, $\frac{1}{2} \leq \frac{d_s}{m_s} \leq 1$ since the mean residual service time for an Erlang-k distribution with parameter μ is $\frac{k+1}{2\mu}$. In fact arrivals are not random and the MRHT is much closer to the bound given by our approximation. This is because the service completion that caused the arrival may have also just unblocked another task and so the new arrival may be blocked at a pin which has just become active. In this case we have $d_s=m_s$, but we may also have $d_s>m_s$ if an arrival at one pin tends to occur more frequently during longer intervals of activity of the other pin. Since the pin holding time distribution seen by an arrival is not known, it is difficult to determine whether $d_s<m_s$ or $d_s>m_s$. However, the assumption that $d_s=m_s$ certainly appears very reasonable.

Intuitively, the AOP should not hold exactly, but should provide a good approximation. In Appendix A we compare the approximate method with the exact solution for a two-stage saturated, uniform network and show that the AOP does not hold by deriving explicitly the steady state probability that an input pin is active at the instant a task arrives on the other input pin of a crossbar in the final stage. However, the approximating assumptions do turn out to be mild and yield extremely accurate results compared with the exact solution in the two stage case and with simulations of larger networks.

5.2.2 A 2 by 2 Crossbar in Equilibrium

COROLLARY TO LEMMA 5.1

Assuming that the AOP holds and that $m = d$ in a crossbar with inputs that have utilisations π_0 and π_1, the output pin utilisation p satisfies:

$$p = \frac{\pi_0}{2+\pi_1} + \frac{\pi_1}{2+\pi_0}$$

PROOF

Since output pins are selected with equal probability and the AOP holds, $B_i = \frac{1}{2}\pi_{1-i}$ (i=0,1). The result follows from Lemma 5.1 after substitution. □

For a crossbar with $\pi = \pi_0 = \pi_1$, e.g. one in which the input processes are the same for each input pin, which has the AOP and $m = d$,

$$p = \frac{2\pi}{2+\pi}$$

If further the crossbar is *saturated* (i.e $\pi = 1$), $p = \frac{2}{3}$

5.2.3 Recursive Analysis of Delta-2 Networks

In this section we derive the throughput of a delta-2 network with uniform routing, conditional on the number of active inputs to the network. For an s-stage delta-2 network ($s \geq 1$) we define the set $V_s = \{(n_0,...,n_{2^s-1}) \mid n_i \in \{0,1\}, 0 \leq i < 2^s\}$ and the following random variables:

(i) $\underline{N} \in V_s$ which represents the state of the input pins to the network (N_i=0 means that input pin i is inactive and N_i=1 means that it is active, i.e. the DMA server connected to it is wishing to transmit).

(ii) $\underline{Z} \in V_s$ which represents the state of each network output pin.

We also define the function # for which #($\underline{V}$) is the number of non-zero components in the vector $\underline{V}$ and the analogous function $\#_1$ for which $\#_1(\underline{V})$ is the number of non-zero components in the *first half* of the vector $\underline{V}$.

We will require the following probability distribution: $Q_s(i|n) = \Pr(\#_1(\underline{N})=i \mid \#(\underline{N})=n)$ which is the probability that i active inputs are in the upper part of the s-stage network (i.e. the first 2^{s-1} pins) given that there are n active inputs altogether. By symmetry, when there are n active inputs to the network, they are uniformly distributed over the 2^s input pins. Since all arrangements of the n active inputs are equally likely:-

$$Q_s(i \mid n) = \frac{{}^kC_i \, {}^kC_{n-i}}{{}^{2k}C_n} \qquad \text{where } k = 2^{s-1} \qquad (5.1)$$

We also define the following probability distribution:

$$T_s(n) = \Pr(Z_0=1 \mid \#(\underline{N})=n)$$

which represents the probability that the topmost output pin of the s-stage network is active given that there are n active network inputs. This distribution can be defined by using the recursive structure of the delta network with the partial shuffle topology.

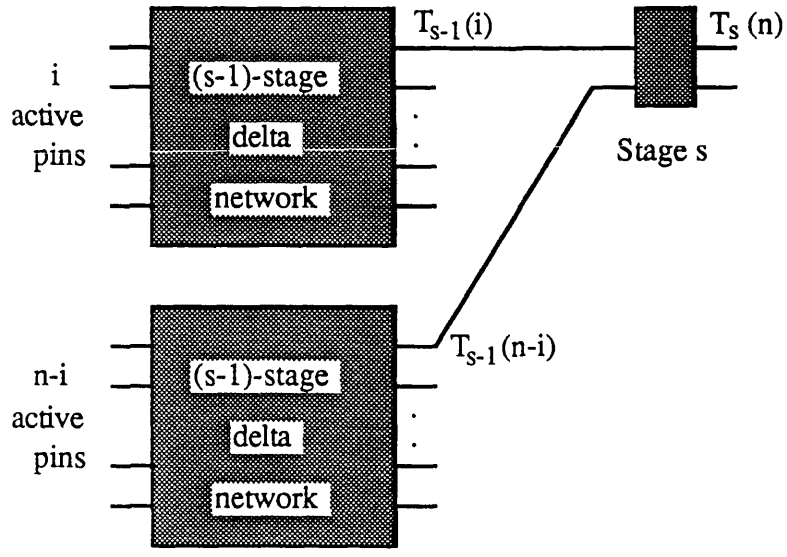


Figure 5.2 Recursive Structure of s-stage Delta-2 Networks

Consider the rightmost, topmost switch in an s-stage network. This takes its inputs from two separate $(s-1)$-stage sub-networks (Figure 5.2). When we condition on the number of active inputs to each sub-network (their sum being $\#(\underline{N})=n$), the output pin utilisation of either network can be found. Hence, we can determine the utilisation of both the inputs to the rightmost, topmost crossbar by using the result obtained for the next smaller networks in the recursion. The distribution of the state of the top output pin then follows by considering a single crossbar and the results of the previous sections. This gives the recurrence formula in the following theorem.

THEOREM 5.2

Under the assumptions that every crossbar in the network has the AOP and outputs with MRHT = MHT, the output pin utilisation of a s-stage delta-2 network conditional on the number of active inputs being n is given by the following:-

$$\text{For } s>1, \qquad T_s(n) = \sum_{i=\max(0,n-2^{s-1})}^{\min(n,2^{s-1})} Q_s(i|n)\, U(T_{s-1}(i), T_{s-1}(n-i)) \qquad (5.2)$$

$$T_1(0) = 0, \qquad T_1(1) = U(0,1), \qquad T_1(2) = U(1,1) \tag{5.3}$$

where $$U(\pi_0,\pi_1) = \frac{\pi_0}{2+\pi_1} + \frac{\pi_1}{2+\pi_0}$$

PROOF

For an s-stage delta network (s>1) we define the following random variables (see Figure 5.2):-

$Z \in \{0,1\}$ the state of output pin 0 of an s-stage network

$I_0 \in \{0,1,2,..2^{s-1}\}$ the number of active inputs to the upper (s-1)-stage sub-network

$I_1 \in \{0,1,2,..2^{s-1}\}$ the number of active inputs to the lower (s-1)-stage sub-network

$$\Pr(Z=1, I_0+I_1=n) = \sum_{i=0}^{n} \Pr(Z=1, I_0=i, I_1=n-i)$$

$$= \sum_{i=0}^{n} \Pr(I_0=i, I_1=n-i) \Pr(Z=1 \mid I_0=i, I_1=n-i)$$

$$\Pr(Z=1 \mid I_0+I_1=n) = \sum_{i=0}^{n} \Pr(I_0=i, I_1=n-i \mid I_0+I_1=n) \Pr(Z=1 \mid I_0=\mathrm{i}, I_1=n-i)$$

The recurrence (5.2) follows from the following substitutions:–

$$T_s(n) = \Pr(Z=1 \mid I_0+I_1=n)$$

$$Q_s(i \mid n) = \Pr(I_0=i, I_1=n-i \mid I_0+I_1=n)$$

$$U(T_{s-1}(i), T_{s-1}(n-i)) = \Pr(Z=1 \mid I_0=i, I_1=n-i)$$

where $U(\pi_0,\pi_1)$ (the crossbar output pin utilisation when the crossbar inputs have utilisations π_0 and π_1) is p in the Corollary to Lemma 5.1. □

COROLLARY 1 TO THEOREM 5.1

The expected number of active output pins conditional on n active inputs to a J-stage network is given by:-

$$E(m|n) = 2^J T_J(n) \tag{5.4}$$

From the theorem and its corollary, the required conditional throughput function $\mu_n=\mu E(m|n)$ can be used to solve the birth and death process defined in Chapter 2.

5.2.4 Saturated Delta Networks

The delta network becomes saturated when the queue lengths at each of its inputs grows very large. This occurs as the number of customers in the closed system of DMA servers and delta network approaches infinity.

COROLLARY 2 TO THEOREM 5.2

For a saturated delta-2 network, p_s, the probability that the top pin in stage s is active is given by the following:-

$$p_s = \frac{2}{s+2} \tag{5.5}$$

and the expected number of active outputs in a J-stage network is given by:-

$$E(m \mid 2^J) = \frac{2^{J+1}}{J+2} \tag{5.6}$$

PROOF

Since all input pins to the network are active the recurrence in Theorem 5.2 simplifies considerably in the absence of splitting probabilities $Q_s(i|n)$ since $Q_s(2^{s-1} \mid 2^s)=1$ (from (5.1))

Consider an s-stage delta network with all 2^s inputs active. The (s-1)-stage sub-networks are also saturated (see Figure 5.2) and so the recurrence (5.2) in Theorem 5.2 simplifies to give the following:-

$$p_s = \frac{2p_{s-1}}{2+p_{s-1}} \quad \text{where } p_s = T_s(2^s)$$

This recurrence has solution (5.5), and (5.6) follows from Corollary 1 of Theorem 5.2. □

For a 2-stage, saturated delta-2 network with μ=1, we find $p_2 = \frac{1}{2}$ using (5.5). Thus, the throughput of the saturated network is 2. This compares with a figure of $2^2/7$ for a full crossbar network.

Although the recurrence (5.2) has only been simplified under these rather restricted conditions, these are of course the conditions of greatest interest, viz. high loading.

5.3 Approximate Analysis of General Delta Networks

In this final section on uniform delta networks, we present an approximate analysis of delta networks of any dimension. The main approximation arises from the assumption that sub-networks in the network behave independently. We briefly discuss this approximation in the following sub-section before presenting the main recursive analysis for the non-saturated and saturated cases.

5.3.1 Definitions and Assumptions

We consider delta networks with a partial shuffle topology, so an s-stage delta network with $b\times a$ crossbars consists of b $(s-1)$-stage sub-networks (labelled $0,1,\ldots,b-1$ from the top as in Figure 5.3). By symmetry, all these sub-networks have identical statistical properties and so we only need to determine properties of one representative sub-network (the one labelled 0, for example) and the top crossbar in stage s.

For an s-stage delta network ($s\geq 1$) constructed using $b\times a$ crossbars, we define the following set:

$$V_s(c) = \{(n_0,\ldots,n_{c^s-1}) \mid n_i \in \{0,1\},\ i=0,1,\ldots,c^s-1\}$$

and the following random variables:

(i) $\underline{N}(s) \in V_s(b)$ which represents the state of the input pins to the network ($N_i(s)=0$ means that input pin i is inactive and $N_i(s)=1$ means that it is active, i.e. the DMA server connected to it is wishing to transmit).

(ii) $\underline{Z}(s) \in V_s(a)$ which represents the state of each network output pin.

(iii) $\underline{Z}(s,k) \in V_{s-1}(a)$ which represents the state of the output pins of sub-network k in an s-stage network.

We also define the function $\#_k$ ($k=0,1,\ldots,b-1$) which takes a single parameter of the form $\underline{N}(s)$ and returns the number of non-zero components in the $(k+1)^{\text{th}}$ partition of the vector $\underline{N}(s)$ i.e. in the vector $N_{kb^{s-1}}(s),\ldots,N_{(k+1)b^{s-1}-1}(s)$.

We will require the following probability distribution:

$$Q_s((i_0,i_1,\ldots,i_{b-1}) \mid n) = \Pr(\#_0(\underline{N}(s))=i_0,\#_1(\underline{N}(s))=i_1,\ldots,\#_{b-1}(\underline{N}(s))=i_{b-1} \mid \#(\underline{N})=n)$$

which is the probability that i_k active inputs are in sub-network k ($k=0,1,\ldots,b-1$) given that there are n active inputs altogether. By symmetry, when there are n active inputs to the network, they are uniformly distributed over the b^s input pins. Since all

arrangements of the n active inputs are equally likely:–

$$Q_s(\underline{i} \mid n) = \frac{b^{s-1}C_{i_0}\, b^{s-1}C_{i_1} \cdots b^{s-1}C_{i_{b-1}}}{b^sC_n} = \frac{\prod_{k=0}^{b-1} b^{s-1}C_{i_k}}{b^sC_n}$$

We also define the following probability distribution:

$$T_s(z \mid n) = \Pr(Z_0(s)=z \mid \#(\underline{N}(s))=n)$$

which represents the probability that the topmost output pin of the s-stage network is in state z given that there are n active inputs to the network.

In the approximate analysis we make the following assumptions:

- All pin holding times are exponentially distributed.
- The input processes to each input pin of a crossbar are independent.

These assumptions are stronger than those given in the analysis of delta-2 networks. In fact they imply both MRHT = MHT for all pins and that the AOP holds for crossbars. The crucial assumption in this analysis is that sub-networks in the recursive definition of delta network behave independently. This assumption is exact for synchronous protocols but for the asynchronous protocols that we are concerned with, it does not hold because downstream blocking affects both sub-networks. For example, if one sub-network becomes heavily loaded, then tasks in some other sub-network are more likely to be blocked further downstream thereby increasing the utilisation of that sub-network's output pins.

5.3.2 Recursive Analysis of General Delta Networks

In this section we derive an approximate expression for the conditional throughput (μ_n) of a J-stage delta network constructed from $b \times a$ crossbars. We consider the decode tree of the top output pin in this network since all outputs are identical. For the partial shuffle topology, each (J–1)-stage sub-network is a delta network with identical characteristics and so in the recursive step we only need consider sub-network 0. Figure 5.3 shows the structure of the network being considered when there are $s \geq 2$ stages.

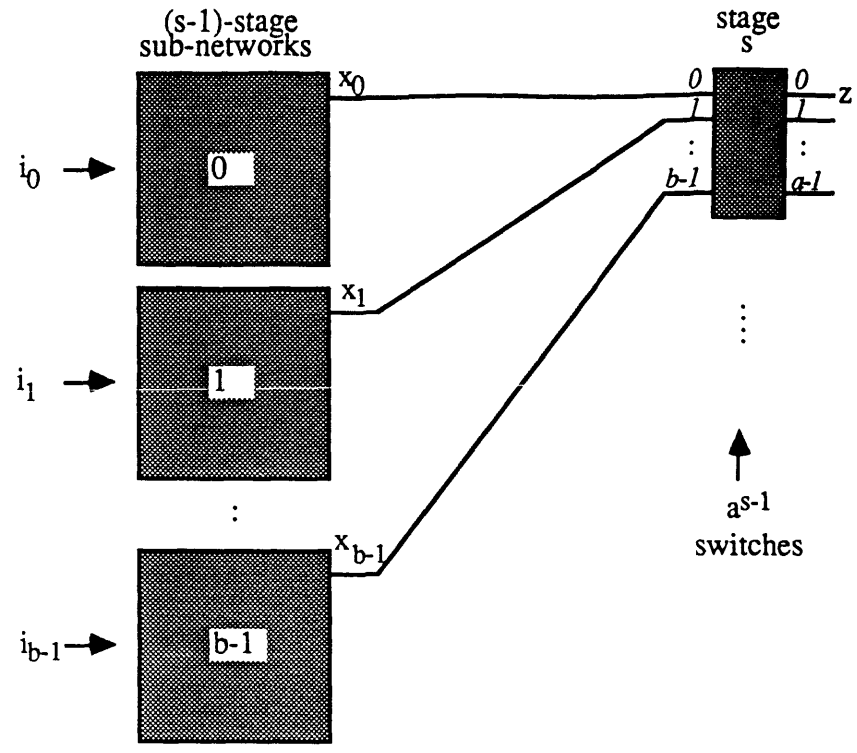


Figure 5.3 Recursive analysis of s-stage delta network

Consider the rightmost, topmost switch in an s-stage delta network. This takes its inputs from b separate $(s-1)$-stage sub-networks (Figure 5.3). When we condition on the number of active inputs to each sub-network (their sum being $\#(\underline{N}(s))=n$), the sub-networks are stochastically independent (by assumption). Hence, we can determine the joint probability distribution of the states of the inputs to the rightmost, topmost crossbar by using the result obtained for the next smaller networks in the recursion and simple conditional probability arguments. The distribution of the state of the top output pin then follows by considering a single crossbar and the conditional throughput results of Chapter 3. This gives the recurrence formula in the following theorem.

THEOREM 5.3

Under the assumptions of Section 5.3.1, for an s-stage delta network constructed from $b\times a$ crossbars with n active inputs, the probability that the top output port is in state $z\in\{0,1\}$ is given by the following recurrence.

For $s>1$, $n = 0,1,\ldots,b^s$

$$T_s(z \mid n) = \sum_{\substack{\underline{i} \\ \sum_{k=0}^{b-1} i_k = n}} Q_s(\underline{i} \mid n) \sum_{\underline{x}} T_1(z \mid \sum_{k=0}^{b-1} x_k) \prod_{k=0}^{b-1} T_{s-1}(x_k \mid i_k)$$

and for $n = 0,1, \ldots,b$

$$T_1(0 \mid n) = \frac{a-1}{a+n-1} \quad \text{and} \quad T_1(1 \mid n) = \frac{n}{a+n-1}$$

PROOF

For the base case, we note that the throughput of a $b \times a$ crossbar is given by (3.14)

$$\mu_n = \frac{an}{a+n-1}\mu$$

Since all output pins are equally utilised, the throughput of each pin is μ_n/a and so the probability that the top pin is active is given by $\frac{n}{a+n-1}$.

For an s-stage delta network ($s>1$) we define the following random variables (see Figure 5.3):–

$Z \in \{0,1\}$ the state of output pin 0 of an s-stage network

$N \in \{1,2,\ldots,b^s\}$ the number of active inputs to the s-stage network

$I_k \in \{0,1,2,\ldots,b^{s-1}\}$ the number of active inputs to sub-network k

$X_k \in \{0,1\}$ the state of output pin 0 of sub-network k

$$\begin{aligned} \Pr(Z,N) &= \sum_{\underline{I},\underline{X}} \Pr(Z,N,\underline{I},\underline{X}) \\ &= \sum_{\underline{I}\underline{X}} \Pr(Z \mid N,\underline{I},\underline{X}) \Pr(N,\underline{I},\underline{X}) \\ &= \sum_{\underline{I}\underline{X}} \Pr(Z \mid N,\underline{I},\underline{X}) \Pr(\underline{X} \mid N,\underline{I}) \Pr(N,\underline{I}) \end{aligned}$$

Given $\underline{X}$, Z is independent of $\underline{I}$ and N. In fact, Z only depends on the number (and not distribution) of active inputs to the topmost, rightmost crossbar, so

$$\Pr(Z \mid N,\underline{I},\underline{X}) = \Pr(Z \mid \underline{X}) = \Pr(Z \mid X_0+X_1+\ldots+X_{b-1})$$

Given I_k, X_k is independent of N, I_j ($j \neq k$), so

$$\Pr(\underline{X} \mid N,\underline{I}) = \Pr(X_0 \mid I_0) \Pr(X_1 \mid I_1) \ldots \Pr(X_{b-1} \mid I_{b-1})$$

$$\Pr(Z,N) = \sum_{\underline{I}\underline{X}} \Pr(N,\underline{I}) \Pr(Z \mid \sum_{k=0}^{b-1} X_k) \prod_{k=0}^{b-1} \left[\Pr(X_k \mid I_k)\right]$$

Dividing by Pr(N), we have:

$$\Pr(Z \mid N) = \sum_{\underline{I}} \Pr(\underline{I} \mid N) \sum_{\underline{X}} \Pr(Z \mid \sum_{k=0}^{b-1} X_k) \prod_{k=0}^{b-1} \left[\Pr(X_k \mid I_k)\right]$$

The recurrence follows by appropriately identifying the random variables: $Z=Z_0(s)$, $N=\#(\underline{N}(s))$, $I_k=\#_k(\underline{N}(s))$, $X_k=Z_0(s,k)$ for an s-stage delta network with a partial shuffle topology. The recurrence holds for all delta network topologies since they are all equivalent. □

COROLLARY 1 TO THEOREM 5.3

The expected number of active output pins conditional on n active inputs to a J-stage delta network constructed using $b \times a$ crossbars is given by:–

$$E(m \mid n) = a^J \, T_J(1 \mid n)$$

PROOF

Each one of the a^J outputs is equally utilised with utilisation $T_J(1 \mid n)$. □

From the theorem and its corollary, the required conditional throughput function $\mu_n=\mu E(m \mid n)$ can be used to solve the birth and death process defined in Chapter 2.

