

Message Routing in Level 1 of the Wide-Band All-Optical Network

by

Stan James Reiss

B.S., Electrical Engineering
Cornell University, 1993

Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degrees of

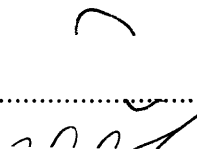
Master of Science in Electrical Engineering and Computer Science
and
Master of Science in Operations Research

at the
Massachusetts Institute of Technology
May 1993

© Stan Reiss. All rights reserved.

The author hereby grants to MIT permission to reproduce and to distribute publicly paper
and electronic copies of this thesis document in whole or in part.

Signature of Author.....



Certified by.....

Steven G. Finn, Lecturer, Department of Electrical Engineering and Computer Science
Thesis Advisor

Certified by.....

Alvin W. Drake, Professor of Systems Science and Engineering
Thesis Reader, Operations Research

Accepted by.....

Frederic R. Morgenthau, Chairman, Committee on Graduate Students

Accepted by.....

Thomas L. Magnanti, Director, Operations Research Center
MASSACHUSETTS INSTITUTE
OF TECHNOLOGY

JUL 17 1995

Message Routing in Level 1 of the Wide-Band All-Optical Network

by

Stan James Reiss

Submitted to the Department of
Electrical Engineering and Computer Science
in May, 1995,
in partial fulfillment of the requirements for the degrees of

Master of Science in Electrical Engineering
and
Master of Science in Operations Research

Abstract

We consider the middle hierarchical level of an All-Optical Network and develop the theories and algorithms that enable maximum utilization of its resources. We develop a method for optimally configuring the network, vis-à-vis the use of two routing devices, in order to maximize throughput. The method is applicable to arbitrary traffic patterns. We show that when the network is configured according to this method network delay is minimized at heavy traffic loads. We further show, using simulation, that under light loads the same network configuration results in average delay that approaches the minimum possible. Finally, we address the problem of dynamically scheduling network resources in order to minimize average delay. We use simulation to compare the performance of several scheduling algorithms.

Keywords: Wavelength Routing, All-Optical Networks.

Thesis Advisor: Dr. Steven G. Finn

Lecturer, Department of Electrical Engineering and Computer Science

Acknowledgements

This thesis is dedicated to my family, friends, and loved ones.

I would especially like to thank my thesis advisor, Dr. Steven Finn, for guiding me throughout this endeavor. This work would have been impossible without his insight and perspective.

Many thanks also go to Prof. Robert Gallager and Prof. John Tsitsiklis who were quick to offer their advice and helped me through many roadblocks.

The work was performed at the Laboratory for Information and Decision Systems (LIDS) at MIT. I would like to thank all members of the LIDS community for their various help throughout my stay there. I would particularly like to acknowledge Peter Marbach, Phil Lin, Steve Patek, and Muriel Medard for their input.

Finally, I would like to thank my fiancée, Kathy Glod, for her support and utmost patience. Her understanding and encouragement helped keep my spirits up throughout the thesis process.

This research was partially supported by the Advanced Research Project Administration Grant #MDA972-92-J-1038.

Contents

1. Introduction	13
1.1 Optical Networks	13
1.1.1 Electro-Optical Networks.....	14
1.1.2 All-Optical Networks	14
1.2 AON Consortium	15
1.3 AON Architecture	16
1.3.1 Level 0	17
1.3.2 Level 1	17
1.3.3 Level 2	19
1.4 Thesis Problem	19
1.5 Thesis Organization.....	20
2. Level 1 Network Model	21
2.1 λ -Router/Transmissive Star Configuration.....	21
2.2 Traffic Patterns.....	24
3. Maximum Throughput Analysis	27
3.1 Constraints	27
3.2 Necessary Conditions for Optimality.....	30
3.3 Optimum Configuration.....	33
3.4 Application to an Example.....	39
3.5 Collection of Traffic Data and Configuration Changes	40

4. Minimum Delay Analysis	43
4.1 Minimum Delay vs. Blocking Probability	43
4.2 Minimum Delay Configuration.....	47
4.3 Simulation Results and Discussion.....	48
5. Scheduling	51
5.1 The Need for Efficient Scheduling of the Transmissive Star	51
5.2 λ -Router Scheduling.....	52
5.3 Star Scheduling Strategies Considered.....	53
5.4 Delay Limits	56
5.5 Delay Calculations.....	57
5.6 Simulation Results.....	58
5.7 Discussion.....	62
5.7.1 Longest Queue vs. Bandwidth Reservation.....	62
5.7.2 Longest Queue vs. Round Robin	63
5.8 Call Redirection and the Longest Queue Strategy	64
6. Conclusion	67
A. Simulation	69
A.1 Simulation Tool.....	69
A.2 Simulation Model	69
A.3 Validation Using Theoretical Results	70
A.4 Simulation Code.....	71
A.4.1 Node Model Attributes.....	73
A.4.2 λ -Router Node Process Code.....	74
A.4.3 Transmissive Star Process Code Using Longest Queue Strategy	78
A.4.4 Transmissive Star Process Code Using Round Robin Strategy	80
A.4.5 Variable Initiation Node Process Model.....	82