5.3.3 Saturated Delta Networks

The saturated delta network (Section 5.2.4) can be interpreted as an open network of servers connected to a delta network with a very large load. The throughput results of this network are useful in two ways. First, it gives an approximate upper bound on throughput for asynchronous, circuit-switched networks in which partial paths are held. Thus, for saturated delta networks, this protocol can be directly compared with other protocols. Secondly, the saturated throughput results provide a more efficient way of finding μ_{b^J} for a delta network with b^J inputs.

LEMMA 5.2

For $t \in (0,1]$, $a \geq 1$, $b \geq 1$

$$\sum_{n=1}^{b} \frac{n}{a+n-1} {}^{b}C_n \, t^n \, (1-t)^{b-n} = \sum_{r=1}^{b} \frac{b!}{(b-r)!} \frac{(a-1)!}{(a+r-1)!} \frac{(-1)^{r+1}}{(r-1)!} t^r \sum_{n=1}^{r} \left[(-1)^{n-1} \, {}^{r-1}C_{n-1} \prod_{\substack{j=0 \\ j \neq n-1}}^{r-1} (a+j) \right]$$

PROOF

By expanding the left hand side (lhs) of the lemma

$$\text{lhs} = \sum_{n=1}^{b} \frac{n}{a+n-1} {}^{b}C_n \, t^n \left(\sum_{i=0}^{b-n} (-1)^i \, {}^{b-n}C_i \, t^i \right)$$

Now the coefficient of t^r in the lhs requires the coefficient of t^{r-n} in the binomial expansion.

$$\text{lhs} = \sum_{r=1}^{b} t^r \sum_{n=1}^{r} \frac{n}{a+n-1} {}^{b}C_n \, (-1)^{r-n} \, {}^{b-n}C_{r-n}$$

$$\text{lhs} = \sum_{r=1}^{b} \frac{b!}{(b-r)!} (-1)^{r+1} t^r \sum_{n=1}^{r} \frac{(-1)^{n+1}}{(a+n-1)(n-1)!(r-n)!}$$

$$\text{lhs} = \sum_{r=1}^{b} \frac{b!}{(b-r)!} \frac{(a-1)!}{(a+r-1)!} \frac{(-1)^{r+1}}{(r-1)!} t^r \sum_{n=1}^{r} \left[\frac{(-1)^{n-1}}{a+n-1} \frac{(r-1)!}{(n-1)!(r-n)!} \prod_{j=0}^{r-1} (a+j) \right] = \text{rhs}$$

□

LEMMA 5.3

Let $F(a,r) = \sum_{k=0}^{r} (-1)^k \, {}^{r}C_k \prod_{\substack{j=0 \\ j \neq k}}^{r} (a+j)$

then for $a \geq 1$ and $r \geq 0$ the function $F(a,r) = r!$

PROOF

The proof is by induction on r.

The lemma holds trivially for base case $r=0$ since $F(a,0)=1$.

For the inductive case consider the difference $\Delta = F(a,r+1) - F(a,r)$

By substituting for the function F and isolating the last term of the first summation we have:

$$\Delta = (-1)^{r+1}\left[\prod_{j=0}^{r}(a+j)\right] + \sum_{k=0}^{r}\frac{(-1)^k}{a+k}\left[\prod_{j=0}^{r}(a+j)\right]\left[(a+r+1)\,{}^{r+1}C_k - {}^{r}C_k\right]$$

For $k>0$ and $r\geq 0$, ${}^{r+1}C_k = {}^{r}C_{k-1} + {}^{r}C_k$, so by taking out the first term of the summation we have:

$$\Delta=(-1)^{r+1}\left[\prod_{j=0}^{r}(a+j)\right]+(a+r)\left[\prod_{j=1}^{r}(a+j)\right]+\sum_{k=1}^{r}\frac{(-1)^k}{a+k}\left[\prod_{j=0}^{r}(a+j)\right]\left[(a+r+1)\,{}^{r}C_{k-1}+(a+r)\,{}^{r}C_k\right]$$

$$\Delta = (-1)^{r+1}\left[\prod_{j=0}^{r}(a+j)\right] + (a+r)F(a,r) + \sum_{k=1}^{r}\frac{(-1)^k\,{}^{r}C_{k-1}}{a+k}\left[\prod_{j=0}^{r+1}(a+j)\right]$$

$$\Delta = (a+r)F(a,r) + \sum_{k=1}^{r+1}\frac{(-1)^k\,{}^{r}C_{k-1}}{a+k}\left[\prod_{j=0}^{r+1}(a+j)\right]$$

$$\Delta = (a+r)F(a,r) - \sum_{k=0}^{r}\frac{(-1)^k\,{}^{r}C_k}{a+k+1}\,a\left[\prod_{j=0}^{r}(a+j+1)\right]$$

$\Delta = (a+r)\,F(a,r) - a\,F(a+1,r)$

Now assuming $F(a,r) = r!$ for $a\geq 1$, then

$$\begin{aligned} F(a,r+1) &= F(a,r) + \Delta \\ &= r! + (a+r)\,r! - a\,r! \quad \text{by the inductive hypothesis} \\ &= (r+1)! \ \square \end{aligned}$$

THEOREM 5.4

Under the assumptions of Section 5.3.1, for a saturated delta network constructed from $b\times a$ crossbars, the probability t_s that an output pin at stage s is active is given by the following:

$$t_1 = \frac{b}{a+b-1}$$

For $s>0$,

$$t_s = \sum_{r=1}^{b}(-1)^{r+1}\,t_{s-1}{}^{r}\prod_{i=0}^{r-1}\frac{b-i}{a+i}$$

Thus for a J-stage network, the throughput $T(\infty) = a^J\,t_J\,\mu$ where μ is the transfer rate for servers connected to the network.

PROOF

Since all input pins to the network are active the recurrence of Theorem 5.3 simplifies considerably in the absence of splitting probabilities $Q_s(\underline{i} \mid n)$ since for all k, $i_k = b^{s-1}$ which implies $Q_s(\underline{i} \mid b^s)=1$.

Consider an s-stage delta network with all b^s inputs active. The $(s-1)$-stage sub-networks are also saturated (see Figure 5.3) and so the recurrence in Theorem 5.2 simplifies to give the following:–

$$T_s(1 \mid b^s) = \sum_{\underline{x}} T_1(1 \mid \sum_{k=0}^{b-1} x_k) \prod_{k=0}^{b-1} T_{s-1}(x_k \mid b^{s-1})$$

Suppose $\#(\underline{x})=n$, i.e. n active inputs to the rightmost, topmost switch. By symmetry, all ${}^bC_n$ arrangements of n active inputs are equally likely and each of these arrangements corresponds to a unique binary vector $\underline{x}$. So, instead of summing over $\underline{x}$, we can sum over n and introduce a factor ${}^bC_n$.

$$T_s(1 \mid b^s) = \sum_{n=0}^{b} {}^bC_n \, T_1(1 \mid n) \, {T_{s-1}}^n(1 \mid b^{s-1}) \, {T_{s-1}}^{b-n}(0 \mid b^{s-1})$$

Let $t_s = T_s(1 \mid b^s)$, so $t_1 = \frac{b}{a+b-1}$ and for $s>1$,

$$t_s = \sum_{n=1}^{b} {}^bC_n \frac{n}{a+n-1} {t_{s-1}}^n (1-t_{s-1})^{b-n}$$

By Lemma 5.2,

$$t_s = \sum_{r=1}^{b} \frac{b! \quad (a-1)! \; (-1)^{r+1}}{(b-r)!(a+r-1)! \; (r-1)!} {t_{s-1}}^r \sum_{n=1}^{r} \left[(-1)^{n-1} \, {}^{r-1}C_{n-1} \prod_{\substack{j=0 \\ j \neq n-1}}^{r-1} (a+j) \right]$$

By Lemma 5.3,

$$t_s = \sum_{r=1}^{b} \frac{b!(a-1)!(-1)^{r+1}}{(b-r)!(a+r-1)!} {t_{s-1}}^r$$ from which the result follows. □

In the following chapter, we derive similar recurrence formulae for the case when the output pins are non-uniformly selected, which allows us to investigate the effect of hot-spots on performance.

Chapter 6

Hot-spot Contention in Delta Networks

Outline

6.1 Terminology

6.2 Reachability

6.3 Non-uniform Traffic Model

- 6.3.1 Network Routing Probabilities
- 6.3.2 Switch Routing Probabilities
- 6.3.3 Relation between Network and Switch Routing Probabilities
- 6.3.4 Induced Routing Probabilities

6.4 Path Blocking Probabilities

6.5 Exact Analysis of Biased Delta Networks

- 6.5.1 Definitions
- 6.5.2 Biased Crossbars
- 6.5.3 Biased Delta Networks

6.6 Analysis of Biased Delta-2 Networks

- 6.6.1 Simplifying Assumptions
- 6.6.2 Biased 2 by 2 Crossbars
- 6.6.3 Recursive Analysis of Biased Delta-2 Networks

6.7 Approximate Analysis of Biased Delta Networks

- 6.7.1 Definitions and Assumptions
- 6.7.2 Approximate Recursive Analysis

6.8 Fixed-point Equation for Release Time Ratios

- 6.8.1 Fixed-point Equation
- 6.8.2 Iterative Method

In the previous chapter we considered a delta network in which both the input ports and output ports were uniformly utilised. However, in reality the use of memory modules connected to the outputs may not be uniform. For example, if there is a shared lock or a frequently accessed part of a data structure residing in a particular memory module, then much of the traffic from the processing elements will be routed to this "hot" memory. We can characterise this memory reference pattern in our traffic model by assigning a routing probability to each memory module for all requests.

More generally, each processing element may have its own memory reference pattern, i.e. a favoured set of memory modules that it frequently accesses. However, if work is assigned randomly to processors and each processor has a large variety of work to do at any time, then we would expect each processor to have an identical memory reference pattern. The simplest non-uniform reference pattern is one in which there is a single hot-spot at a given memory module and where the remaining memories are uniformly utilised. In our analysis, we investigate the biased delta network which also includes multiple hot-spots and collections of hot memory modules.

The overall purpose of this chapter is to determine the conditional throughput function (μ_n) for a delta network with a biased traffic pattern. This function can then be used in Theorem 2.1 to determine the throughput function for the FES of the delta network. Apart from the traffic model, the assumptions for this analysis are the same as that of the uniform delta network analysis in the previous chapter.

We begin the analysis by defining some terminology. In particular, we group the output pins together depending on their *distance* from the hot-spot and each group of pins form a *class*. We derive reachability results (i.e. which output pins are reachable from within the network) in terms of classes. The main outcome of this is that paths to output pins of the same class have the same steady state probability of being blocked to an arrival, i.e. have the same degree of contention. The reachability properties also simplify the model for an arbitrary crossbar in the network. We recursively extend this model to the biased delta network case and describe iteration schemes to determine unknown variables in the model.

6.1 Terminology

In this section we describe some of the terms used in this chapter and extend the graph terminology first used in Chapter 1. The terms refer to a J-stage delta network constructed from $b \times a$ crossbars with a partial shuffle topology. A sequence of

connected nodes and links between the network input ports and output ports is called a *path*, and any contiguous portion of a path which begins at an input port is called a *partial path*. The *decode tree* of a network *output* port comprises all paths to that pin. Similarly, the *decode tree* of a network *input* port is composed of all paths originating from that port. In our analysis, we will consider the decode tree of the topmost input port, as shown by the shaded nodes in Figure 6.1.

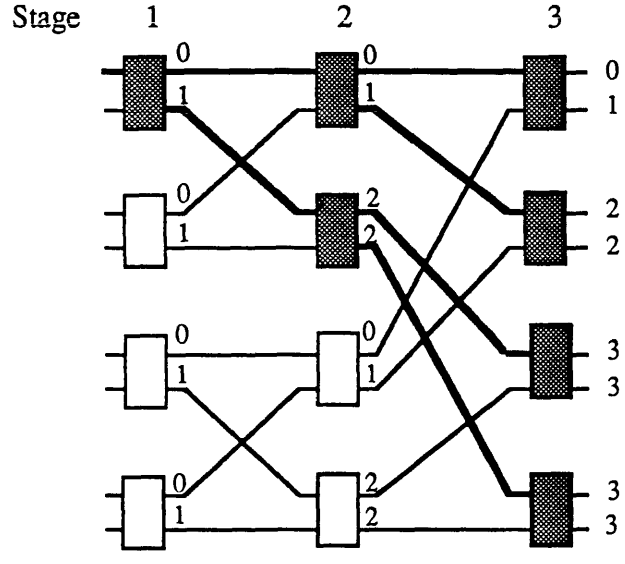


Figure 6.1 Pin Classes in a 8-way Delta-2 Network

As before the pins in any stage of the network are numbered consecutively, starting at zero for the top pin. However, for the decode tree of the top input port, we have an additional numbering scheme (as shown in Figure 6.1). Each output pin in stage s (s=1,2....,J) belongs to a *class* such that class 0 contains the pin numbered 0 and class k (k=1,2,...,s) contains all pins numbered i (where i=a^{k-1},a^{k-1}+1,...,a^k–1). Thus there are a^k–a^{k-1} = (a–1)a^{k-1} class k (k=1,2,...,s) pins in stage s.

This decode tree is obtained by recursively considering sub-network 0 in the definition of the partial shuffle topology. Other sub-networks with the same number of stages have similar characteristics and so have the same pin class labelling as sub-network 0. Thus in stage s, all outputs belong to some class k (k=0,1,...,s).

In the analysis of a single hot-spot, we choose network output port 0 as the hot-spot and refer to the other outputs as *cool* ports or *cool* pins.

6.2 Reachability

In this section we determine the class of network output pins that are reachable from a arbitrary switch in the delta network. An output pin of class j is reachable from a class k output pin in the previous stage if there exists a link from a class k pin to a switch with a class j output pin. From Figure 6.1, we can see that pins of class k+1 are reachable from class $k \geq 0$ pins in the previous stage. Suppose a task traverses a path across the network, then from a non-zero pin the class of the selected output pin increases by one over each stage. Thus starting at a pin of class $k>0$ with t remaining stages, the task will reach class $t+k$ pins. These reachability properties are described more formally in the following lemma and proposition for a J-stage delta network constructed from $b \times a$ crossbars with a partial shuffle topology. These properties have been derived for delta-2 networks in [PaHa88].

LEMMA 6.1

An s-stage delta network constructed from $b \times a$ crossbars with a partial shuffle topology (so that there are b (s–1)-stage sub-networks) has the following inter-stage reachability properties:

- Only the class 0 and class 1 output pins are reachable from the class 0 (and only from class 0) sub-network output pins.
- Only class k+1 (k=1,2,…,s–1) network output pins are reachable from class k (and only from class k) sub-network output pins.

PROOF

The class 0 pins of the (s–1)-stage sub-networks connect to the top switch in stage s, so only class 0 and class 1 pins of the s-stage network are reachable from the class 0 (and only from class 0) pins of the sub-networks.

Consider the $(a-1)a^{k-1}$ class k (k=1,2,…,s) pins labelled i ($i=a^{k-1},a^{k-1}+1,\ldots,a^k-1$) in the ($s$–1)-stage sub-networks. Since switch i in stage s takes its inputs from output pin i in stage s–1 and has outputs labelled $ai,ai+1,\ldots,a(i+1)-1$, only pins numbered $j=a(a^{k-1}),\ldots,a(a^k-1+1)-1 = a^k,\ldots,a^{k+1}-1$ are reachable from (and only from) sub-network pin of class k. In other words, only class k+1 pins in stage s are reachable from class k (k=1,2,…,s–1) (and only from class k) pins in the sub-networks. □

PROPOSITION 6.1

An s-stage sub-network (s=1,2,...,J) of a J-stage delta network constructed from $b \times a$ crossbars with a partial shuffle topology has the following network reachability properties:

- Only network output pins of class k=0,1,...,$J-s$ are reachable from the class 0 (and only from class 0) output pins of the sub-networks.
- Only network output pins of class $J-s+k$ (k=1,2,...,s) are reachable from class k (and only from class k) output pins of the sub-networks.

PROOF

The proof is by induction on the number of remaining stages $t=J-s$ for a task at stage $s=J,J-1,\ldots,1$.

For the base case (t=0), the proposition with $s=J$ is trivially true.

For the case $t>0$, let us assume the proposition holds for $J-s=t$ remaining stages. Now consider an arbitrary ($s-1$)-stage sub-network (i.e. $t+1$ remaining stages). By Lemma 6.1, only stage s output pins of class 0 and 1 can be reached from class 0 (and only from class 0) pins in the ($s-1$)-stage sub-networks. By the inductive hypothesis, the s-stage class 0 pin (i.e. t remaining stages) leads to network output pins of class 0,1,...,t and the s-stage class 1 pin leads to class $t+1$ network output pins. Hence class 0,1,...,$t+1$ network output pins can be reached from class 0 (and only from class 0) pins in the ($s-1$)-stage sub-network.

Again by the Lemma 6.1, only stage s output pins of class $j+1$ (j=1,2,....,$s-1$) can be reached from class j (and only from class j) pins in the ($s-1$)-stage sub-networks. By the inductive hypothesis, the s-stage class $j+1$ pins (i.e. t remaining stages) lead to class $t+(j+1)$ network output pins. Hence class $(t+1)+j$ network output pins can be reached from class j (and only from class j) pins of the ($s-1$)-stage sub-networks.
Thus, the proposition holds for $t+1$ remaining stages and this completes the proof. □

6.3 Non-uniform Traffic Model

In this section we define a number of ways to describe the traffic that uses the delta network. As mentioned at the start of the chapter, we assume that tasks are uniformly distributed over the network inputs but the routing of tasks to the network outputs may

Glossary of Terms

The following is a list of the main abbreviated terms used in this thesis. The page numbers indicate where the term is defined.

bpd	backward path descriptor (Page 21)
DMA	Direct Memory Access (Page 35)
FES	Flow-Equivalent Server (Page 34)
fpd	forward path descriptor (Page 21)
IN	Interconnection Network (Page 9)
LRS	Last Releasing Stage (Page 137)
MHT	Mean Holding Time (Page 79)
MIMD	Multiple Instruction stream, Multiple Data stream (Page 13)
MIN	Multi-stage Interconnection Network (Page 9)
MISD	Multiple Instruction stream, Single Data stream (Page 13)
MM	Memory Module (Page 14)
MRHT	Mean Residual Holding Time (Page 79)
NSD	Network State Descriptor (Page 122)
PE	Processing Element (Page 9)
SIMD	Single Instruction stream, Multiple Data stream (Page 13)
SISD	Single Instruction stream, Single Data stream (Page 13)
VBIS	Virtual Bank of Infinite Servers (Page 43)

Corrigenda

Page 19 — last line should read: "... Chapters 5 and 6."

Page 22 — line 4 should read: "... labels $\alpha_1,\alpha_2,\ldots,\alpha_{j-1},a,\alpha_j,\ldots,\alpha_{J-2}$ ..."

Page 36 — line 7 should read: "higher than switching times ..."

Page 47 — last line should read: "... $\sum_{j=0}^{a-1} m_j=\#(\underline{n})$ ..."

Page 50 — in equations 3.1, the second equation should read:

"$\theta_2(n,m) = m\{(n-m)m^2+(m-1)(2m-1)(a-m+1)\} / n(n-1)a^2$..."

Page 57 — line 8 of the proof should read: "... which represents the request"

Page 111 — line 8 of section 6.7.1 should read:

"the representative pin for all class k (k=1,2,...,s) pins."

Page 115 — lines 3 and 4 of the page and line 6 of section 6.8.2 should read:

"...$C(B(A(\underline{\omega},\underline{r})))$..." instead of "...$A(B(C(\underline{\omega},\underline{r})))$..."

Page 121 — line 18 should read: "where first($k,\underline{a}$)=($a_{J-1},a_{J-2},\ldots,a_{J-k}$) ..."

Page 151 — the equation for p_5 should read: "$2p_5 = \frac{1}{2}p_6 + \frac{1}{8}p_7 + \frac{1}{4}p_8$"

be non-uniform. We characterise the traffic by specifying the routing probability to each class of network output pin. This is called the *biased traffic model* and is a subset of the full non-uniform traffic model which can also be used in the delta network model (after a simple extension) presented in this chapter. The single hot-spot traffic model is a special case of the biased traffic model in which the routing probability to output pins of non-zero class is the same. Of course, the uniform traffic model is a special case of all these non-uniform traffic models. These traffic models for the delta-2 network are also described in [PaHa88].

6.3.1 Network Routing Probabilities

Since all network inputs are treated in the same way, we consider the decode tree of the top network input. For a J-stage delta network constructed from $b \times a$ crossbars we define ρ_k (k=0,1,...,J) as the routing probability to a class k network output pin. Figure 6.2 shows the decode tree of the topmost input of a delta-2 network.

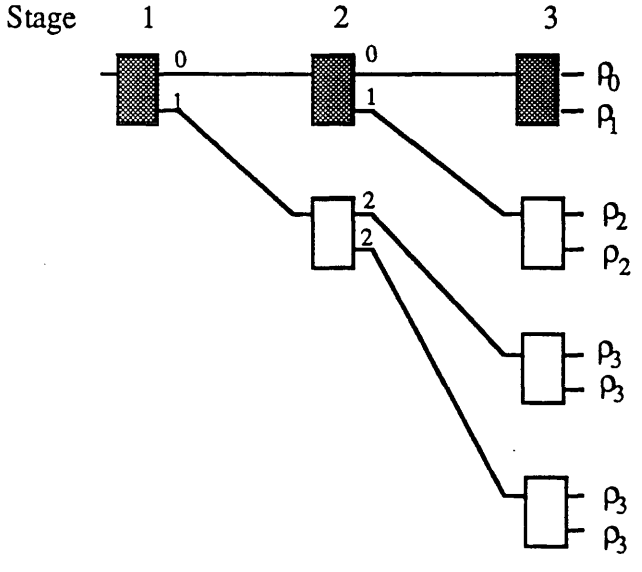


Figure 6.2 Decode Tree of Topmost Network Input

Since all arriving tasks are routed to some network output and there are $(a-1)a^{k-1}$ output pins of class k=1,2,...,J we have the following normalising condition on the routing probability vector $\underline{\rho}$:

$$\rho_0 + (a-1)\sum_{k=1}^{J} a^{k-1}\rho_k = 1$$

For single hot-spot traffic $\rho_k = \frac{1-\rho_0}{a^J-1}$ $\quad k=1,2,\ldots,J$

and for uniform traffic $\rho_k = \frac{1}{a^J}$ $\quad k=1,2,\ldots,J$

6.3.2 Switch Routing Probabilities

An alternative way of specifying the traffic model is to define the switch routing probabilities for an arbitrary switch in the network, i.e. the probability that a task arriving at the switch inputs selects the top switch output. From Figure 6.2, it is easy to see that only the top row of switches (shaded in the diagram) have non-uniform switch routing probabilities. This is also the case for general delta networks because the switch outputs of non-top switches lead to network outputs of the same class (from Proposition 6.1) and so have the same switch routing probability ($1/a$). Thus the biased traffic model can be described by a function $\omega(s)$ which returns the switch routing probability to the top switch output in stage s (s=1,2,...,J). For uniform traffic, the function is a constant $\omega(s)=1/a$.

6.3.3 Relation between Network and Switch Routing Probabilities

Given a network routing probability vector ρ we can determine the switch routing probabilities $\omega(s)$ by considering the reachability of network outputs from the switch outputs. Consider the top switch in stage s and let $t=J-s$ be the number of remaining stages. It has been shown in Proposition 6.1 that from the top switch output (class 0), network output pins of class 0,1,...,t are reachable and from the other switch outputs (class 1) network output pins of class t+1 are reachable. Since there are $(a-1)a^{k-1}$ pins of class k=1,2,...,J we have :

$$\omega(s) = \frac{\text{routing probabilities from top switch output}}{\text{routing probabilities from switch}}$$

$$= \frac{\rho_0 + (a-1)\sum_{k=1}^{t} a^{k-1}\rho_k}{\rho_0 + (a-1)\sum_{k=1}^{t+1} a^{k-1}\rho_k}$$

Conversely, given the switch routing probabilities $\omega(s)$ we can determine the network routing probability vector ρ by considering a task which is establishing a path to a network pin of a certain class and using the forward path descriptor. For a path to the top network output, the task has to choose the top switch output in all stages. Thus,

$$\rho_0 = \prod_{s=1}^{J} \omega(s)$$

For a path to a class k (k=1,2,...,J) network output pin, the task has to select the following:

- top switch output in the first $J-k$ stages (i.e. with probability $\prod_{i=1}^{J-k} \omega(i)$)
- non-top pin in stage $J-k+1$ (i.e. with probability $\frac{1-\omega(J-k+1)}{a-1}$)
- any given switch output in the remaining $k-1$ stages (i.e with probability $\frac{1}{a^{k-1}}$)

Thus, $$\rho_k = \left[\prod_{i=1}^{J-k} \omega(i)\right] \frac{1-\omega(J-k+1)}{(a-1)a^{k-1}} \qquad k=1,2,\ldots,J$$

6.3.4 Induced Routing Probabilities

In a J-stage delta network, let t_k be the utilisation of a network output pin of class k. If the service rate of DMA servers is unity then t_k is also the throughput of the class k pin. Now we can derive an expression for the network routing probabilities $\underline{\rho}$ from the utilisation vector $\underline{t}$ as follows. In the steady state, if a fraction of the total traffic is routed to a particular output pin then the same fraction of the total throughput must flow through that pin. Hence, the routing probabilities can be found from:

$$\rho_k = \frac{t_k}{t_0 + (a-1)\sum_{j=1}^{J} a^{j-1} t_j}$$

Thus these induced routing probabilities provide a useful check on the delta network model since they have to match the network routing probabilities used to determine the pin utilisations. In addition, they are used to derive an iterative solution in Section 6.8.

6.4 Path Blocking Probabilities

For each link in a delta network there is an associated steady state probability that an arriving task wishing to use that link will be blocked: this is called the *link blocking probability*. Each path contains a set of links across the network and has a similarly defined blocking probability. Two paths have the same blocking probability if all links between corresponding stages have the same blocking probability. In general, there is a strong dependence between these link blocking probabilities and so little can be said about them using local information about nodes. However, we can identify links which have the same blocking probability simply by looking at the structure of the network for a given traffic pattern.

In our analysis, we consider a biased traffic pattern, i.e. the task routing probabilities to each class of network output pin can be different. Now suppose that when a task arrives at a particular switching node it selects any switch output pin with equal probability (because the routing probabilities to all possible destination addresses from this node are the same); then the link blocking probability for any switch output link will be identical. By applying this observation to appropriate nodes in the network, we can show that paths to certain output pins have the same blocking probability despite the non-uniform traffic pattern. This is formalised by the following proposition.

PROPOSITION 6.2

For an s-stage delta network constructed from $b \times a$ crossbars with a partial shuffle topology and a biased traffic pattern, the path blocking probability is the same for all paths to pins of the same class.
(I.e. paths to $(a-1)a^{k-1}$ class k (k=1,2,...,s) pins have the same blocking probability.)

PROOF

For the base case (s=1), the network is a $b \times a$ crossbar. Since output pins i=1,2,...,a–1 are treated identically, paths to them have the same blocking probability.

For the multi-stage case (s>1), let us assume that the result holds for an (s–1)-stage sub-network. Since all sub-networks are disjoint, the partial path blocking probabilities to a pin of class k (k=1,2,...,s–1) in any sub-network are the same (by the inductive hypothesis). By Lemma 6.1, we know that pins of class k in the sub-networks connect via switches to output pins of class k+1 in stage s. Routing at these switches is uniform because all reachable pins from the switch outputs belong to the same class. Thus the s-stage switch outputs have the same link blocking probabilities and so path blocking probabilities are the same for all pins within a certain class k+1=2,3,...,s.

Finally, we note that the class 0 pins in the sub-networks connect to s-stage switch 0 in which tasks have uniform routing to class 1 output pins of stage s and so equal path blocking probabilities. Thus the proposition holds for k=1 which completes the proof. □

The proposition shows that for general biased traffic, paths to output pins of the same class experience the same degree of contention. In particular, this holds when the top output pin is a traffic hot-spot and all other pins of the delta network are uniformly utilised (i.e. a single hot-spot traffic pattern). Since class 0 is the hot-spot, we would

expect tasks requesting it to experience different contention from tasks which request other classes of output. Furthermore, the other classes of output each experience different contention even though they are uniformly utilised. The reason for this is summarised by the following argument.

For a single hot-spot traffic pattern, all nodes in the decode tree of the top output pin will be busier than in the uniformly utilised case, so we refer to it as the *hot* decode tree. The degree of overlap with another output pin's decode tree will determine the extent of the extra contention caused by the hot-spot. For example, customers destined for pin number one (adjacent to the hot pin) suffer the greatest contention because pin one's decode tree nodes are identical to the hot-pin's decode tree nodes. However, the customers destined for the bottom output pin suffer less hot-spot contention because its decode tree nodes only overlap the hot decode tree at the leaves (crossbars in stage one). Consequently, in equilibrium, a greater proportion of the customers will build up for pin number one than for the bottom pin. It is easy to see, by an argument analogous to the one used to prove Proposition 6.2, that certain output pins have the same build up of customers and that these pins can be grouped together to form pin classes.

If different classes of network output pins are reachable from a given switch in the network then paths from the top switch output may have different blocking probabilities to paths from the other switch outputs. Consequently, when the top and a non-top switch output become active, the mean time taken for them to become inactive (possibly instantaneously), which we call their *release time*, will be different. Hence modelling the presence of a hot-spot differs from the uniform routing case in two important aspects: the different routing probabilities and the different release times for outputs of a given switch.

6.5 Exact Analysis of Biased Delta Networks

Little's result can be applied to pins in a biased delta network in the steady state (as with the uniform case in Section 5.1). In this section, we begin by extending some of the definitions given for the uniform network and then analyse a biased crossbar in equilibrium. The crossbar results are then used to derive an expression that gives the utilisation of each crossbar in the network. However, three parameters for each crossbar in the network are unknown and so require estimation or iteration. This will be the subject of Section 6.6.

6.5.1 Definitions

The following definitions for a $b \times a$ crossbar are extensions of those given in Section 5.1.1 for the uniform crossbar.

- The utilisation of input pin i of the crossbar is denoted by π_i (i=0,1,...,b–1).
- The release time ratio r is the ratio of the MHT of pin j (for any j=1,2,...,a–1) and the MHT of pin 0.
- The blocking probability B_{ij} (i=0,1,...,b–1; j=0,1,...,a–1) is the steady state probability that, at the instant a task arrives on input pin i, the output pin it requires (pin j) is already held by some other task.

6.5.2 Biased Crossbars

LEMMA 6.2

For a biased $b \times a$ crossbar with output pins 0 and j (j=1,2,...,a–1) that have, respectively:

- selection probability ρ and $\frac{(1-\rho)}{(a-1)}$
- MHT m and rm
- MRHT d_0 and d_1

the probability that output pin 0 is active is given by:

$$p = \rho m \sum_{i=0}^{b-1} \frac{\pi_i}{\rho(m+B_{i0}d_0) + (1-\rho)(rm+B_{i1}d_1)}$$

and the probability that output pin j>0 is active is given by:

$$q = \frac{(1-\rho)r}{(a-1)\rho} p$$

PROOF

Let the arrival rate on input pin i be λ_i (i=0,1,...,b–1).
Applications of Little's result to pins then yield:

(i) For input pin i

$$\pi_i = \lambda_i[\rho(m+B_{i0}d_0) + (1-\rho)(rm+B_{i1}d_1)]$$

(ii) For output pin 0

$$p = \rho(\lambda_0+\lambda_1+\ldots+\lambda_{b-1})m$$

(iii) For any output pin $j>0$

$$q = \frac{(1-\rho)}{a-1}(\lambda_0+\lambda_1+\ldots+\lambda_{b-1})rm$$

The first result follows from (i) and (ii), and the second from (ii) and (iii) □

6.5.3 Biased Delta Networks

For the decode tree of the top network input pin in a J-stage delta network constructed from $b\times a$ crossbars we define the following terms:

- Let π_{sk} (s=1,2,...,J; k=0,1,...,s) be the utilisation of a class k output pin in stage s and π_{00} be the utilisation of the network input pins.

- For non-uniform crossbars in stage s we define the following:
  ω_s is the routing probability to pin 0 of the crossbar
  r_s is the release time ratio
  β_{sj} is the blocking probability for an arrival which selects pin j
  α_{sj} is the ratio MRHT:MHT for output pin j

- For uniform crossbars in stage s we define the following:
  β_s is the blocking probability for an arrival at the crossbar
  α_s is the MRHT:MHT for output pins

THEOREM 6.1

The equilibrium probability π_{sk} (s=0,1,...,J; k=0,1,...,s) that an output pin of class k in stage s of a J-stage network is active satisfies the following recurrence formulae:

$$\pi_{s0} = \frac{b\omega_s\pi_{s-1,0}}{\omega_s(1+\beta_{s0}\alpha_{s0}) + (1-\omega_s)(r_s+\beta_{s1}\alpha_{s1})}$$

$$\pi_{s,1} = \frac{(1-\omega_s)r_s}{(a-1)\omega_s}\pi_{1,0}$$

for k=2,3,...,s:

$$\pi_{s,k} = \frac{b\pi_{s-1,k-1}}{a(1+\beta_s\alpha_s)}$$

PROOF

Using class reachability properties (Lemma 6.1), we know that class 0 and 1 output pins of stage s can be reached from class 0 pins of stage s–1. Since each input to the top stage s crossbar has utilisation $\pi_{s-1,0}$, the crossbar's output pin utilisation can be determined from Lemma 6.2. Hence the first two recurrence must hold. Similarly crossbars in stage s with class k>1 output pins take their inputs from class k–1 pins in stage s–1 and so all inputs have utilisation $\pi_{s-1,k-1}$. Since these crossbars are uniform, the final recurrence follows from Lemma 5.1. □

6.6 Analysis of Biased Delta-2 Networks

In this section we make some approximating assumptions so that blocking probabilities and MRHTs can be determined. Initially we assume that all release time ratios are known, but later we derive a fixed-point equation which they must satisfy (Section 6.8) and describe a iterative scheme by which they can be determined.

We first analyse a biased 2×2 crossbar and then extend it recursively to biased delta-2 networks. This leads to a recurrence formula for the conditional throughput function which can be used in the birth and death process of Chapter 2 to determine FES rate of the delta-2 network in a closed system.

6.6.1 Simplifying Assumptions

As with the uniform case (Section 5.2.1), we make the following assumptions:

(i) For all pins in the network, MRHT=MHT, so $\alpha_{s0} = \alpha_{s1} = \alpha_s = 1$

(ii) The AOP holds for all crossbars, so an arriving task "sees" the other input pin in its equilibrium state. In particular, when an arrival at one input finds the other input active, the probability that the latter input is connected to the upper or lower output is the same as the steady state probability that it is connected to the upper or lower output (respectively) conditional on only one input being active.

Both of these assumptions are discussed in Section 5.2.1.

6.6.2 Biased 2 by 2 Crossbars

COROLLARY TO LEMMA 6.2

Assuming that the non-uniform crossbar has the AOP and that the MRHT = MHT for both output pins,

$$p = \rho(\rho+(1-\rho)r)\left[\frac{\pi_0}{G(\pi_1)} + \frac{\pi_1}{G(\pi_0)}\right] \quad \text{and} \quad q = \frac{(1-\rho)r\,p}{\rho}$$

where for i=0,1: $G(\pi_i) = (1+\pi_i)(\rho^2+(1-\rho)^2r^2) + 2\rho(1-\rho)r$

PROOF

Conditional on only the lower input being active, let $P(\uparrow)$ be the probability that the task in the crossbar is holding the upper output and let $P(\downarrow)$ be the probability that it is holding the lower output.

Applying Little's result to the crossbar at times when only its lower input is active, is equivalent to using Lemma 6.2 with π_0=0 and π_1=1 (so that B_{10}=B_{11}=0). This gives:

$$p' = \frac{\rho}{\rho+(1-\rho)r} \qquad \text{and} \qquad q' = \frac{(1-\rho)r}{\rho+(1-\rho)r}$$

where p' and q' are the utilisations of the upper and lower output pins when only the lower input pin is active. Suppose $\pi(S)$ is the stationary probability that the crossbar is in state S where S (defined in Section 5.1.1) represents the state of each crossbar pin. Thus we have:

$$p' = \frac{\pi(0\ 1\ 1\ 0)}{\pi(0\ 1)} \qquad \text{and} \qquad q' = \frac{\pi(0\ 1\ 0\ 1)}{\pi(0\ 1)}$$

and so $P(\uparrow) = p' = \dfrac{\rho}{\rho+(1-\rho)r}$ and $P(\downarrow) = q' = \dfrac{(1-\rho)r}{\rho+(1-\rho)r}$

Given that the AOP holds and that one input pin is active (by symmetry), a new arrival on the other input sees the active input connected to the upper output with probability $P(\uparrow)$ and to the lower output with probability $P(\downarrow)$. Thus we have the following blocking probabilities:

$$B_{00} = \pi_1 P(\uparrow) \qquad \text{and} \qquad B_{01} = \pi_1 P(\downarrow),$$
$$B_{10} = \pi_0 P(\uparrow) \qquad \text{and} \qquad B_{11} = \pi_0 P(\downarrow)$$

The result follows by substituting these blocking probabilities and d_0=m, d_1=rm in Lemma 6.2. □

Let $U_k(\pi_0,\pi_1,\rho,r)$ be the utilisation of output pin k of a crossbar with input utilisations π_0 and π_1; upper pin routing probability ρ; and release time ratio r. Thus $U_0(\pi_0,\pi_1,\rho,r) = p$ and $U_1(\pi_0,\pi_1,\rho,r) = q$ in the Corollary to Lemma 6.2. The result here agrees with the Corollary to Theorem 4.4 in that:

$$U_k(1,1,\rho,r) = W_k(1|2,\rho,r) \text{ and } U_k(0,1,\rho,r) = W_k(1 \mid 1,\rho,r)$$

6.6.3 Recursive Analysis of Biased Delta-2 Networks

In this sub-section, we derive the throughput of a delta network with non-uniform routing, conditional on the number of active inputs to the network. Although the approach presented here is applicable to full non-uniform routing in which all output pins are selected with different probabilities, for simplicity, we only consider the biased traffic model.

Given $\underline{N}$ and $\underline{Z}$, as defined in Section 5.1, we now define the following probability distributions:-

$$T_s^{(0)}(n) = \Pr(Z_0=1 \mid \#(\underline{N})=n)$$

and for $k>0$,

$$T_s^{(k)}(n) = \Pr(Z_{2^{k-1}}=1 \mid \#(\underline{N})=n)$$

$T_s^{(k)}(n)$ is the probability that an output pin of class k is active in an s-stage network with n active inputs and typically we use pin number 2^{k-1} as the representative pin for all class k pins.

In this recursive analysis, we consider the decode tree of the top input pin of the network. For the base case (s=1) of the recursion, the interconnection is a 2-way crossbar with switch routing probability for the top switch output given by $\omega(1)$. For the s-stage case (s>1), the upper and lower (s-1)-stage sub-networks are separate so that we can determine utilisations of the inputs to the rightmost crossbars by using the result obtained for the smaller network in the recursion. This gives the recurrence formulae in the following theorem.

THEOREM 6.2

Suppose there are nJ active inputs to a J-stage delta network with a partial shuffle topology. Assuming that every crossbar in the network has the AOP and outputs with MRHT = MHT, the utilisation of a class k output pin in stage s is given by the following recurrences.

for $s=2,3,\ldots,J$; $k=0,1$; $n=1,2,\ldots,n_J$,

$$T_s^{(k)}(n) = \sum_{i=\max(0,n-2^{s-1})}^{\min(n,2^{s-1})} Q_s(i|n)\ U_k(T_{s-1}^{(0)}(i),\ T_{s-1}^{(0)}(n-i),\ \omega(s),\ r(s,n_J))$$

for $s=2,3,\ldots,J$; $k=2,3,\ldots,s$; $n=1,2,\ldots,n_J$,

$$T_s^{(k)}(n) = \sum_{i=\max(0,n-2^{s-1})}^{\min(n,2^{s-1})} Q_s(i|n)\ \ U_0(T_{s-1}^{(k-1)}(i),\ T_{s-1}^{(k-1)}(n-i),\ \frac{1}{2},\ 1)$$

and for $k=0,1$ we have:- $T_1^{(k)}(0) = 0$

$$T_1^{(k)}(1) = U_k(0,\ 1,\ \omega(1),\ r(1,n_J))$$

$$T_1^{(k)}(2) = U_k(1,\ 1,\ \omega(1),\ r(1,n_J))$$

where $\omega(s)$ is the switch routing probability

and $r(s,n_J)$ is the release time ratio for the top switch in stage s when there are n_J active network input pins.

PROOF

The proof is analogous to that of Theorem 5.2. In addition it uses Lemma 6.1, i.e. a pin of class k in stage s is reachable from a pin of class k–1 only. □

COROLLARY TO THEOREM 6.2

The expected number of active output pins conditional on n_J active inputs to a J-stage network is given by:-

$$E(m|n_J) = T_J^{(0)}(n_J) + (2^J-1)\ T_J^{(1)}(n_J)$$

From the theorem and its corollary with $n_J = 1,2,..2^J$, the required conditional throughput function $\mu_n = \mu E(m|n)$ can be used to solve the birth and death process defined in Chapter 2. This gives $T(N)$, the throughput of the closed system for a given population N. However, the release time ratios $r(s,n_J)$ are unknown in Theorem 6.2. It now remains to obtain these ratios and this will be the subject of Section 6.8.

The delta-2 network model presented in this section does not generalise easily to networks constructed from large crossbars mainly because, in general, the input

processes at crossbar inputs are not independent. When there are more than two input pins in a crossbar and we condition on an arrival at one of the inputs, the joint probability distribution of the activity of the remaining inputs is required (to determine the mean waiting time in Little's result). This joint distribution cannot be readily determined from the input pin utilisations that we obtain recursively. Even if the joint distribution can be determined, it is still difficult to calculate the mean waiting time of the new arrival because the arrangements of the connections from the remaining inputs to the output pins are required. Hence in the following section we adopt a different approach to modelling general delta networks.