List of Figures

1.1 AON Structure.....	16
1.2 AON Level 0 Structure	17
1.3 L1 λ -router/transmissive star combination	18
2.1 Illustrative Division of FSRs Between the λ -router and the Transmissive Star	23
2.2 A Sample Server Network.....	25
2.3 The Traffic Matrix for the Network of Figure 2.2	26
3.1 Example Optimal Solution as Dictated by Necessary Condition 3	33
3.2 Maximum Throughput allowed in a network using the star only, the λ -router only, and an optimal combination of the two.....	40
4.1 Network Model Using Independent Servers	44
4.2 Approximate Blocking Probability for an Example Network	45
4.3 Behavior of Minimum Delay r^λ vs. Network Load for an Example Network	48
4.4 Average Queuing Delay for a Sample Network.....	50
5.1 Average Delay Using Three Star Sharing Strategies.....	61
5.2 Round Robin Strategy	64
A.1 Network Model.....	71
A.2 λ -Router Node Process Model	72
A.3 Transmissive Star Process Model	72

List of Tables

4.1 Data for Figure 4.4.....	50
5.1 Comparison of the Average Delays vs. r^λ Generated by the Three Strategies under Heavy Load.....	60
5.2 Comparison of the Average Delays vs. r^λ Generated by the Three Strategies under Medium Load.....	60
5.1 Comparison of the Average Delays vs. r^λ Generated by the Three Strategies under Low Load.....	61
A.1 Comparison of Simulated and Theoretical Results	70

Chapter 1

Introduction

This chapter provides a general introduction to Optical Networks, and in particular to the network being constructed by the Wide-Band All-Optical Network Consortium. It also introduces the problem that is addressed in this thesis.

1.1 Optical Networks

In this thesis, we concern ourselves with high-speed optical communication networks. Modern high-speed networks almost always consist of nodes and optical links. The nodes, in addition to originating and receiving messages, perform functions such as scheduling and routing. The actual sources and destinations of messages are in general electronic and typically data is converted between electrical and optical signals within the network whenever it leaves or enters an optical fiber.

There are two general classes of optical networks: Electro-Optical Networks (EONs) and All-Optical Networks (AONs) [Bar93]. EONs are the conventional networks in which nodes convert a signal from optical to electronic form, route the message, and then re-convert it into an optical signal for transmission. EONs are also referred to as Multi-Hop Networks [Aca93] and Second Generation Optical Networks [Gre93]. In contrast to the EON, an optical signal in an AON is not converted back into electronic form until it leaves the network. All routing functions are performed in the optical domain. AONs are also called Third Generation Optical Networks [Gre93] and Single-Hop Networks.

1.1.1 Electro-Optical Networks

Virtually all modern high-speed communication networks, ranging from private FDDI rings to the long distance telephone networks, are EONs. EONs also support electronic store and forward routing, a substantial advantage over AONs where storing optical signals is problematic. Routing technology using electronic digital computers is commonplace and, for low data rates, inexpensive. However, electronic routing runs into problems at high data rates as electronic devices are limited to speeds that are low in comparison to those supported by optical fibers.

Using multiple electronic devices in parallel can compensate for their limited speed. However, because the potential data rate through an optical fiber is orders of magnitude higher than the maximum rate of an electronic device, a very large number of these devices would have to be set up to run in parallel in order to exploit the potential fiber bandwidth. The resulting cost/throughput tradeoff [Bar93] limits the maximum data rate achievable practically in an EON to one that is significantly lower than the capacity of an optical fiber.

1.1.2 All-Optical Networks

All-Optical Networks seek to improve the cost/throughput tradeoff. Since electronic routing is not used, the costs associated with numerous electronic components disappear and the entire fiber may be economically exploited. However, message routing must now be accomplished using only optical components -- a difficult technological challenge. Optical packet switching requires optical memory and lasers with very short tuning times. Optical circuit switching, while simpler than packet switching, still requires optical logic more complex than what is currently available.