6.7 Approximate Analysis of Biased Delta Networks

In this section we derive an expression for the conditional throughput (μ_n) of a J-stage delta network constructed from $b \times a$ crossbars with the biased traffic pattern described in Section 6.3. As before we consider the decode tree of the top network input since all inputs have identical characteristics. From Chapter 4 we have models of all nodes in the decode tree, so now we use them to build up a model for the entire delta network.

6.7.1 Definitions and Assumptions

We require the same definitions as in Section 5.1, viz. the random variables $\underline{N}(s)$, $\underline{Z}(s)$, $\underline{Z}(s,k)$ and the probability distribution $Q_s(i \mid n)$. However, now we need to extend the output pin state probability distribution to the following:

$$T_s^{(0)}(z \mid n) = \Pr(Z_0(s)=z \mid \#(\underline{N}(s))=n)$$

$$T_s^{(k)}(z \mid n) = \Pr(Z_{a^{k-1}}(s)=z \mid \#(\underline{N}(s))=n) \qquad k=1,2,\ldots,s$$

$T_s^{(k)}(z \mid n)$ is the probability that the state of an output pin of class k is $z \in \{0,1\}$ in an s-stage network with n active inputs. In the definition above, we use pin number a^{k-1} as the representative pin for all class $k=1,2,\ldots,s$ pins.

In the approximate analysis we make the following assumptions:

- All pin holding times are exponentially distributed.
- The input processes to each input pin of a crossbar are independent.

The consequences of these assumptions (especially the latter) are discussed in Section 5.3.1 for the uniform case.

6.7.2 Approximate Recursive Analysis

In this recursive analysis of a J-stage delta network, the base case is a $b\times a$ crossbar with top switch output routing probability $\omega(1)$ and some release time ratio r. For the s-stage case (s=2,3,...,J), the (s–1)-stage sub-networks are disjoint and we assume that they are independent. Hence we can determine the probability distribution of the states of the inputs to the rightmost crossbars by using the result obtained for the smaller network in the recursion and simple conditional probability arguments. The state of the output pins of these crossbars can be determined using the biased crossbar model described in the previous section. This gives the recurrence formulae in the following theorem.

THEOREM 6.3

Suppose there are n_J active inputs to a J-stage delta network constructed from $b\times a$ crossbars with a biased traffic pattern. Under the assumptions of Section 6.7.1, the conditional output state probabilities for a class k output pin of a s-stage sub-network are given by $T_s^{(k)}$:

Base case (s=1, $k\in\{0,1\}$) $\quad T_1^{(k)}(z \mid n) = W_k(z \mid n,\omega(1),r(1,n_J))$

s=2,3,...,J; $k\in\{0,1\}$

$$T_s^{(k)}(z \mid n) = \sum_{\substack{\underline{i} \\ \sum_{k=0}^{b-1} i_k = n}} Q_s(\underline{i} \mid n) \sum_{\underline{x}} W_k(z \mid \sum_{k=0}^{b-1} x_k,\omega(s),r(s,n_J)) \prod_{k=0}^{b-1} T_{s-1}^{(0)}(x_k \mid i_k)$$

For s=2,3,...,J; k=2,3,...,s

$$T_s^{(k)}(z \mid n) = \sum_{\substack{\underline{i} \\ \sum_{k=0}^{b-1} i_k = n}} Q_s(\underline{i} \mid n) \sum_{\underline{x}} W_0(z \mid \sum_{k=0}^{b-1} x_k, \frac{1}{a},1) \prod_{k=0}^{b-1} T_{s-1}^{(k-1)}(x_k \mid i_k)$$

where W_k is defined in Section 4.3.3 and $r(s,n_J)$ is the release time ratio for the top switch in stage s when there are n_J active inputs to the J-stage network.

PROOF

The proof is analogous to that of Theorem 5.3 but requires class reachability properties and a biased crossbar model. From Lemma 6.1, class $k \in \{0,1\}$ pins in stage s can be reached from class 0 pins in the sub-networks (hence the superscripts in the first recurrence). Likewise, class k (k=2,3,...,s) pins in stage s are reachable from class k–1 pins in the sub-networks. □

COROLLARY TO THEOREM 6.3

The expected number of active output pins conditional on n_J active inputs to a J-stage delta network with a biased traffic model is given by:

$$E(m \mid n_J) = T_J^{(0)}(1 \mid n_J) + (a-1)\sum_{k=1}^{J} a^{k-1} T_J^{(k)}(1 \mid n_J)$$

PROOF

All $(a-1)a^{k-1}$ pins within a class k (k=1,2,...,J) have the same utilisation. The result follows by summing the utilisation of each network output pin. □

Thus the conditional throughput function $\mu_n = \mu E(m \mid n_J)$ can be determined from the corollary and its theorem (given μ is the DMA transfer rate). By substituting this into the birth-death process of Chapter 2, we can determine $T(N)$ (the throughput of the closed system of DMA servers connected to a biased delta network).

However, in Theorem 6.3 the release time ratios $r(s,n_J)$ are unknown. In the next section we develop an iterative scheme based on a fixed-point equation to determine these ratios.

6.8 Fixed-point Equation for Release Time Ratios

In this section we describe a fixed-point approach by which the release time ratios can be determined. Although only delta-2 networks are considered (so, in particular, Theorem 6.2 will be invoked) the approach can be generalised easily to networks with larger crossbars. First we define the following notation:

- Suppose there are n_J active inputs to a J-stage delta network, so that the release time ratios $r(s,n_J)$ can be abbreviated to r_s for the topmost crossbar in stage s (s=1,2,...,J), where r_s>0.

- Let t_k be the utilisation of a class k pin (k=0,1,...,J). From Theorem 6.2, we have $t_k = T_J^{(k)}(n_J)$.

- Let ρ ($0 \le \rho < 1$) be the hot-spot routing probability that describes the network traffic and ρ_k be the routing probability to a class k output pin, so that

$$\rho_0 = \rho \text{ and } \rho_k = \frac{1-\rho}{2^J - 1} \quad (k = 1,2,...,J)$$

- Let $\omega(s)$, which we abbreviate to ω_s, be the corresponding top stage s switch routing probability function, where $0 \le \omega_s < 1$ (s=1,2,...,J).

- Let $\underline{r} = (r_1, r_2,...,r_J)$, $\underline{t} = (t_0, t_1,...,t_J)$, $\underline{\rho} = (\rho_0, \rho_1,...,\rho_J)$ and $\underline{\omega} = (\omega_1, \omega_2,..., \omega_J)$

6.8.1 Fixed-point Equation

We now derive a fixed-point equation for the pair $(\underline{\omega},\underline{r})$ using the functions A, B and C defined below. Each function takes a pair of vectors and returns a pair of vectors in which the second component ($\underline{r}$) is the same in both pairs. However, in function A, the $\underline{r}$ influences the value of the first vector in the result.

- Firstly, from Theorem 6.2, the utilisation vector is a function of the switch routing probabilities and the release time ratios. Suppose F is this function and define $(\underline{t},\underline{r}) = A(\underline{\omega},\underline{r}) = (F(\underline{\omega},\underline{r}),\underline{r})$.

- Now $\underline{t}$ induces a set of network routing probabilities since the proportion of the total throughput flowing out of a pin in equilibrium must be equal to its selection probability,

$$\text{i.e.} \qquad \rho_k = \frac{t_k}{t_0 + \sum_{j=1}^{J} 2^{j-1} t_j} = [G(\underline{t})]_k, \text{ say.}$$

Now let $(\underline{\rho},\underline{r}) = B(\underline{t},\underline{r}) = (G(\underline{t}), \underline{r})$.

- The switch routing probability is a function of the network routing probability, (Section 6.3.3):

$$\omega_s = \frac{\rho_0 + \sum_{k=1}^{J-s} 2^{k-1}\rho_k}{\rho_0 + \sum_{k=1}^{J-s+1} 2^{k-1}\rho_k} = [H(\underline{\rho})]_s, \text{ say.}$$

Now let $(\underline{\omega},\underline{r}) = C(\underline{\rho},\underline{r}) = (H(\underline{\rho}),\underline{r})$.

Thus we have the following fixed-point equation: $(\underline{\omega},\underline{r}) = A(B(C(\underline{\omega},\underline{r})))$. Although for all $(\underline{\omega},\underline{r})$, $A(B(C(\underline{\omega},\underline{r}))) = (\underline{\omega}',\underline{r})$ for some $\underline{\omega}'$, the choice of $\underline{r}$ determines $\underline{\omega}'$ and hence the fixed-point pair.

We require fixed-points $(\underline{\omega},\underline{r})$ of the above equation. Typically, $\underline{\omega}$ is given by the traffic pattern and a guess is required for $\underline{r}$. From the analysis, we know that r_J=1 because complete paths suffer no further contention and DMA transfer rates are the same regardless of the output pin. The proof of uniqueness of $\underline{r}$ with respect to the fixed-point equation is not given here but we conjecture that $\underline{r}$ is unique because in the 2-stage network different release time ratios in the first stage crossbar give output pin utilisations that induce different switch routing probabilities. This means that both cannot satisfy the fixed-point equation.

6.8.2 Iterative Method

Let $r_s^{(i)}$ be the approximation for r_s after the i^{th} iteration. As an initial guess, we choose the following: $r_s^{(0)}$=1 for s=1,2,...,J. This guess is exact when n_J=1 and also for uniform traffic. Furthermore, in all approximations to $\underline{r}$ we know that r_J=1.

The switch routing probabilities which are induced when the model is parameterised by release time ratios $\underline{r}^{(i)}$ are given by the following:

$$\underline{\omega}^{(i)} = first(\, A(B(C(\underline{\omega},\underline{r}^{(i)})))\,) \quad \text{where} \quad first((p,q))=p$$

The difference between the required and the observed switch routing probabilities can be quantified as follows:

$$d_s^{(i)} = \frac{\omega_s^{(i)} - \omega_s}{\omega_s}$$

Now if $d_s^{(i)}$ is positive, then the utilisation of the upper pin is higher than expected and so the release time ratio for the top switch in stage s needs to be increased. Thus we use the following to update the value of $\underline{r}$:

$$r_s^{(i+1)} = r_s^{(i)}\,[1 + D d_s^{(i)}]$$

where D (typically between 1 and 4) is a damping factor used to reduce the number of iterations.

The stopping condition is $| d_s^{(i)} | < \varepsilon$, $s=1,2,\ldots,J$ for some pre-defined ε.

Chapter 7

Towards a better Protocol for Switching Networks

Outline

7.1 Circuit Switching Protocols

7.2 The Delta Network Topology

- Definitions
- Adjacent Input Address
- Clashing of Two Paths

7.3 The Combined Release-Hold Protocol

7.4 Choosing the Final Releasing Stage

7.5 Unblocking Time of Blocked Tasks

- 7.5.1 Worst Case Analysis
- 7.5.2 Average Case Analysis
- 7.5.3 Saturated Networks

7.6 Implementation of the Release-Hold Protocol

In the Introduction we noted that there are two main types of data transmission protocols for MINs: circuit switching and packet switching. For both protocols, the control aspect of the operational characteristics is normally described as being either centralised (routing of messages determined by some global controller) or decentralised (messages are self-routing). In this chapter we propose a different circuit switching protocol which has partly centralised control. This novel protocol delivers better MIN performance (albeit at the expense of some extra hardware) and is less susceptible to hot-spot tree saturation.

We begin the chapter by comparing two commonly used circuit switching protocols for MINs, i.e. the holding and releasing of partial paths. After some preliminary definitions about the structure of the delta network, we give an informal description of the new protocol. Some heuristics which are required to determine certain parameters in the protocol are considered using the analytical techniques developed in earlier chapters. Finally, the implementation of the protocol is discussed together with other issues such as fault tolerance and cost-effectiveness.

7.1 Circuit Switching Protocols

In this section we compare two types of circuit switching protocols; those in which partially established paths are released and those in which they are held. We find that an important factor in determining which protocol gives better performance is the delay involved in switching a crossbar from one state to another. Typically, the crossbar inputs are polled on a clock pulse and so a new arrival usually has to wait before it can obtain a free output and a blocked arrival (i.e. blocked on some active output) keeps on polling the output pin until it becomes available. (The clock is necessary to avoid simultaneous access to the same output pin.) Thus we define the *switching delay* as the time elapsed between an arrival at the crossbar input (requiring an inactive output) and the point at which the free output pin is acquired.

First we consider the circuit switching protocol in which partial paths are released. Assuming that messages are self-routing (so that there is decentralised control), this protocol can be implemented using very simple hardware. Since partial paths are released, tasks only hold shared resources (links) when a complete path across is obtained (or during the process of path building or releasing). Thus other tasks cannot become blocked by a path which is already blocked (unless it is also in the process of building/releasing). However, all the work done in establishing a path to the blockage is lost and this can be considerable if switching delays are high. The task then has to

retry after some random time and the chosen retry rate is crucial to the task-based performance measures. If the retry rate is too high, the link required will not have changed state and so the next attempt is likely to fail. Even though this increases network utilisation, the throughput may actually decrease because some other tasks may be blocked by the re-attempt but, in the absence of the retrying task, would otherwise have obtained a complete path. If, on the other hand, the retry rate is too low, a task may be waiting at the inputs even though a complete path may have been available for some time. In this case, the opportunity of a complete path may even be lost to some other task. If the overtaking continues the overall response time of that task will become unacceptably high. Thus provided that optimal retry rates are used and switching delays are negligible, the releasing partial paths protocol for MINs should give improved throughput.

The holding of partial paths in circuit-switched MINs can also be implemented using simple hardware provided that messages are self-routing. For this protocol there is no need to determine optimal retry rates since all blocked tasks hold all their acquired links. The hold protocol is fairer than the release protocol in that overtaking is not possible in any crossbar. This is so because tasks which are blocked in a crossbar by some other task can be ordered according to their time of arrival, so that the first task that was blocked acquires the output link as soon as it becomes available. Even though tasks may be blocked by tasks which are themselves blocked, deadlock is avoided because there is no circularity; a task can only be blocked by a task which is further ahead in the network, so in the final stage there must be some active task (with a complete path) that can unblock at least one of those tasks. The inherent drawback of this protocol is that tasks which are blocked by another blocked task may otherwise be able to obtain a complete path. This reduces throughput and makes the network more prone to hot-spot tree saturation. However, when switching delays are large (compared to data transmission times), it is preferable to hold partial paths.

7.2 The Delta Network Topology

The topology of the delta network is important in designing new protocols. Here we only consider the partial shuffle topology, but since all delta topologies are equivalent, there is no loss in generality. The recursive definition for this topology is useful in determining various properties of the network, such as the stages in which two given paths can clash. First we will require the following definitions.

Definitions

Consider a J-stage delta network constructed from $b \times a$ crossbars with a partial shuffle topology. The input pins are labelled consecutively from the top (starting at zero) in base b and similarly for the output pins (in base a). Figure 7.1 shows the labelling in a delta-2 network with 3 stages.

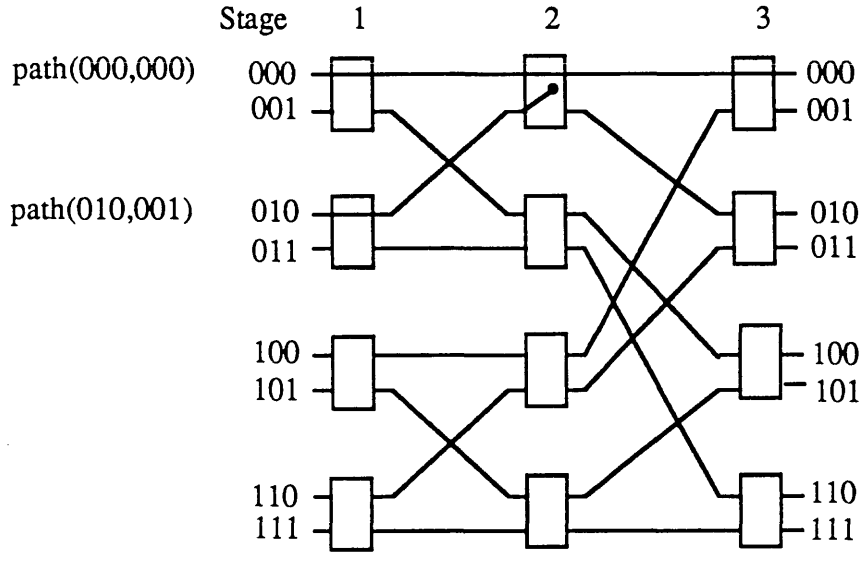


Figure 7.1 Pin Addresses in a Delta Network

We define the following sets:

- $X(c) = \{(x_{J-1},x_{J-2},\ldots,x_0) \mid x_i=0,1,\ldots,c;\ i=0,1,\ldots,J-1\}$.
- $I=X(b)$ is the set of input addresses.
- $U=X(a)$ is the set of output addresses.
- $P=I\times U$ is the set of all possible paths.
- $S=\{0,1,\ldots,J\}$ is the set of link numbers.
- $B=\{$true, false$\}$ represents a boolean data type.

Adjacent Input Address

Consider the following function: Adj : $I\times I\times S\rightarrow B$. Adj($\underline{x}$, $\underline{y}$, s) is true if and only if $s<J$ and paths originating from inputs $\underline{x}$ and $\underline{y}$ can first clash in stage $s+1$, i.e. the two inputs are in adjacent s-stage sub-networks but in the same $(s+1)$-stage sub-network (in

the recursive definition of a (s+1)-stage sub-network). For example consider Figure 7.1 in which the paths from inputs 000 and 010 first clash in stage 2.

Adj can be defined as follows:

$$\text{Adj}(\underline{x}, \underline{y}, s) = [\forall i=J-1,J-2,\ldots,s+1: x_i=y_i] \wedge [x_s \neq y_s]$$

This follows because if two inputs are in the same s+1 stage sub-network, the network backward path descriptor (bpd) to any crossbar (in stage s+1) in that sub-network has to be the same. Hence the first J–s–1 digits of the input address strings have to be the same in the partial shuffle topology. After reaching the (s+1)-stage sub-network, the bpd must select different s-stage sub-networks to reach the $\underline{x}$ and $\underline{y}$ because there is no possibility of a clash before stage s+1. This gives the condition $x_s \neq y_s$.

Clashing of Two Paths

Suppose there are just two tasks in the delta network requiring paths (i_1,d_1) and (i_2,d_2). We require a function which returns the stage in which they clash, if they do interfere, and zero otherwise. We let Clash:$P \times P \rightarrow S$ be this function and define:

$$\begin{aligned}\text{Clash}((i_1,d_1),(i_2,d_2)) &\\ &= s+1 \quad \text{if } \exists s=0,1,\ldots,J-1[\ \text{Adj}(i_1,i_2,s) \wedge (\text{first}(s+1,d_1)=\text{first}(s+1,d_2))]\\ &= 0 \quad \text{otherwise}\end{aligned}$$

where first(k,$\underline{a}$)=(a_{J-1},a_{J-1},…,a_{J-k}), i.e. the first k digits of the address $\underline{a}$

This follows because for the two paths to first clash in stage s+1, the inputs must belong to the same s+1 stage sub-network (i.e. Adj(i_1,i_2,s)) and the first s+1 digits of the destination address have to be the same.

For example, for the two paths in Figure 7.1, Clash returns 2 because 000 and 001 are in adjacent 1-stage sub-networks and the first two digits of the destination addresses are the same.

7.3 The Combined Release-Hold Protocol

In the combined release-hold protocol, paths that are blocked in stages up to and including some stage s* (the *last releasing stage*) are released. Any path blocked further downstream is held. For a J-stage network we have s*=0,1,…,J. When s*=0, the protocol is identical to holding partial paths and when s*=J it is the same as releasing partial paths. The value chosen for s* depends on the switching delays at each crossbar. For high switching delays, s* is low so that all but possibly the shortest

partial paths are held. Conversely, for low switching delays, s^* is high so that all but possibly the longest partial paths are released.

The combined release-hold protocol is a simple combination of the release and hold protocols and so has decentralised control which is easy to implement in crossbars. It has the advantages of both the release protocol (releasing shared resources, etc.) and the hold protocol (time spent establishing a long path is not lost). However, it also has the disadvantages of both, viz. optimal retry rates need to be determined and overtaking is possible. Furthermore, the last releasing stage (s^*) also needs to be determined. Many of the disadvantages no longer apply when the method of releasing paths is enhanced. This is the subject of the remaining part of this section.

All delta network protocols described so far have decentralised control since messages are self-routing. In this section we relax the completely decentralised control of routing by introducing central controllers which control admission of tasks into the network but maintain the self-routing property thereafter. Each central controller is just a gatekeeper which maintains information regarding the current tasks attempting to cross a particular sub-network. Thus when there is a new arrival at a particular sub-network, the sub-network controller will have enough information to determine whether the task will be blocked in the s^*-stage sub-network. In this case the task remains at the input and waits for the conflicting task to complete. When that task completes and releases its path, the controller can re-examine the 'blocked' tasks. The controller uses 'Adj' and 'Clash' to achieve this function.