Two alternatives for routing exist in an AON. The first involves using a broadcast star, a device which combines wavelengths from all origins and subsequently broadcasts them to all the destinations. The second, called wavelength routing, allows a node to select a destination by tuning to the appropriate wavelength and sending the message through a device called a λ -router. Wavelength routing allows for bandwidth re-use in different parts of the network but is less flexible than broadcast routing since individual wavelengths are assigned to specific routes through the network.

Networks using only λ -routers and transmissive stars as routing mechanisms have been shown to be unscalable, i.e., they can support only a limited amount of traffic, regardless of the ingenuity of the routing algorithm [Bar93]. Thus, until optical switching becomes practically and economically feasible, throughput in wavelength routed optical networks is limited.

1.2 AON Consortium

In 1993, AT&T, the Digital Equipment Corporation (DEC), and the Massachusetts Institute of Technology (MIT) formed a pre-competitive consortium to study all-optical networks. The goal of the consortium is to "address the challenges of utilizing the evolving terahertz bandwidth capability of optical fiber technology to develop a national information infrastructure capable of providing flexible transport, common conventions and common servers" [A+93]. Towards this end, the consortium is building a test-bed All-Optical Network (AON) at the members' sites in Massachusetts and New Jersey. The test-bed consists of small sub-networks (called Level 0 networks) at each of the members' sites. Eventually, the networks will be connected together in a hierarchical manner to form a large Wide Area Network. The AON will incorporate advanced optical and

opto-electronic technology to enable the utilization of the low-loss wavelength window of the single-mode optical fiber and the 4-5 THz passband of the erbium doped fiber amplifier.

1.3 AON Architecture

The AON architecture is based on a combination of wavelength division multiplexing (WDM) and time division multiplexing (TDM) technologies. The architecture provides for wavelength re-use within the AON. The AON is being designed with careful attention to scalability, modularity, and flexibility. Because current technological limitations make all-optical packet switching impractical, the test-bed AON is based on scheduled circuit-switching data services and unscheduled datagram control services. The AON currently uses broadcasting and wavelength routing to direct messages to their destinations.

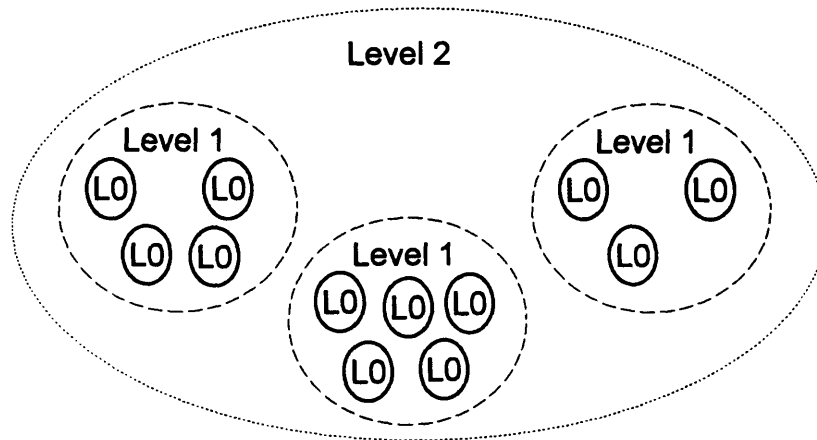


Figure 1.1: AON Structure

The AON is divided into three hierarchical levels: Level 0 (L0), Level 1 (L1), and Level 2 (L2). Figure 1.1 illustrates the network hierarchy.

1.3.1 Level 0

The L0's, which are analogous to Local Area Networks, lie at the lowest level of the AON. Figure 1.2 illustrates a Level 0 network. This is the level where Optical Terminals (OTs), or the network's users, attach to the network through an Access Point (AP). Each L0 has a set of wavelengths (λ 's) used specifically for intra L0 communication. These λ 's can be reused among different L0's since they are blocked from entering into the next hierarchical AON level, the L1. They are broadcast within each individual L0, thus eliminating any need for intra-L0 routing. The remaining λ 's in each L0 are sent to the L1 and used for inter-L0 and inter-L1 communication.