The enhanced release-hold or Network State Descriptor (NSD) protocol requires extra hardware to maintain the information required by the central controllers, but there are many benefits. Firstly, there is no need to determine retry rates; the task does not attempt to form a path that the controller knows is not available. The retry time for a blocked path is, in some sense, optimal because a free path is guaranteed in the sub-network as soon as it becomes available. Unlike the decentralised release protocol, this protocol is fair in that the blocked tasks can be prioritised (in order of arrival time) by the central controller. In the presence of high switching delays these advantages become particularly important.

The state of tasks in the MIN with the NSD release-hold protocol can be described (informally) by a state diagram (Figure 7.2). The nodes in the graph represent states (of a task) and the arcs represent events. The network is open and arriving tasks start in

the SOURCE node. The SINK node is special in that it represents the absorbing state "transaction terminated".

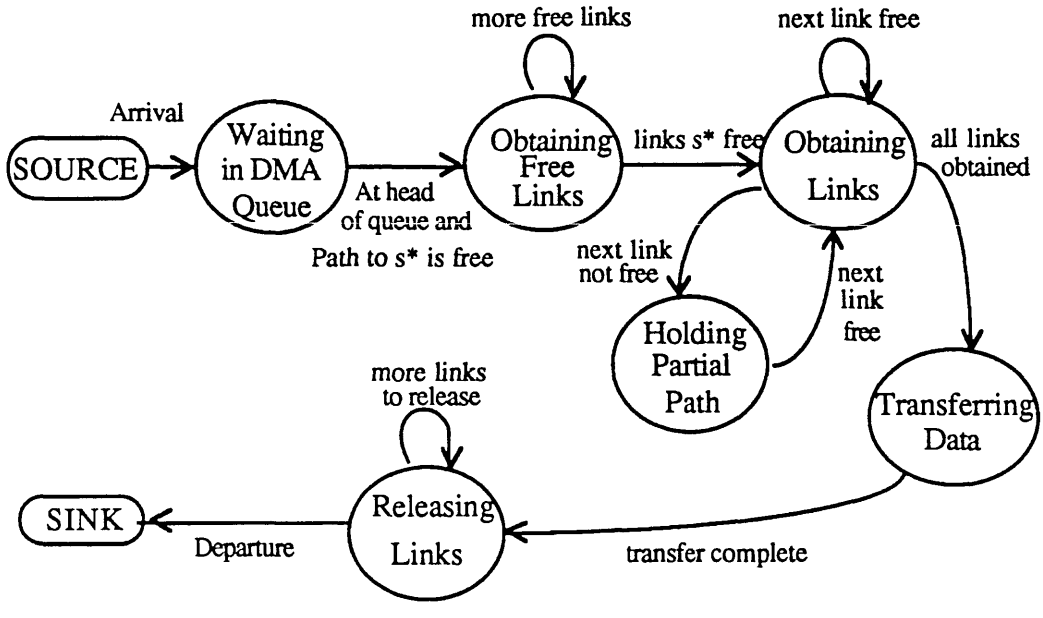


Figure 7.2 Task States in the NSD Release-Hold Protocol

When the switching delays are significant there may be timing problems caused by using current network information to determine some future state. For fixed switching delays, these problems can be readily solved because of the deterministic way in which paths across the network are formed. For example, if a link is currently free but a task has already been sent to acquire it, then the time at which it will actually be acquired can be pre-determined. Thus any new tasks requiring that link have to be blocked immediately by the controller. Similarly, suppose it is known that a task has completed its transfer and is releasing a link in the future. Then a new task requiring the same link may be dispatched if the link will be available by the time the new task gets there.

7.4 Choosing the Final Releasing Stage

The choice of the stage beyond which partial paths are held is important for both the decentralised and enhanced versions of the combined release-hold protocol. A number of factors influence this choice, especially switching delays and time intervals during which links remain active. We assume that tasks incur a fixed delay d in acquiring or releasing an output pin of a crossbar. The mean active time interval for a pin can be approximated by using the delta network models of the previous chapters.

For a J-stage MIN with the NSD protocol and low switching delays, $s^*=J$ gives the best performance, but this requires a central controller which holds information about the entire network. Such a configuration may not be very cost-effective if the extra hardware required to maintain the network information is expensive and a number of smaller central controllers may deliver almost as good performance at a much lower cost.

In the remaining part of this section we will consider a J-stage delta network with the normal combined release-hold protocol. First we require some definitions.

Let $U(s)$, the unblocking time in stage s, denote the mean time elapsed between an arrival to a crossbar which finds the required output link active and the release of that link so that the blocked task can acquire it. Suppose an arriving task in stage s finds the required link active, releases the path acquired and retries after a random time with mean γ. So, on average, the second attempt to acquire the link will occur $2sd+\gamma$ units of time after the first.

Thus it seems reasonable to release the partial path if $U(s)>2sd+\gamma$. As s increases the LHS of the inequality decreases (since there is less contention downstream) and the RHS increases. Hence the last releasing stage can be given by:

$$s^* = \max_s [U(s) > 2sd + \gamma] \qquad s=1,2,\ldots,J$$

The mean retry time can be made stage dependent, i.e. $\gamma(s)$ is the mean retry time for a task which was blocked in stage s. In this case we choose the following retry times:

$$\gamma(s) = U(s) - 2sd \qquad s=1,2,\ldots,J$$

However, $U(s)$ is unknown and so in the next section we describe an approximate method to determine it.

7.5 Unblocking Time of Blocked Tasks

In this section we consider the unblocking time for tasks blocked in stage s of a delta-2 network with the circuit-switched partial paths held protocol and unit data transmission times. An optimistic guess for the unblocking time would be $\forall s=1,2,\ldots,J$: $U(s)=1$, but this assumes that tasks only become blocked by active tasks (i.e. those with a complete path) and this is only true in the final stage.

Let $R(s)$, the release time in stage s, denote the mean time elapsed between an output pin in stage s becoming active and the time at which the task holding the pin releases it. Thus $R(J)=1$, since data transmission times have unit mean.

We make the approximate assumption that arriving tasks which get blocked on an output link of a crossbar, "see" the link as if it has just become active. This is the same as assumption (i) in Section 5.2.1 and implies that $U(s)=R(s)$.

7.5.1 Worst Case Analysis

In order to determine a pessimistic guess for $R(s)$, we require the following definition for maximally blocked tasks:

(i) A task experiences maximal blocking from the final stage if it is blocked in the final stage.

(ii) A task experiences maximal blocking from stage s (s=1,2,...,J–1) if it is blocked in every stage k=s,s+1,...,J–1 by a task which is maximally blocked in stage k+1 and blocked in the stage J.

PROPOSITION 7.1

A task which experiences maximal blocking from stage s (s=1,2,...,J) of a J-stage delta-2 network with the holding protocol has total waiting time equal to 2^{J-s+1} DMA service times (including its own).

PROOF

The proof is by induction on t=0,1,...,J–1 where t is the number of remaining stages after stage s, i.e. t=J–s (Figure 7.3).
For t=0, the task is blocked in stage J and so has to wait for two service times: that of the blocking task and its own service.
For t>0, assume that the proposition holds for t–1 remaining stages and consider the task that is maximally blocked in stage s, i.e. with t remaining stages. By the inductive hypothesis, the blocking task is maximally blocked in stage s+1 and has total waiting time equal to 2^{J-s} service times. When this task releases its path, the blocked task in stage s becomes maximally blocked in stage s+1 and (again by the inductive hypothesis) has to wait another 2^{J-s} service times. Hence the total waiting time for the blocked task is $2^{J-s} + 2^{J-s} = 2^{J-s+1}$ (since transit time between stages is negligible), as required. □

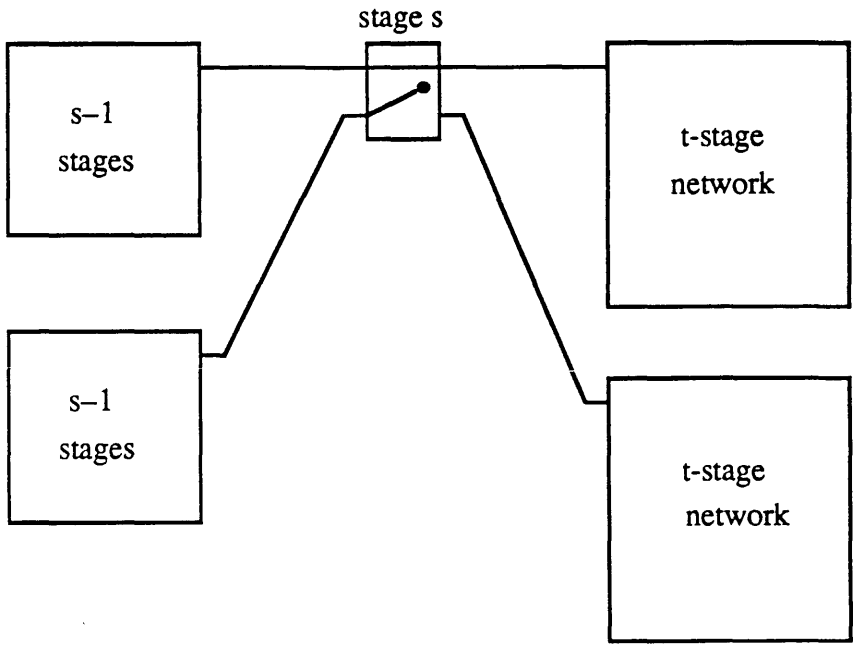


Figure 7.3 Fan-in Fan-out Structure of the Delta Network

A pessimistic guess for the unblocking times is $\forall s=1,2,\ldots,J$: $U(s)=2^{J-s}$ which assumes that the blocking task experiences maximal blocking. Thus the unblocking times for a blocked task in stage s can range from 1 to 2^{J-s}. For the release protocol there is a possibility that the waiting time (of the task waiting for the link to be released) goes beyond this range (because of unfair overtaking) but the probability of this occurring is very low. For the hold protocol we can estimate the release time using the proposition in the following section.

7.5.2 Average Case Analysis

PROPOSITION 7.2

Consider a uniform J-stage delta-2 network with the hold protocol. The mean response time for tasks leaving stage s ($s=0,1,\ldots,J-1$) is given by:

$$R(s) = \frac{\pi_s}{\pi_J} \qquad \text{where } \pi_s \text{ is the utilisation of an output pin in stage } s$$

PROOF

The proposition follows from Little's result, since the tasks leaving stage s enter a t-stage sub-network (Figure 7.3) where $t=J-s$. The utilisation of each of the inputs to

this sub-network is π_s, so the mean number of tasks is $2^t\pi_s$ in equilibrium. The throughput of the sub-network is $2^t\pi_J$. □

From Theorem 5.2 we can determine π_s for s=1,2,...,J–1. For the saturated network we have:

$$\pi_s = T_s(2^s) = \frac{2}{s+2} \qquad \text{by Corollary 2 of Theorem 5.2}$$

Then the proposition becomes $R(s) = \frac{J+2}{s+2}$

Recall the assumption that a task arriving in stage s and becoming blocked "sees" the link as if it has just become active. This would be the case if the length of activity of output pins has the exponential distribution. Under this assumption, the unblocking time is given by $U(s) = \frac{J+2}{s+2}$.

7.5.3 Saturated Networks

Table 7.1 summarises the unblocking times of a task blocked in stage s (s=1,2,...,6) of a 6-stage saturated delta-2 network.

Stage (s)	Best Case Analysis	Average Case Analysis	Worst Case Analysis
1	1	2.67	32
2	1	2.00	16
3	1	1.60	8
4	1	1.33	4
5	1	1.14	2
6	1	1.00	1

Table 7.1 Unblocking Times for Blocked Tasks

This shows that although the range of unblocking times is large, on average they are relatively low. However, in the presence of a hot-spot, the unblocking times can be potentially very high. The saturated network results also suggest that better throughput will be obtained if the retry times are stage dependent.

7.6 Implementation of the Release-Hold Protocol

Special purpose hardware is required to implement the NSD release-hold protocol. However, it can be an add-on performance booster or be integrated with the DMA processors. In either case the sub-network controllers can be of various sizes. The

largest one controls the entire network and gives the best throughput when switching delays are low but also costs the most. The controller's function is quite straightforward, i.e. admit tasks into the network if the path to stage s^* is available, but the way in which network information is stored and updated is more complex, and this causes extra delay.

The controller holds all its information in the network state descriptor (NSD) which maintains the destination addresses of all input addresses. For delta-2 networks this requires s^* bits per input address and larger switch sizes are easy to accommodate in binary if they are some power of two. In order to allow parallel access to all destination addresses, the NSD is held in some form of associative memory. When an event (such as an arrival or departure of a task in the sub-network) occurs, s^* parallel comparisons are required to determine which tasks to block and which to allow into the network. A departure event may cause up to s^* arrivals (i.e. one task unblocked in each stage).

As an example consider a new arrival to the network. Each parallel comparison considers the possibility of blocking in a particular stage s (s=1,2,...,s^*). This can be found from the definitions of the functions Clash and Adj. For there to be a clash between two tasks, the corresponding paths must have destination addresses with a common prefix (i.e first s bits need to be identical) and input addresses with a common prefix (i.e. first J–s bits need to be identical). At the end of the s^* sets of comparisons the arriving task is allowed to enter the network if and only if there were no clashes with the existing paths across the s^* stages.

Simulation models of the NSD release-hold protocol have been developed using the NSD to maintain network information. Numerical results from the simulation are presented in the following chapter.

Chapter 8

Validation and Numerical Results

Outline

8.1 Standard Traffic Schemes

8.2 Validation of the Analytical MIN Models

8.2.1 Validation of Model A against Simulation

8.2.2 Validation of Analytical Model B

8.3 Numerical Results of the Analytical MIN Models

8.3.1 16-way Interconnection Networks

8.3.2 Delta Networks with a Hot-spot

8.4 Simulation Results for the NSD Release-Hold Protocol

8.4.1 NSD Protocol with Standard Traffic Schemes

8.4.2 Varying the Last Releasing Stage

Two different analytical models for delta networks have been presented in previous chapters. We term the first one (in Sections 5.2 and 6.6) *analytical model A* and the second one (in Sections 5.3 and 6.7) *analytical model B*. In deriving model A, we used only mild assumptions whereas for model B, a much coarser independence assumption was required and so some form of validation is necessary. We validate model A with respect to a simulation model and find that both models give very similar results. We then use model A to validate model B and find that model B gives higher throughput. Furthermore, many models that have been presented in the previous chapters have no closed-form solution and so we present some numerical results of these models also. Finally, we present the simulation results of the delta network employing the combined release-hold protocol of the previous chapter and see how it compares with more common protocols. But first we define the types of traffic that use the delta network in all models.

8.1 Standard Traffic Schemes

Four types of traffic models are considered:

- Saturated network with uniform traffic.
- Saturated network with a hot-spot.
- Non-saturated network with uniform traffic.
- Non-saturated network with a hot-spot.

Suppose there is a hot-spot at one of the outputs such that the routing probability (ρ) to it is k times the routing probability to a cool pin. For a J-stage network the cool pin routing probability is $\frac{1-\rho}{2^J-1}$ i.e we require $\rho = \frac{1-\rho}{2^J-1}k$.
Thus we take $\rho = \frac{k}{2^J-1+k}$ as the hot-spot routing probability.

Unless otherwise stated, we will use the following parameters:

- the hot-spot model considered is one in which the hot pin has routing probability twice that of a cool pin, i.e. k=2.
- when the network with J stages is not saturated we take the population N= 2^J, so that there is an average of one task at each input pin.

- the data transmission times are assumed to be exponentially distributed with unit mean.

These traffic models will be used as standards for the comparison of different network protocols in the following sections.

8.2 Validation of the Analytical MIN Models

In this section we consider the results of the models of delta-2 networks presented in Chapters 5 and 6. Model A is validated against a simulation model and model B is validated against model A. All these models represent a closed system of parallel DMA servers (Chapter 2) connected to a circuit-switched delta network in which partial paths are held. The 2-stage delta-2 network is also validated against an exact model in Appendix A. Once a complete path is established, it is held for a random time interval which has a negative exponential distribution with unit mean.

8.2.1 Validation of Model A against Simulation

The throughput results of the validation are tabulated in Tables 8.1, 8.2, 8.3 and 8.4. As can be seen from the first two tables, analytical model A is very accurate (less than 1% error) when the network is saturated. The relative errors are noticeably higher (up to 2.9%) in the case of the non-saturated network because of the additional assumptions required in modelling the closed system by the birth and death process (Chapter 3), i.e. that all arrangements of customers on a given number of active inputs are equally likely.

No. of Stages	Analytical Model A	Simulation Model	95% Confidence Interval	Relative Error (%)
2	2.000	1.992	(1.952,2.032)	0.40
3	3.200	3.185	(3.143,3.228)	0.47
4	5.333	5.375	(5.313,5.437)	–0.78
5	9.143	9.163	(9.101,9.225)	–0.22
6	16.00	15.97	(15.85,16.08)	0.19

Table 8.1 Saturated, Uniform Delta Network Results

No. of Stages	Hot-spot Probability	Analytical Model A	Simulation Model	95% Confidence Interval	Relative Error (%)
2	0.400000	1.896	1.892	(1.866,1.917)	0.21
3	0.222222	3.055	3.057	(3.017,3.097)	–0.07
4	0.117647	5.174	5.193	(5.115,5.271)	–0.37
5	0.060606	8.996	8.989	(8.898,9.079)	0.08
6	0.030769	15.88	15.84	(15.71,15.97)	0.25

Table 8.2 Saturated, Non-uniform Delta Network Results

No. of Stages	Network Population	Analytical Model A	Simulation Model	95% Confidence Interval	Relative Error (%)
2	4	1.612	1.644	(1.603,1.685)	–1.9
3	8	2.548	2.543	(2.498,2.567)	0.20
4	16	4.283	4.227	(4.172,4.283)	1.3
5	32	7.460	7.248	(7.198,7.299)	2.9
6	64	13.28	12.98	(12.89,13.08)	2.3

Table 8.3 Non-saturated, Uniform Delta Network Results

No. of Stages	Network Population	Hot-spot Probability	Analytical Model A	Simulation Model	95% Confidence Interval	Relative Error (%)
2	4	0.400000	1.564	1.579	(1.559,1.598)	–0.95
3	8	0.222222	2.479	2.485	(2.440,2.531)	–0.24
4	16	0.117647	4.206	4.174	(4.104,4.244)	0.77
5	32	0.060606	7.385	7.216	(7.139,7.293)	2.3
6	64	0.030769	13.21	12.88	(12.77,12.99)	2.6

Table 8.4 Non-saturated, Non-uniform Delta Network Results

The confidence intervals were estimated from the simulation results by using the batch means approach. Each sample run modelled 5000 units of time and 5 samples were taken, giving a total simulation time of 25,000 units. For large networks, the whole simulation run took over an hour to complete on a Sun 3 workstation. Notice that in the saturated, uniform network with 2 stages, the simulator's 95% confidence limits do include the exact result of 1.9993 derived in Appendix A, but that the analytical model provides the more accurate estimate.

8.2.2 Validation of Analytical Model B

Table 8.5 compares the throughput results for model B, i.e. the approximate delta network model (Section 5.3) and model A. Delta networks with up to six stages are considered with a variety of different traffic models. Under all types of traffic, the approximate model gives a higher throughput and the relative errors increase with the number of stages. The discrepancy between the two models is caused by the independence assumption used in the approximate model which does not fully capture the contention caused by blocked tasks further downstream and so gives higher throughput results. Sub-networks in large networks have greater contention than the same sized sub-networks in small networks and so the approximate model becomes less accurate as the number of stages increases.

No. of Stages	Network Population	Hot-spot Probability	Analytical Model A	Analytical Model B	Relative Error(%)
2	saturated	uniform	2.000	2.074	3.7
2	saturated	0.400000	1.896	1.954	3.1
2	4	uniform	1.612	1.662	3.1
2	4	0.400000	1.564	1.614	3.2
3	saturated	uniform	3.200	3.431	7.2
3	saturated	0.222222	3.055	3.255	6.5
3	8	uniform	2.548	2.708	6.2
3	8	0.222222	2.479	2.640	6.5
4	saturated	uniform	5.333	5.881	10.3
4	saturated	0.117647	5.174	5.687	9.9
4	16	uniform	4.283	4.673	9.1
4	16	0.117647	4.206	4.598	9.3
5	saturated	uniform	9.143	10.32	12.9
5	saturated	0.060606	8.996	10.15	12.8
5	32	uniform	7.460	8.320	11.5
5	32	0.060606	7.385	8.249	11.7
6	saturated	uniform	16.00	18.42	12.9
6	saturated	0.030769	15.88	18.28	15.1
6	64	uniform	13.28	15.10	13.7
6	64	0.030769	13.21	15.03	13.8

Table 8.5 Throughput Results for Models A and B

8.3 Numerical Results of the Analytical MIN Models

8.3.1 16-way Interconnection Networks

The graph of throughput against the number of customers, N, for a closed system compares four types of 16-way interconnections.

- The full 16-way crossbar with uniform routing (Chapter 3)
- The 2-stage delta-4 network with uniform routing (model B in Chapter 5)
- The 4-stage delta-2 network with uniform routing (model A in Chapter 5)
- The 4-stage delta-2 network with hot-spot routing probability 0.2 (model A in Chapter 6)

As shown in Figure 8.1, the full crossbar gives the highest throughput and approaches its asymptote closely only when N is large (greater than 100). The uniform delta-4 network is best of the other three networks and follows a similar path to the full crossbar graph. The uniform delta-2 network, becomes saturated much faster (when N is about 100) because tasks face more path conflicts on top of memory conflicts than the delta-4 network since it has extra stages. In the presence of a hot-spot this network gives even lower throughput and becomes saturated even faster as the hot pin's decode tree saturates causing more path conflicts. The uniform interconnections only show similar throughput when the population is very small (less than 5). This indicates that even at low loads the blocking caused by path conflicts significantly reduces the throughput of a delta network.

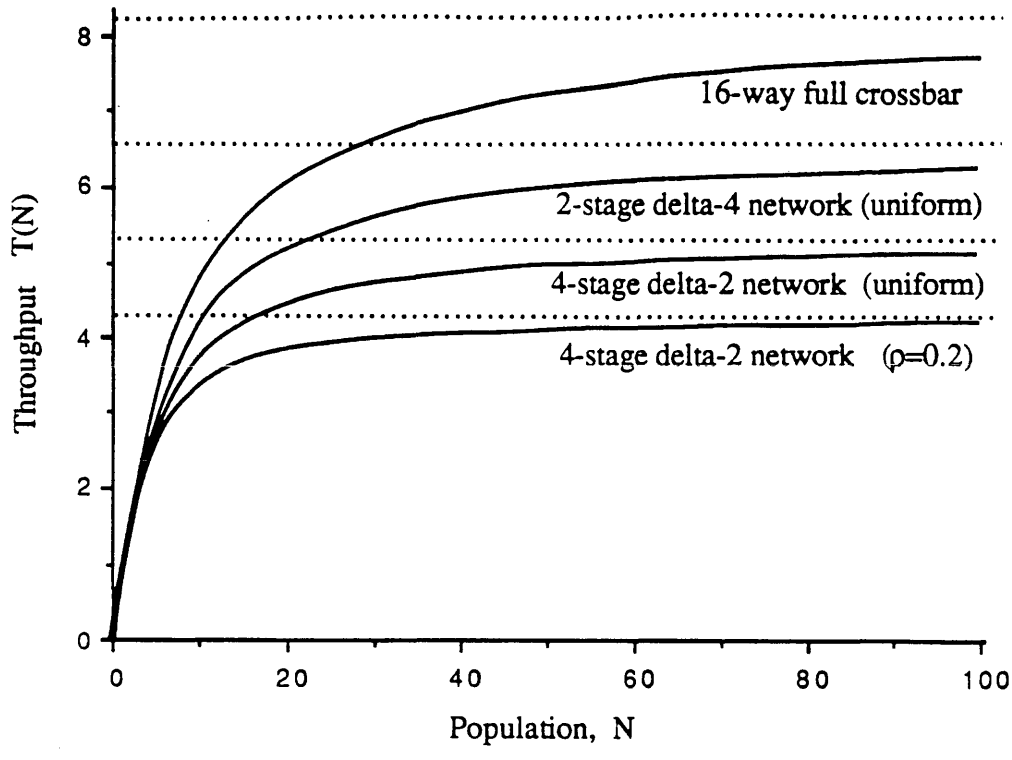


Figure 8.1 A Performance Comparison of 16-way Networks

The graph clearly shows that as the degree of crossbars used in the network increases, the throughput also increases for the same connectivity. This occurs simply because there are fewer stages of contention when larger crossbars are used.

8.3.2 Delta Networks with a Hot-spot

Figure 8.2 shows the effect of hot-spot contention in a 16-way, 4-stage delta-2 network connected to a bank of DMA servers in a closed system with various populations, N. The graph of throughput against the routing probability displays the classic hot-spot effect for populations N=8, 16 and the saturated case in which all input pins are always active. This phenomenon is caused by the additional internal contention placed by the hot-spot on other decode trees. For the saturated case, as ρ increases the throughput reaches its peak when all the output pins are uniformly utilised (ρ=1/16), drops sharply as more of the traffic is routed to the hot-spot, and then follows the curve $1/\rho$ very closely, ultimately giving the throughput of a serial link, when ρ=1. In fact we can easily show that the $1/\rho$ curve provides an upper bound for *all* protocols as follows.

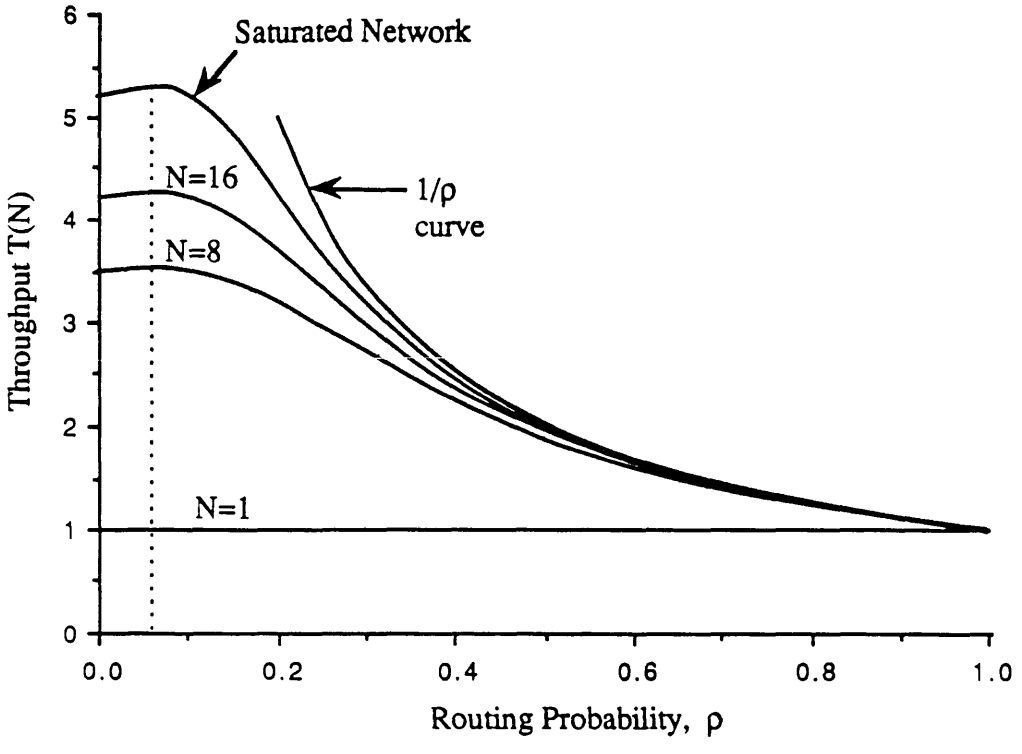


Figure 8.2 Graph of Throughput vs Hot-spot Routing Probability

Suppose the routing probability to network output pin i is ρ_i and the probability that pin i is active (i.e. pin i's throughput) is t_i. Now since ρ_i of the total throughput $T(N)$ is routed to output pin i, we have $t_i = \rho_i T(N)$ which implies $T(N) \leq 1/\rho_i$ for all i, i.e. $T(N) \leq 1/max(\rho_i)$. Hence if ρ is the hot-spot routing probability $T(N) \leq 1/\rho$.

When ρ is large, the throughput curves tend towards the curve $1/\rho$, particularly for large N, because the hot output pin is almost always active. These curves relate to a circuit-switched network in which partial paths are held. When partial paths are released and the server retries immediately, throughput increases and so would lie somewhere between the curve $1/\rho$ and the corresponding curve shown in Figure 8.2; $1/\rho$ is still an upper bound by the same reasoning. In this idealised case, when there is uniform routing (ρ=1/16), the throughput is 16 because all output pins are busy, so the throughput would drop considerably faster if ρ increased slightly, thereby giving a more dramatic hot-spot effect. Similarly, packet-switched networks would give an even more drastic reduction in throughput for a slight increase in traffic to a particular output.

For all of these protocols, in larger networks and with large N, the hot-spot effect is more pronounced because the throughput is greater in the uniformly utilised case and the throughput curve closely follows the $1/\rho$ curve as ρ increases past about 0.2.

However, it may be unlikely that very high hot-spot routing probabilities are encountered in practice, especially when there are a large number of output pins.

Further study is required to determine the sort of non-uniform traffic patterns that are likely to occur in shared-memory multiprocessors and for how long they persist. All multiprocessors which employ a unique path MIN suffer from the hot-spot effect but the addition of enough redundant paths should alleviate the problem as well as improve fault tolerance.

8.4 Simulation Results for the NSD Release-Hold Protocol

The results of the delta network simulator with the NSD release-hold protocol are presented in Tables 8.6 to 8.9 for delta-2 networks with 4-way and 64-way connectivity. This simulator requires two more parameters:

- the switching delay (d)
- the choice of the last releasing stage (LRS)

We present results for the case when there is zero switching delay (d=0, as before) and when there is some significant delay, d>0, as in optical networks. Currently, the switching delay in optical networks is about one microsecond but the data transmission speed is very high: over one gigabits per second. Thus the optical network is only viable in cases where large blocks of data are transferred in one transmission, as in image processing. We will assume that the mean packet length is 10,000 bits so that d=0.1, the ratio of the actual switching delay and the mean transfer time.

8.4.1 NSD Protocol with Standard Traffic Schemes

In this section, the last releasing stage is taken to be zero (as for the all partial paths held case) or the final stage (J) so that the full NSD release method is used.

The results for two sizes of delta network are given (two stages and six stages) with both uniform and hot-spot traffic patterns. The type of hot-spot is different for each size: the 2-stage network has a hot-spot which is twice as likely to be selected as a cool output pin. The same kind of traffic model for the 6-stage network did not significantly affect the throughput results of the uniform model and so a more severe hot-spot (5 times as likely to be chosen as cool pins) is used.

Hot-spot Probability	Switching Delay (d)	LRS	Throughput $T(N)$	95% Confidence Interval
0.250000	0	0	1.976	(1.927, 2.024)
0.250000	0	2	2.094	(2.070, 2.117)
0.250000	0.1	0	1.599	(1.582, 1.617)
0.250000	0.1	2	1.664	(1.644, 1.684)
0.400000	0	0	1.892	(1.852, 1.933)
0.400000	0	2	1.994	(1.955, 2.033)
0.400000	0.1	0	1.526	(1.513, 1.539)
0.400000	0.1	2	1.573	(1.548, 1.598)

Table 8.6 Saturated 2-stage Network Results

Population N	Hot-spot Probability	Switching Delay (d)	LRS	Throughput $T(N)$	95% Confidence Interval
4	0.250000	0	0	1.618	(1.587, 1.649)
4	0.250000	0	2	1.632	(1.604, 1.660)
4	0.250000	0.1	0	1.264	(1.239, 1.290)
4	0.250000	0.1	2	1.268	(1.256, 1.281)
4	0.400000	0	0	1.581	(1.546, 1.616)
4	0.400000	0	2	1.587	(1.552, 1.622)
4	0.400000	0.1	0	1.233	(1.224, 1.242)
4	0.400000	0.1	2	1.239	(1.222, 1.255)

Table 8.7 Non-saturated 2-stage Network Results

Hot-spot Probability	Switching Delay (d)	LRS	Throughput $T(N)$	95% Confidence Interval
0.015625	0	0	15.93	(15.85, 16.01)
0.015625	0	6	22.13	(22.10, 22.16)
0.015625	0.1	0	11.03	(11.00, 11.06)
0.015625	0.1	6	13.20	(13.14, 13.26)
0.073529	0	0	12.54	(12.29, 12.81)
0.073529	0	6	13.65	(13.31, 14.00)
0.073529	0.1	0	9.579	(9.497, 9.662)
0.073529	0.1	6	8.113	(7.919, 8.307)

Table 8.8 Saturated 6-stage Network Results

Population N	Hot-spot Probability	Switching Delay (d)	LRS	Throughput $T(N)$	95% Confidence Interval
64	0.015625	0	0	12.92	(12.80, 13.04)
64	0.015625	0	6	15.60	(15.50, 15.70)
64	0.015625	0.1	0	8.674	(8.652, 8.696)
64	0.015625	0.1	6	9.413	(9.368, 9.458)
64	0.073529	0	0	11.18	(10.99, 11.37)
64	0.073529	0	6	12.90	(12.74, 13.05)
64	0.073529	0.1	0	7.938	(7.830, 8.045)
64	0.073529	0.1	6	7.823	(7.637, 8.009)

Table 8.9 Non-saturated 6-stage Network Results

Note that the throughput values given in the table are normalised by the data transmission times and so they also give the mean number of active output pins. In optical networks, the switching delays lower the normalised throughput, but each of the active outputs have actual throughput of about one gigabits per second whereas in electronic networks the throughput is between 10 to 100 megabits per second for each active output.

We first consider saturated networks with no switching delay which deliver the highest throughput. The results indicate that the full NSD release protocol (LRS=J) gives much better throughput than the partial paths hold protocol (LRS=0). For example, there is a 6% increase in the uniform 2-stage and a 39% increase in the uniform 6-stage network. It is reasonable to assume that the new protocol's benefits increase with the size of the network. However, the performance improvements diminish in the presence of hot-spots with the non-uniform 2-stage network giving a 5% increase and the non-uniform 6-stage network giving only a 9% increase in throughput. Note that the hot-spot is more severe in the 6-stage case than in the 2-stage case, as explained earlier.

Let's now consider the effect of introducing a switching delay at each crossbar in a saturated network. The results for the 2-stage network indicate that the full NSD release protocol does improve throughput over the hold protocol but the improvement is less than in the zero delay case; only a 4% increase for the uniform network. However, it is the 6-stage network which gives the more surprising result because the complex NSD protocol actually gives worse throughput than the simple hold protocol for hot-spot traffic (up to a 15% drop in throughput although a 20% increase in the uniform network). The reason for this is discussed in Section 8.4.2.

When the network population is large there is more possibility of blocking and since the combined release-hold protocol attempts to reduce blocking, its benefits are much reduced in low traffic conditions. For example, the non-saturated 2-stage network with partial paths held gives less than 1% improvement in throughput over the full NSD protocol. For the 6-stage case the improvement is about 21%. These results indicate that large networks do benefit considerably by employing the NSD release-hold protocol even when the population is relatively low (average of one task per input pin).

8.4.2 Varying the Last Releasing Stage

Table 8.10 shows the simulated throughput of a saturated 6-stage delta-2 network with non-uniform routing (ρ=0.073529) and switching delay 0.1 units of time for all choices of the last releasing stage (s^*).

Last Releasing Stage	Throughput	95% Confidence Intervals
0	9.579	(9.497, 9.662)
1	9.543	(9.474, 9.612)
2	9.715	(9.567, 9.863)
3	9.965	(9.778, 10.15)
4	10.18	(10.05, 10.31)
5	9.934	(9.832, 10.04)
6	8.113	(7.919, 8.307)

Table 8.10 Effect of the Last Releasing Stage

First consider the difference in throughput from s^*=0 and s^*=6. This is caused by the considerable switching delay incurred by a task in traversing the network when the path is clear. In the NSD protocol a blocked task has to wait in the input queue and is only allowed to enter the network when a complete path is available. So when the task is unblocked it will experience a 0.6 unit delay (0.1 delay at each of the 6 stages) before it can begin its service. However, in the hold protocol, the delay is much less because the task is unblocked at some stage within the network. With such large switching delays, it appears that the extra blocking caused by holding partial paths is a better option than the extra delays involved in keeping tasks at the inputs.

As we increase the last releasing stage s^*, we see from Table 8.10 that the throughput dips at s^*=1 and then increases, reaching a maximum at s^*=4 and then decreases. The decrease in throughput from s^*=0 and s^*=1 is expected for all traffic schemes with

switching delays because when a task (blocked in the first stage) holds the partial path it does not block any other paths (as the input queues are strictly first-in first-out and link 0 is not shared with any other task). The case $s^*=1$ means that such tasks are not allowed to enter the network and so when they are subsequently released, they incur an extra 0.1 switching delay in crossing the first stage. Thus it is better to choose $s^*=0$ because partial paths of unit length are then held. Of course, when there are zero switching delays, both $s^*=0$ and $s^*=1$ give identical results. In fact, for all these protocols, it is best to hold partial paths of unit length since no other tasks can be blocked by them.

All this substantiates the importance of correctly parameterising the NSD release-hold protocol depending on both the switching delay and the type of traffic. Furthermore, the LRS should be chosen dynamically during network operation and so some set of heuristics are required to make the decision depending on fixed parameters such as switching delays and varying parameters such as traffic conditions. Some of the heuristics described in the previous chapter appear to provide a suitable basis for this purpose.

Chapter 9

Conclusion and Future Work

In recent years there has been a growing interest in MINs of various types and MIN models are required to compare their relative performance. Much of this interest originates from the design of new parallel computer architectures which require some form of switch to interconnect the processing elements. MINs also have other notable applications, such as the switching function in communication networks. Thus the design of the MIN is vitally important especially since it affects the entire system performance. For example the MIN in a shared-memory multiprocessor forms the major bottle-neck when all the processes are busy accessing memory and so it limits the maximum throughput achievable in the entire system. Any model for a MIN with a given set of operational characteristics will be useful in two ways: as a sub-model of the entire system and in isolation as a means of assessing the relative performance of different MINs and protocols. Bearing both these objectives in mind, we now outline the overall conclusions of the thesis and make suggestions for future work in this area.

We have extended the FES aggregation method using Norton's result so that the IN and associated servers in a queueing network model of the parallel system being modelled can be replaced by a composite server. Using a simple birth-and-death process, the problem of finding the FES service rate (as a function of the sub-network population) was reduced to the sub-problem of finding the throughput as a function of the number of active inputs (which we termed the conditional throughput). Thus the use of the composite server allows easy integration of the MIN with the queueing model of the entire system. This simplifies considerably the modelling of newly designed parallel computers which incorporate large MINs.

Full crossbar networks are not feasible to construct in large dimensions using contemporary electronic components (typically 8×8 is the maximum possible) but recent advances in optical crossbars indicate that much larger crossbars may be constructed in the future. Thus analytical models of crossbars will be required in preference to simulation models as these may be too slow to produce accurate results. An exact Markov model has been presented for an asynchronous crossbar with uniform routing. This can be simplified so that the conditional throughput function can be readily obtained. Using the birth-and-death process, a simple expression for the FES service rate was derived for the asynchronous crossbar. In addition, the synchronous crossbar was also analysed and Strecker's result [St70] was extended to include the probability distribution of the number of active outputs (and not just the mean).

In general the traffic across the crossbar is not uniform: certain output pins may have more requests than others. An exact analysis based on a Markov process appears to be

too complex to derive any simple solutions but numerical evaluation from the balance equations is possible. Simplification of the model leads to the recurrence relation of the birth-and-death process encountered earlier and so validates both models to some extent but the boundary conditions are difficult to obtain. Being a model with a two-dimensional state space, reduction to a boundary value problem in complex analysis is an obvious approach to pursue but since the state transition rates for this process are state-dependent, such techniques are difficult to apply. However, an ad hoc transformation method based on the synchronous crossbar with non-uniform routing does allow us to obtain approximate results for the asynchronous crossbar.

The recursive structure of the delta network can be exploited to determine the conditional throughput function required in the birth-and-death process. The base case of the recursive analysis is a single crossbar and applications of Little's result yielded the output pin utilisation in terms of the input pin utilisations. The recursive case also requires the single crossbar results and so the entire network was modelled exactly using the crossbar model. For the delta-2 network, we obtained throughput results based upon some mild approximating assumptions. For delta networks with larger switches, the same mild assumptions are not applicable and a much coarser independence assumption was used which appears to overestimate the throughput. Thus we can see the degradation in performance suffered by a MIN compared with the equivalent full crossbar offering the same connectivity (if it were possible to fabricate such a device).

A hot-spot at a destination pin of a MIN has a widespread effect on all tasks in the network since traffic destined for other pins is hindered by the traffic to the hot-spot (because of the shared internal links). Analysis of the topology of delta networks shows that traffic to certain groups of output pins experience the same contention. The topology also dictates the reachability properties in delta networks, i.e. the groups of network output pins that are reachable from given pins in the network. These properties are useful in determining the relationship between the two ways of specifying non-uniform routing: global (routing to *network* output pins) and local (routing to *crossbar* outputs). An exact analysis of non-uniform delta networks was presented based upon applications of Little's result to pins of the crossbar. Similar assumptions as for the uniform delta network model were required but the analysis is further complicated by an unknown parameter, and a fixed-point equation was derived from which it can be obtained iteratively. Thereby the now classic hot-spot effect was modelled in asynchronous MINs with the circuit switching protocol.

The circuit switching protocol has been used in telephone networks for many years and more recently in switching networks of parallel computers. For this protocol there is a choice of either holding or releasing partially established paths. The new combined release-hold MIN protocol when appropriately parameterised has been shown to give better performance and less hot-spot contention than the original options. However, the new protocol's benefits can only be fully realised when extra hardware is used to control admission of messages into the network. It is hoped that contents addressable memory can be used for this purpose and the recursive structure of the network can be exploited to produce a much improved MIN.

A number of extensions to the delta network have been proposed in the literature, and typically they improve performance and achieve some degree of fault tolerance. The simplest extension is the addition of extra stages which provide multiple paths between a given source and destination. This provision means that some dynamic routing strategy is needed to re-route a blocked task on one path to another path. Modelling a multipath MIN with a given strategy becomes more complex but is close to Kelly's work [Ke86] on telephone networks.

Currently, MINs are constructed using electronic components, but advances in optics do allow the construction of large crossbars using optical technology [JO86]. The optical crossbars can be cascaded to form an optical MIN and this opens up the possibility of INs with extremely large dimensions. However, the switching delays in optical crossbars is relatively large when compared to data transmission times. In this case, the path building process is not instantaneous and so the analytical model has to be extended so that there is a fixed delay at each of the stages. (The simulation model already caters for this switching delay.)

In addition to new MIN designs, new protocols are being designed for existing MINs. The circuit switching protocol considered (with partial paths held) is an important one, being the simplest to fabricate and the one to be used in optical networks. However, there are a number of other protocols that should be considered. The simplest of these is infinitely buffered packet switching since there is no blocking. This protocol has already been studied [MiCi87], but finitely buffered variants are more realistic. The release protocol is a commonly used circuit switching protocol in which partial paths are released and a pilot study based on fixed-point methods can be found in [Co87]. The hybrid protocol is a combination of the circuit and packet switching protocols and is being used in the Flagship machine [Wa87]. Modelling such a protocol analytically is generally quite difficult because the two constituent protocols are normally modelled so

differently. For much the same reasons, the combined release-hold protocol is also hard to model analytically. In both cases simulation models have been used instead.

The hot-spot traffic model that we have developed relates to biased routing to groups of output pins. A relatively simple extension is to consider full non-uniform routing to output pins. In addition, certain destinations may be favoured by certain sources so tasks at different input pins have a different set of routing probabilities to the outputs; a variety of different traffic types use the MIN in a parallel machine. Much of this traffic is for memory requests but there may be some special traffic generated by the operating system, such as broadcast traffic or access to a shared lock. Besides this the traffic may be altered by some load balancing algorithm which may also need to be modelled. However, research should now focus on parallel programs that run on the machines and the extent and type of non-uniformity of traffic that they generate. This would then allow a more integrated approach to modelling parallel software and parallel hardware that constitute parallel processing systems.

Appendices

Appendix A: Exact Analysis of a 2-stage Delta-2 Network

A.1 State Space

A.1.1 Definitions

A.1.2 Symmetric States

A.1.3 Aggregated States

A.2 Markov Process

A.2.1 Balance Equations

A.2.2 Steady State Distribution

A.2.3 Derived Distributions

A.3 Analysis of an Arriving Observer

A.3.1 Notation

A.3.2 State seen by an Arriving Observer

A.4 Summary of Results

Appendix B: Exact Results for a 2 by 2 Crossbar

B.1 Definitions

B.2 Two by Two Crossbar Results

Appendix C: Conversion of Decimals into Rationals

Appendix A: Exact Analysis of a 2-stage Delta-2 Network

An exact analysis of the simplest delta-2 network is possible despite the large number of states involved. Here we analyse a saturated two-stage delta-2 network in which complete paths are held for a random time and all output pins are uniformly distributed. The network uses the circuit switching protocol in which partial paths are held. We make the following assumptions in the analysis:–

- switching times are zero, i.e. paths can be set up and released instantly
- transfer times have negative exponential distribution with unit mean

A.1 State Space

A.1.1 Definitions

We define the state space for a saturated 2-stage delta network by the following set:

$$S = \{ (s_{ij}) \mid i = 1,2;\ s_{1j} = 0,1;\ s_{2j} = 0,1,1_k\ (k=0,1),\ 0 \le j \le 3 \}$$

This forms a 2 by 4 matrix which describes the state of output pin j in stage i. s_{ij}=0 indicates that output pin j in stage i is inactive and s_{ij}=1 indicates that it is active. In stage 2, the state of the active output pin sometimes needs to be extended to indicate precisely which of the stage 1 crossbar pins is connected to it. This is achieved by adding subscripts:

> $s_{2j} = 1_k$ indicates that input pin k (of the crossbar) is holding network output pin j, where k=0 implies that the upper crossbar input (output pin 0 or 1 in stage 1) is connected and k=1 implies that the lower crossbar input (output pin 2 or 3 in stage 1) is connected.

Not all of these states are feasible because, for example, blocking in the first stage limits the number of active outputs in the final stage.

We assume without loss of generality that the servers have rate one and $\pi(\underline{X})$ is the steady state probability for state $\underline{X} \in S$, i.e. the proportion of time in the long term that the system spends in state $\underline{X}$. We make the following abbreviations for the marginal probabilities over an $(n+1)$-dimensional state space (here $n=7$):

$$\pi(x_0 \ldots x_n) = \sum_{\substack{y_i \in \{0,1,1_k\}\ (x_i=*) \\ y_i=x_i \text{ otherwise}}} \pi(y_0 \ldots y_n)$$

For clarity, we will optionally omit any trailing *'s.

A.1.2 Symmetric States

Many of the states in S are equivalent by the symmetry of the delta network. We will consider the states which are equivalent to $\underline{u};\underline{v}$ (as shown in Figure A.1).

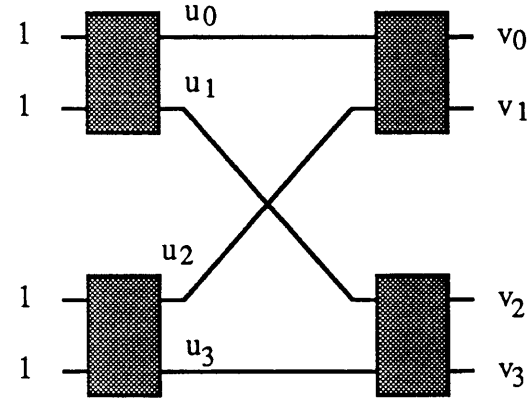


Figure A.1 Symmetric States in a 2-stage Network

By symmetry, we can define the following identities:

(i) $\pi(\underline{u}\,;\,\underline{v}) = \pi(\underline{u}\,;\,v_1\,v_0\,v_2\,v_3) = \pi(\underline{u}\,;\,v_0\,v_1\,v_3\,v_2) = \pi(\underline{u}\,;\,v_1\,v_0\,v_3\,v_2)$
which are obtained by swapping the output pins of crossbars in the final stage: upper crossbar only, lower crossbar only, both crossbars, respectively.

(ii) $\pi(\underline{u}\,;\,\underline{v}) = \pi(u_2\,u_3\,u_0\,u_1\,;\,\underline{v}')$ where $v_i'=v_i$ if $v_i \in \{0,1\}$
$v_i'=1_{1-k}$ if $v_i=1_k$
which is obtained by swapping the stage 1 crossbars and swapping the inputs to each stage 2 crossbar.

(iii) $\pi(\underline{u}\,;\,\underline{v}) = \pi(u_1\ u_0\ u_3\ u_2\ ;\ v_2\ v_3\ v_0\ v_1)$
which is obtained by swapping stage 2 crossbars and swapping the outputs of each crossbar in stage 1.

A.1.3 Aggregated States

Using the above symmetries, we can identify ten symmetry classes in the state space S. We first partition the state space by the number of active outputs:

When there is one active output, we find that only one equivalence class exists:

$$S_0 = \{(1\ 0\ 1\ 0\ ;\ 1\ 0\ 0\ 0),\ldots\}, \qquad |S_0| = 4$$

However, when there are two active outputs, six equivalence classes can be distinguished:

$$S_1 = \{(1\ 0\ 1\ 0\ ;\ 1\ 1\ 0\ 0),\ldots\}, \qquad |S_1| = 2$$

$$S_2 = \{(1\ 0\ 0\ 1\ ;\ 1\ 0\ 0\ 1),\ldots\}, \qquad |S_2| = 8$$

$$S_3 = \{(1\ 0\ 1\ 1\ ;\ 1_0\ 0\ 0\ 1),\ldots\}, \qquad |S_3| = 16$$

$$S_4 = \{(1\ 0\ 1\ 1\ ;\ 1_1\ 0\ 0\ 1),\ldots\}, \qquad |S_4| = 16$$

$$S_5 = \{(1\ 1\ 1\ 1\ ;\ 1_0\ 0\ 1_0\ 0),\ldots\}, \qquad |S_5| = 8$$

$$S_6 = \{(1\ 1\ 1\ 1\ ;\ 1_0\ 0\ 1_1\ 0),\ldots\}, \qquad |S_6| = 8$$

When there are three active outputs, we identify two equivalence classes:

$$S_7 = \{(1\ 0\ 1\ 1\ ;\ 1\ 1\ 0\ 1),\ldots\}, \qquad |S_7| = 8$$

$$S_8 = \{(1\ 1\ 1\ 1\ ;\ 1\ 0\ 1\ 1),\ldots\}, \qquad |S_8| = 4$$

Finally, when there are four active outputs, there is only one equivalence class:

$$S_9 = \{(1\ 1\ 1\ 1\ ;\ 1\ 1\ 1\ 1)\}, \qquad |S_9| = 1$$

Hence there are 75 feasible states in S, which we represent by $S' = \bigcup_{i=0}^{9} S_i$

A.2 Markov Process

Consider the process $X = \{X_t \mid t>0\}$ with state space S' where X_t is the state of the network at time t.

Under the assumption that complete paths are held for a negative exponential time interval with unit mean, the renewal period has a negative exponential distribution with mean 1, 1/2, 1/3, 1/4 depending on the number of active outputs being 1,2,3,4 respectively. Thus X is a non-null, recurrent Markov process since it has finite state space and non-zero transition rates out of any state.

A.2.1 Balance Equations

Let p_i (i=0,1,...,9) be the steady state probability that process X is in state S_i. By symmetry, the steady state probability of each state in S_i can be determined by dividing p_i by $|S_i|$. By considering all the possible state transitions, we can form the following balance equations. For equations with p_i on the LHS, the coefficient of p_j on the RHS is the transition rate from set S_j to set S_i.

$$
\begin{aligned}
p_0 &= \tfrac{1}{4}p_0 + \tfrac{1}{2}p_1 + \tfrac{1}{2}p_3 + \tfrac{1}{2}p_4 \\
2p_1 &= \tfrac{1}{4}p_0 + \tfrac{1}{2}p_1 + \tfrac{1}{2}p_7 \\
2p_2 &= p_2 + \tfrac{1}{2}p_4 + \tfrac{1}{2}p_7 \\
2p_3 &= \tfrac{1}{4}p_0 + \tfrac{1}{2}p_1 + \tfrac{1}{2}p_2 + \tfrac{1}{2}p_3 + \tfrac{1}{4}p_4 + p_5 + \tfrac{1}{4}p_7 + \tfrac{1}{2}p_8 \\
2p_4 &= \tfrac{1}{4}p_3 + \tfrac{1}{2}p_4 + p_6 + \tfrac{1}{4}p_7 + \tfrac{1}{2}p_8 \\
2p_5 &= \tfrac{1}{2}p_6 + \tfrac{1}{8}p_3 + \tfrac{1}{4}p_8 \\
2p_6 &= \tfrac{1}{8}p_3 + \tfrac{1}{2}p_5 + \tfrac{1}{4}p_8 \\
3p_7 &= \tfrac{1}{4}p_0 + \tfrac{1}{2}p_1 + \tfrac{1}{2}p_2 + \tfrac{1}{4}p_3 + \tfrac{1}{4}p_4 + p_7 + \tfrac{1}{2}p_8 + 2p_9 \\
3p_8 &= \tfrac{1}{4}p_3 + \tfrac{1}{2}p_5 + \tfrac{1}{2}p_6 + \tfrac{1}{4}p_7 + \tfrac{3}{4}p_8 + p_9 \\
4p_9 &= \tfrac{1}{8}p_3 + \tfrac{1}{8}p_7 + \tfrac{1}{4}p_8 + p_9
\end{aligned}
$$

A.2.2 Steady State Distribution

The probabilities p_i can be determined by Gaussian elimination after replacing one of the balance equations by the normalising equation. The exact probabilities (expressed in fractions) were determined from the decimal results by using the decimal to fraction

converter described in Appendix C and verified independently. Thereby, the following distribution is obtained:

$$
\begin{aligned}
p_0 &= 2124/8719 \approx 0.2436 \\
p_1 &= 790/8719 \approx 0.0906 \\
p_2 &= 1048/8719 \approx 0.1202 \\
p_3 &= 1608/8719 \approx 0.1844 \\
p_4 &= 788/8719 \approx 0.0904 \\
p_5 &= 195/8719 \approx 0.0224 \\
p_6 &= 210/8719 \approx 0.0241 \\
p_7 &= 1308/8719 \approx 0.1500 \\
p_8 &= 486/8719 \approx 0.0557 \\
p_9 &= 162/8719 \approx 0.0186
\end{aligned}
$$

Let M be the random variable describing the number of active outputs in the network and P(m) be the steady state probability that M=m. Then P(m) has the following distribution:

$$
\begin{aligned}
\mathrm{P}(1) &= p_0 = 2124/8719 \\
\mathrm{P}(2) &= p_1+p_2+p_3+p_4+p_5+p_6 = 4639/8719 \\
\mathrm{P}(3) &= p_7+p_8 = 1794/8719 \\
\mathrm{P}(4) &= p_9 = 162/8719
\end{aligned}
$$

Thus the throughput is given by:

$$E(M) = \sum_{m=1}^{4} m\,\mathrm{P}(m) = \frac{17432}{8719} \approx 1.9993$$

and so the utilisation of each stage 2 output pin $\pi_2 = \frac{1}{4}E(M) = \frac{4358}{8719}$

The approximate analysis gives a throughput of 2 which is very close to the exact result and indicates that the approximations are very mild.

A.2.3 Derived Distributions

Knowledge of the steady state distribution of the delta network enables us to determine the steady state behaviour of each of the crossbars. By symmetry, crossbars in the

same stage have identical overall behaviour, so we only need to consider the topmost crossbars. We start by considering the top crossbar in stage 2.

Crossbar in Stage 2

Let I be the random variable that describes the number of active inputs in the top crossbar of stage 2. In the steady state we have the following distribution for I:

$$\Pr(I=0) = \frac{1}{2}p_0 + \frac{1}{2}p_1 = \frac{1457}{8719} \approx 0.1671$$

$$\Pr(I=1) = p_2 + \frac{1}{2}p_3 + \frac{1}{2}p_4 + \frac{1}{2}p_7 = \frac{2900}{8719} \approx 0.3326$$

$$\Pr(I=2) = \frac{1}{2}p_0 + \frac{1}{2}p_1 + \frac{1}{2}p_3 + \frac{1}{2}p_4 + p_5 + p_6 + \frac{1}{2}p_7 + p_8 + p_9$$
$$= \frac{4362}{8719} \approx 0.5003$$

Let U be the random variable which describes the number of active outputs in the top crossbar of stage 2. Then U and I have the following joint distribution:

$$\Pr(U=1,I=2) = \frac{1}{2}p_0 + \frac{1}{2}p_3 + \frac{1}{2}p_4 + p_5 + p_6 + \frac{1}{2}p_8 = \frac{2908}{8719} \approx 0.3335$$

$$\Pr(U=2,I=2) = \frac{1}{2}p_1 + \frac{1}{2}p_7 + \frac{1}{2}p_8 + p_9 = \frac{1454}{8719} \approx 0.1668$$

Hence $\Pr(U=2 \mid I=2) = \frac{1454}{4362} = \frac{1}{3}$ and $\Pr(U=1 \mid I=2) = \frac{2}{3}$

These conditional probabilities agree with the exact results for crossbars in Appendix B and thereby provide some validation for the calculations in this model.

Crossbar in Stage 1

Now consider the top crossbar in stage 1, which always has two active inputs because the network is saturated. Let V be the random variable which describes the number of active outputs in the top crossbar in stage 1. Then V has the following distribution:

$$\Pr(V=1) = p_0 + p_1 + p_2 + \frac{1}{2}p_3 + \frac{1}{2}p_4 + \frac{1}{2}p_7 = \frac{5814}{8719} \approx 0.6668$$

$$\Pr(V=2) = \frac{2905}{8719} \approx 0.3332$$

As expected the probabilities are not $\frac{2}{3}$ and $\frac{1}{3}$ (but almost) because MRHT≠MHT.

State of Output Pins of the First Stage

For the saturated 2-stage delta-2 network, we can determine the equilibrium probabilities of the state of the output pins of the first stage. This distribution follows from the overall steady state probabilities and the symmetry results (Section A.1.2).

$$p \triangleq \pi(1\,0\,1\,0) = \pi(0\,1\,0\,1) = \tfrac{1}{2}(p_0+p_1) = \frac{1457}{8719}$$

$$p' \triangleq \pi(1\,0\,0\,1) = \pi(0\,1\,1\,0) = \tfrac{1}{2}p_2 = \frac{524}{8719}$$

$$q \triangleq \pi(1\,1\,1\,0) = \pi(1\,1\,0\,1) = \pi(1\,0\,1\,1) = \pi(0\,1\,1\,1) = \tfrac{1}{4}(p_3+p_4+p_7) = \frac{926}{8719}$$

$$r \triangleq \pi(1\,1\,1\,1) = p_5+p_6+p_8+p_9 = \frac{1053}{8719}$$

Thus we have $2p + 2p' + 4q + r = 1$.

A.3 Analysis of an Arriving Observer

In this section, we determine the stationary probability that a crossbar's input pin is active at the instant a task arrives on the other input pin. This can be determined by first finding the proportion of time that the system spends in states, s' say, that *can* provide an arrival to the upper (say) input pin, via a transition into an *input state*, s say. By the Key Renewal Theorem [Ci75], the unnormalised equilibrium probability that the system is in state s immediately after the arrival instant is the expected number of transitions into state s in unit time, which can be determined from the equilibrium probabilities of the states s' and the transition rates from s' to s. This is the approach taken by [SeMi81] which considers the distribution of similar input (and output) states in product-form queueing networks. More details can be found in [HaPa89].

A.3.1 Notation

We will no longer require the *extended* state space S', i.e. the subscript k in 1_k will be omitted when describing active pins. In addition, we define the following:

- Let $A = \{s \in S' \mid s_{10}=1\}$ be the set of states that can exist immediately after an arrival at the top input pin of the second stage, i.e. the set of *arrival* states.
- For $s \in A$, let $B(s) = \{s' \in S' \mid \exists$ a one-step transition $s' \rightarrow s\}$, i.e. the set of *before* states.

Since we are considering a renewal process, the expected number of transitions into state $s \in A$ in unit time is

$$\Phi_s = \sum_{s' \in B(s)} \pi(s')\, p_{s's}$$

where $p_{s's}$ is the probability that a transition from state s' enters state s, conditional on that transition being the result of a service completion at the server which can cause the transition.

A.3.2 State seen by an Arriving Observer

The steady state probability that an arrival at the top input pin of the second stage finds the other input pin of the same (top) crossbar active is therefore $\frac{\Phi^{(1)}}{\Phi}$ where

$$\text{for } k=0,1: \quad \Phi^{(k)} = \sum_{\substack{s \in A \\ s_{12}=k}} \Phi_s \qquad \text{and} \qquad \Phi = \sum_{s \in A} \Phi_s = \Phi^{(0)} + \Phi^{(1)}$$

$\Phi^{(1)}$ is the long term arrival rate on input pin 0 of the upper crossbar in the second stage when the other input pin of that crossbar is active. By symmetry, this is a half of the *total* long term rate of arrivals to the crossbar that result in a state with both input pins active. In the steady state, this is the same as half the departure rate (i.e. throughput) from the crossbar's output pins in states with both its input pins active. Since the crossbar throughput is $\frac{4}{3}$ conditional on both inputs being active (from the Corollary to Lemma B.1), we have

$$\Phi^{(1)} = \tfrac{1}{2}\tfrac{4}{3}\,\pi(1 * 1 *)$$

Similarly, $\Phi^{(0)}$ is the total long term rate of arrivals to the same crossbar that result in only the top input pin being active. In the steady state this is equal to the departure rate from the state with the upper input pin active and the lower one inactive. Thus we have

$$\Phi^{(0)} = \pi(1 * 0 *) = \pi(1 * 0\ 1) \quad \text{for a saturated network}$$

Now, if the AOP held, we would have $\frac{\Phi^{(1)}}{\Phi} = \pi(* * 1 *)$.

Using the distribution of first stage output pins (Section A.2.3):

$$\frac{\Phi^{(1)}}{\Phi} = \frac{2(p+2q+r)}{2p+3p'+7q+2r} = \frac{1454}{2179} = 0.667279 \text{ to 6 decimal places.}$$

and $\pi(* * 1 *) = p + p' + 3q + r = \frac{5812}{8719} = 0.666590$ to 6 decimal places.

Hence the AOP does not hold exactly but does provide a very accurate approximation.

A.4 Summary of Results

Figure A.2 summarises the exact results for the saturated 2-stage network analysed in the previous sections. We use the following terms:

- π_1 and π_2 are the utilisations of outputs from stage 1 and stage 2 respectively
- α_i is the ratio MRHT : MHT of output pins in stage i (i=1,2)
- β_i is the blocking probability for an arrival in stage i (i=1,2)

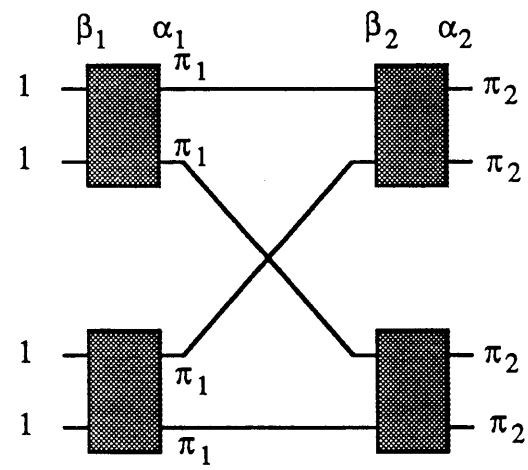


Figure A.2 Pin Utilisations in a 2-stage network

The analysis gives the following exact results for the pin utilisations:

$$\pi_1 = \frac{5812}{8719} \qquad \text{and} \qquad \pi_2 = \frac{4358}{8719}$$

Using Theorem 5.1 for the exact analysis of delta networks (Section 5.1.3), we can use the pin utilisations to derive α_i and β_i. For a 2-stage delta-2 network, Theorem 5.1 becomes: $\pi_1 = \frac{\pi_0}{1+\beta_1\alpha_1}$ and $\pi_2 = \frac{\pi_1}{1+\beta_2\alpha_2}$

Firstly, $\beta_1 = \frac{1}{2}$ and $\pi_0 = 1$ because the network is saturated and $\alpha_2 = 1$ because path holding times have negative exponential distribution. This leaves just two unknowns which have the following solution:

$$\alpha_1 = \frac{2907}{2906} \approx 1.0003 \qquad \text{and} \qquad \beta_2 = \frac{727}{2179} \approx 0.3336$$

Since the assumptions used in Section 5.2 for 2-stage delta-2 networks give $\alpha_1=1$ and $\beta_2=\frac{1}{3}$ it is not surprising that we achieved very accurate results.

Appendix B: Exact Results for a 2 by 2 Crossbar

In this section we derive some exact results for a 2×2 crossbar in the steady state.

B.1 Definitions

As in Appendix A, the equilibrium state probabilities for a 2×2 crossbar are denoted by

$$\pi(i_0\, i_1\, ;\, u_0\, u_1)$$
$$(i_j, u_k \in \{0,1\};\ j,k=0,1)$$

We make the following abbreviations (similar to Appendix A) for the marginal probabilities over an $(n+1)$-dimensional state space (here $n=3$):

$$\pi(x_0 \ldots x_n) = \sum_{\substack{y_i \in \{0,1\}\ (x_i=*) \\ y_i=x_i \text{ otherwise}}} \pi(y_0 \ldots y_n)$$

Also we will optionally omit trailing *'s.

B.2 Two by Two Crossbar Results

LEMMA B.1

The steady state probabilities for a single crossbar with output pins selected with equal probability may be written in terms of marginal probabilities as follows:

(a) $\pi(0\ 1\ 0\ 1) = \pi(0\ 1\ 1\ 0) = \frac{1}{2}\pi(0\ 1)$

(b) $\pi(1\ 0\ 0\ 1) = \pi(1\ 0\ 1\ 0) = \frac{1}{2}\pi(1\ 0)$

(c) $\pi(1\ 1\ 0\ 1) = \pi(1\ 1\ 1\ 0) = \frac{d'}{2m'+d'}\pi(1\ 1)$

(d) $\pi(1\ 1\ 1\ 1) = \frac{2m'-d'}{2m'+d'}\pi(1\ 1)$

where m', d' are respectively the MHT and MRHT of each output pin, conditional on both the crossbar's input pins being active.

PROOF

Cases (a) and (b) follow by symmetry. For the other cases, we apply Little's result in the steady state at times when both inputs are active, i.e. the total queue length is 2. Let the throughput of each output pin be τ (the same for each by hypothesis), and the steady state probability that it is active, conditional on both inputs being active, be p'. Then Applying Little's result to

(i) an output pin gives $p' = \tau m'$

(ii) to the whole switch gives $2 = 2\tau\{m' + \frac{1}{2} d'\}$

Thus $p' = \dfrac{2m'}{2m'+d'}$ and since $1-p' = \dfrac{\pi(1\ 1\ 0\ 1)}{\pi(1\ 1)}$ the result (c) follows.

Result (d) follows from $p' = \dfrac{\pi(1\ 1\ 1\ 0) + \pi(1\ 1\ 1\ 1)}{\pi(1\ 1)}$ □

COROLLARY TO LEMMA B.1

If the MHT = MRHT,

(a) $\pi(1\ 1\ 0\ 1) = \pi(1\ 1\ 1\ 0) = \pi(1\ 1\ 1\ 1) = \frac{1}{3}\pi(1\ 1)$

(b) the throughput of the crossbar is $\frac{4}{3}$ when both its inputs are active.

PROOF

Result (a) follows from $d'=m'$ in Lemma B.1 and (b) follows from

$$\text{conditional throughput} = \frac{1}{\pi(1\ 1)}[\pi(1\ 1\ 0\ 1) + \pi(1\ 1\ 1\ 0) + 2\pi(1\ 1\ 1\ 1)]$$ □

Appendix C: Conversion of Decimals into Rationals

The use of computers to solve numerical problems sometimes leads to inaccurate results because of rounding errors even though the original problem may be solved exactly. For example, many models require numerical solution of a set of simultaneous linear equations and Gaussian elimination is often used to give a fast solution. However, the results are invariably in decimal even though a simple rational result may exist. Here we present a method for converting decimals into rationals which can also be used to compute a sequence of approximating rationals.

Suppose $x_0<1$ is the real which we want to convert to a rational number. If x_0 is rational then it can be determined by the following recurrence

$$x_n = 0 \text{ for some } n$$

$$x_{j-1}=\frac{1}{a_j+x_j} \qquad \text{where } j=1,2,\ldots,n;\ 0<x_j<1;\ a_j\in\{0,1,2,\ldots\}$$

For a given x_0, a_j $(j=1,2,\ldots,n)$ can be readily determined by taking the integer part of the reciprocal:

$$a_j = \lfloor 1/x_{j-1} \rfloor \quad \text{and} \quad x_j = 1/x_{j-1} - a_j \qquad (j=1,2,\ldots,n)$$

PROPOSITION C.1

Given sequence a_j $(j=1,2,\ldots,n)$ generated by the rational x_0, x_0 can be expressed as follows:

$$x_0 = \frac{b_1}{b_0}$$

where $b_n=1$; $b_{n-1}=a_n$

$$b_j = a_{j+1}b_{j+1} + b_{j+2} \quad j=n-2,n-3,\ldots,0$$

PROOF

Since x_j is rational, let $x_j=\frac{t_j}{b_j}$ $(j=0,1,\ldots,n-1)$

$x_n=0$ so let $t_n=0$ and $b_n=1$

Then $x_{j-1}=\frac{1}{a_j+x_j}$ becomes $\frac{t_{j-1}}{b_{j-1}} = \frac{b_j}{a_jb_j+t_j}$ $(j=1,2,\ldots,n)$

Thus $t_{j-1}=b_j$ and $b_{j-1} = a_jb_j+t_j$

Eliminating t_j gives $b_{j-1} = a_jb_j + b_{j+1}$ $(j=1,2,\ldots,n-1)$

$b_j = a_{j+1}b_{j+1} + b_{j+2}$ $(j=0,1,\ldots,n-2)$

Since $x_n=0$, $x_{n-1}=\frac{1}{a_n}$ and so $b_{n-1}=a_n$ as required.

By definition, $x_0 = \frac{t_0}{b_0}$ and $t_0=b_1$, which completes the proof. □

On some occasions, errors in computation of x_0 and the errors incurred in computing the reciprocal make n very large. Hence the sequence should be cut short when x_n is close to zero. In fact a sequence of rationals converging to x_0 can be generated by truncating the sequence $a_1,a_2,\ldots$ appropriately.

The resulting sequence is $a_1, a_2,\ldots,a_{m-2},c_{m-1}$ for some $m>1$ such that

$$c_{m-1} = a_{m-1} \quad \text{if } x_{m-1}<0.5$$
$$= a_{m-1}+1 \quad \text{otherwise}$$

References

[AAS87] G. Adams, D.P.Agrawal, H.J. Siegel, "A Survey and Comparison of Multi-stage Interconnection Networks", *IEEE Computer*, June 1987, pp. 14-27.

[Ak87] I.F. Akyildiz "General Closed Queueing Networks with Blocking", *Proc. of the 12th Annual International Symposium on Computer Performance Modelling*, December 1987, Brussels, Belgium.

[Ak88] I.F. Akyildiz, "Mean Value Analysis for Blocking Queueing Networks", *IEEE Transactions on Software Engineering*, April 1988.

[BaSm76] F. Baskett, A.J. Smith, "Interference in Multiprocessor Computer Systems with Interleaved Memory", *Communications of the ACM*, Vol. 19, No. 6, June 76, pp. 323-334.

[BCMP75] F. Baskett, K. M. Chandy, R.R. Muntz, F.G. Palacios, "Open, Closed and Mixed Networks of Queues with Different Classes of Customers", *Journal of the ACM* Vol. 22, No. 2 (1975)

[Be65] V. Benes, "*Mathematical Theory of Connecting Networks*", Academic Press, New York, 1965.

[Bh75] D. Bhandarkar, "Analysis of Memory Interference in Multiprocessors", *IEEE Transactions on Computers*, September 1975, pp. 897-908.

[Bh84] L.N. Bhuyan, "On the Performance of Loosely Coupled Multiprocessors", *Proc. International Symposium on Computer Architecture*, June 1984.

[Bh85] L.N. Bhuyan, "An Analysis of Processor-Memory Interconnection Networks", *IEEE Transactions on Computers*, March 1985, pp.279-283.

[Bh87] L.N. Bhuyan, "Interconnection Networks for Parallel and Distributed Processing", Guest Editor's Introduction, *IEEE Computer*, June 1987, pp. 9-12.

[Bh88] L.N. Bhuyan, "Performance Evaluation of Multiprocessor Interconnection Networks", tutorial at Sigmetrics 1988 conference, May 1988, Sante Fe, New Mexico.

[BhAg83] L.N. Bhuyan, D.P. Agrawal, "Design and Performance of Generalised Interconnection Networks", *IEEE Transactions on Computers*, December 1983, pp.1081-1090.

[BHJ48] E. Brockmeyer, H.L. Halstrom, A. Jensen, "*The Life and Works of A.K. Erlang*", Academy of Technical Sciences, Copenhagen, 1948.

[Bo72] W.J. Bouknight, et al, "The ILLIAC IV system", *Proceedings IEEE* 60, 4 (April 1972), pp. 369-388.

[CHW75] K. M. Chandy, U. Herzog, L. Woo, "Parametric Analysis of Queueing Networks", *IBM Journal on R&D*, January 1975.

[Ci75] E. Cinlar, *Introduction to Stochastic Processes*, Engelwood Cliffs, N.J.: Prentice-Hall (1975)

[Cl53] C. Clos, "A Study of Nonblocking Switching Networks", *Bell System Technical Journal*, Vol. 32, 1953, pp. 406-424.

[Co77] P. J. Courtois, "*Decomposability: Queueing and Computer System Applications*", Academic Press, 1977.

[Co87] G.A. Cope, "*The Modelling of Circuit-switched Multi-stage Interconnection Networks*", MSc Thesis, Imperial College, September 1987.

[DaRe81] J. Darlington, M. J. Reeve, "ALICE : A Multiprocessor Reduction Machine for the Parallel Evaluation of Applicative Languages", ACM/MIT *Conference on Functional Programming Languages and Computer Architecture*, 1981.

[DiJu81] D.M. Dias, J.R. Jump, "Analysis and Simulation of Buffered Delta Networks", *IEEE Trans on Computers*, Vol. C-30, No. 4, April 1981, pp. 273-282.

[Fe81] T.Y. Feng, "A Survey of Interconnection Networks", *IEEE Computer*, December 1981, pp. 12-27

[FiHa88] A.J. Field, P.G. Harrison, *Functional Programming*, Addison-Wesley 1988.

[Fl66] M.J. Flynn, "Very High-speed Computing Systems", *Proc. IEEE*, December 1966, Vol. 54, pp.1901-1909.

[GaCh87] A. Ganz, I. Chlamtac, "Finite Buffer Queueing Model for P-Persistant CSMA/CD Protocols", *Proc. of the 12th Annual International Symposium on Computer Performance Modelling*, December 1987, Brussels, Belgium.

[GaHu88] U. Garg, Y.P. Huang, "Decomposing Banyan Networks for Performance Analysis", *IEEE Transactions on Computers* C-37, No. 3, 1988, pp.371-376.

[Go83] A. Gottlieb, et al, "The NYU Ultracomputer – Designing an MIMD shared memory parallel computer", *IEEE Transactions on Computers*, C-32, 2 (1983), pp. 173-189.

[GoLi73] L.R. Goke, G.L. Lipovski, "Banyan Networks for Partitioning Multiprocessor Systems", *Proc. of the First Annual Symposium on Computer Architecture*, 1973, pp. 21-28.

[Ha87] P.G. Harrison, "Queueing Models of Large-Scale Parallel Computer Architectures", *ORSA/TIMS Workshop on Queueing Networks and their Applications*, New Jersey, January 1987.

[HaFi86] P.G. Harrison, A.J. Field "Performance Modelling of Parallel Computer Architectures", *Proc. of Performance '86 and ACM Sigmetrics '86 Conference*, May 1986, pp. 18-27.

[HaJu87] D.T. Harper III, J.R. Jump, "Performance Evaluation of Reduced Bandwidth Multi-stage Interconnection Networks", *Proc. International Symposium in Computer Architecture*, June 1987, pp. 171-175.

[HaPa87] P.G. Harrison, N.M. Patel, "The Representation of Switching Networks in Queueing Models of Parallel Systems", *Proceedings of the 12th International Symposium on Computer Performance*, Editors P.-J. Courtois and G. Latouche, Brussels, December 1987, pp.497-512.

[HaPa89] P.G. Harrison, N.M. Patel, "The Representation of Multi-stage Interconnection Networks in Queueing Models of Parallel Systems", Research Report, Dept. of Computing, Imperial College (Feb 1989).

[HaRe86] P. G. Harrison, M. J. Reeve, "The Parallel Graph Reduction Machine, ALICE", *Proc. Workshop on Graph Reduction*, Santa Fe, September 1986, to be published in LNCS series, Springer-Verlag.

[Hi85] D. Hillis, "*The Connection Machine*", MIT Press, Cambridge, Mass., 1985.

[Je83] T. Jenq, "Performance of a Packet Switch Based on Single Buffered Banyan Networks", *IEEE Journal on Selected Areas in Communications*, December 1983, pp. 273-282.

[JiBh88] H. Jiang, L.N. Bhuyan, "A Mean Value Analysis of Multi-stage Interconnection Networks", in preparation.

[JO86] Digital Optics : Application to Computing and Communications. DTI JOERS (Joint Opto-Electronic Research Scheme) proposal, Issue 2, March 1986.

[Ke79] F.P. Kelly, "*Reversibility and Stochastic Networks*", Wiley, Chichester, (1979).

[Ke86] F.P. Kelly, "Blocking Probabilities in Large Circuit-switched Networks", *Advances in Applied Probability* 18, pp. 473-505 (1986).

[KeZi85] F.P. Kelly, I. Ziedens, "Loss Probabilities in Circuit-switched Star Networks", unpublished.

[KiGa88] H.S. Kim, A.L. Garcia, "Performance of Buffered Banyan Networks Under Non-uniform Traffic Patterns", *Proc. IEEE INFOCOM*, March 1988, pp. 344-353.

[Kn68] D. Knuth, *Fundamental Algorithms*, Vol. 1, Addison Wesley, 1968.

[KoXe88] D.D. Kouvatsos, N.P. Xenios, "Maximum Entropy Analysis of General Queueing Networks with Blocking", *Proc. Int. Workshop on Queueing Networks with Blocking*, NC State University, pp. 172-200 (1988).

[KrSn82] C.P. Kruskal, M. Snir, "Some Results on Multi-stage Interconnection Networks for Multiprocessors", *Proc. Conference on Information Sciences and Systems*, Princeton University, March 1982.

[KrSn83] C.P. Kruskal, M. Snir, "The Performance of Multi-stage Interconnection Networks for Multiprocessors", *IEEE Transactions on Computers*, December 1983, pp. 1091-1098.

[Ku79] P.J. Kuehn, "Approximate Analysis of General Queueing Networks by Decomposition", *IEEE Transactions on Communications*, COM-27, No.1, 1979.

[KuPf86] M. Kumar, G.F. Pfister, "The Onset of Hot spot Contention", *Proc. 1986 International Conference on Parallel Processing*, August 1986, pp. 28-34

[KuRe87] V.P. Kumar, S.M. Reddy, "Augmented Shuffle-Exchange Multi-stage Interconnection networks", IEEE Computer, June 1987.

[La75] D.H. Lawrie, "Access and Alignment of Data in an Array Processor", *IEEE Transactions on Computers*, December 1975, pp.1145-1155.

[Li61] J.D.C Little, (1961), "A Proof of the Queueing Formula L = λW", *Operations Research* 9 (3), pp. 383-387.

[Li87] Y.S. Liu, 'The Delta Network Performance and the "Hot Spot" Traffic', Ultracomputer Project Technical Report (1987), Courant Institute, New York University.

[LVA82] T. Lang, M. Valero, I. Alegre, "Bandwidth of Crossbar and Multi-bus Connections for Multiprocessors", *IEEE Transactions on Computers*, Vol C-31, December 1982, pp. 1227-1234.

[MiCi87] D. Mitra, R. Cieslak, "Randomized Parallel Communications on an Extension of the Omega Network", *Journal of the ACM* Vol 34 No.4, October 1987, pp. 802-824.

[Pa81] J.H. Patel, "Performance of Processor-Memory Interconnections for Multiprocessors", *IEEE Transactions on Computers*, October 1981, pp. 771-780

[PaHa88] N.M. Patel, P.G. Harrison, "On Hot-spot Contention in Interconnection Networks", *Proceedings 1988 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems*, May 1988, Santa Fe, NM, pp. 114-123.

[PfNo85] G.F. Pfister, V.A. Norton, "Hot-spot Contention and Combining in Multi-stage Interconnection Networks", *Proc. 1985 International Conference on Parallel Processing*, August 1985, pp. 790-797.

[Po85] J.L. Potter, "*The Massively Parallel Processor*", MIT Press, Cambridge, Mass., 1985.

[PoHa88] A. Pombortsis, C. Halatsis, "Performance of Crossbar Interconnection Networks in Presence of "Hot Spots"", *Electronics Letters* 24, No.3 (1988), pp. 182-184.

[ReGr87] D.A. Reed, D.C. Grunwald, "The Performance of Multicomputer Interconnection Networks", *IEEE Computer*, June 1987, pp. 67-73.

[ReTh86] R. Rettburg, R. Thomas, "Contention is no obstacle to shared-memory multiprocessing", *Communications of the ACM*, Vol. 29, No. 12, December 1986, pp.1202-1212.

[SeMi81] K.C. Sevcik, I. Mitrani, "The Distribution of Queueing Network States at Input and Output Instants", *J. ACM*, Vol 28, No. 2, April 1981, pp. 358-371.

[Si85] H.J. Siegel, "*Interconnection Networks for Large-Scale Parallel Processing: Theory and Case Studies*", Lexington Books, 1985.

[St70] W.D. Strecker, "*Analysis of the Instruction Execution Rate in certain Computer Structures*", PhD dissertation, Carnegie-Mellon University, 1970.

[Wa87] I. Watson, J. Sargeant, P. Watson, V. Woods, "Flagship computational models and machine architecture", *ICL Technical Journal* 5, pp. 555-574 (1987).

[Wh84] W. Whitt, "Open and closed models for networks of queues", *AT&T Technical Journal* 63, 1911-1979, (1984)

[Wh85] W. Whitt, "Blocking when service is required from several facilities simultaneously", *AT&T Technical Journal* (1985), pp.1807-1856.

[WuBe72] Wulf, Bell, "C.mmp -- A Multi-mini Processor", *AFIPS FJCC* 1972, Vol. 41, pp. 765-777.

[WuFe80] C. Wu, T. Feng, "On a Class of Multistage Interconnection Networks", *IEEE Transactions on Computers*, Vol. C-29, No.8, August 1980, pp. 694-702.

[WuFe84] C. Wu, T. Feng, "*Tutorial: Interconnection Networks for parallel and distributed processing*", IEEE Computer Society Press 1984.

[YPD82] D. Yen, J.H. Patel, E. Davidson, "Memory Interference in Synchronous Multiprocessor Systems", *IEEE Transactions on Computers*, November 1982, pp. 1116-1121.