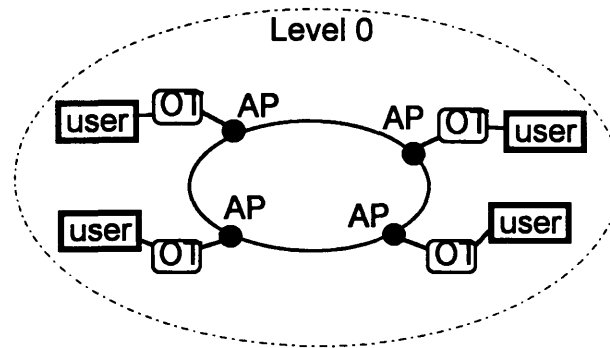


Figure 1.2: AON Level 0 Structure

1.3.2 Level 1

The L1 is the middle hierarchical level of the AON and may be viewed as a Metropolitan Area Network (MAN). This work mainly addresses the L1 network level of the AON. An L1 connects a number of L0 networks within a geographically limited area. It employs a device called a λ -router to route wavelengths between individual L0s and a device called a transmissive star to broadcast selected wavelengths to all the L0s connected to that L1 [A+93]. A λ -router is an optical device capable of receiving wavelengths from an

incoming fiber and sending different wavelengths to different output fibers. A transmissive star broadcasts all the incoming signals to all the outgoing fibers. A wavelength is either split off before reaching the λ -router and sent through the transmissive star or it is sent to the λ -router. Figure 1.3 illustrates the L1 λ -router/transmissive star combination. The λ -router operates identically over several sets of wavelengths referred to as Free Spectral Ranges (FSR's) so that, for example, the first wavelength in each FSR goes from $L0_i$ to $L0_{i+1}$, the second goes from $L0_i$ to $L0_{i+2}$, etc.

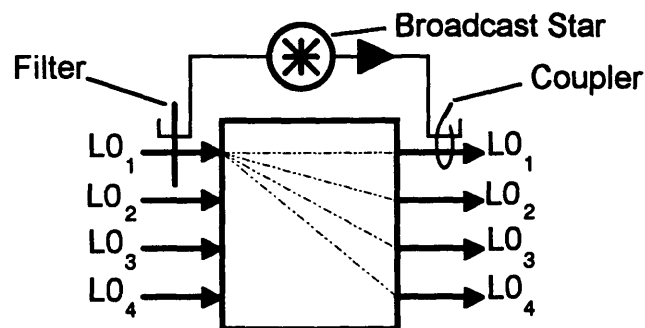


Figure 1.3: L1 λ -router/Transmissive Star Combination

A problem with the λ -router is that it allows for no bandwidth steering. Each $L0$ has a fixed number of λ 's available for communication with each other $L0$ in the λ -router and a heavy user can not "borrow" bandwidth from a user that is idle. The L1 transmissive star seeks to address this problem. Since anyone may use the star, a heavy user is able to utilize the star's bandwidth while others are idle. In addition, a transmissive star has multicast advantages. Any message transmitted over the star is received by all the $L0$ s in the L1. In short, the star has potential advantages for asymmetric (some channels requiring more bandwidth) point-to-point traffic as well as for both symmetric and asymmetric multicast traffic.

A problem with the transmissive star is that it decreases available point-to-point bandwidth. While the total point-to-point bandwidth through the λ -router is proportional to M^2 , where M is the number of L0s attached to the L1, the point-to-point bandwidth through the star is proportional to M . Therefore use of the star can be inefficient for symmetric point-to-point traffic when the aim is to maximize throughput. In short, the use of the star increases flexibility while decreasing total available point-to-point network bandwidth [Ale92].

1.3.3 Level 2

The AON Level 2 is the Wide Area Network connecting the individual Level 1 networks. The Level 2 network contains the expensive long-haul fibers and it is important that its resources are used efficiently. Many of the problems associated with routing and scheduling within the Level 2 are unique to that network level [Fin92].

1.4 Thesis Problem

In this thesis we develop the theories and algorithms that enable the AON Level 1 to maximize the utilization of its network resources. The resources available in the L1 are the λ -router and the transmissive star. Bandwidth can be dynamically allocated between the two devices resulting in a λ -router/transmissive star configuration (or network configuration). Due to current technological limitations, this configuration can only be changed slowly. We address the problem of configuring the L1 network in order to maximize throughput and minimize average delay. Furthermore, we develop an algorithm that allows for the efficient use of the network resources with stochastic traffic when the network is optimally configured.

1.5 Thesis Organization

We divide the problem of routing in the Level 1 of the AON into three separate sub-problems and address them sequentially. In Chapter 3, we use a constant flow model to maximize throughput in the Level 1 assuming a fixed traffic shape¹. This section of the thesis results in an algorithm that can be used to optimally divide the available bandwidth between the λ -router and the transmissive star for any given traffic pattern. In Chapter 4, we analyze the L1 from the standpoint of queueing delay. We search for a λ -router/transmissive star combination that will minimize average queueing delay paying particular attention to the configuration resulting in maximum throughput. Last, in Chapter 5 we tackle the problem of dynamically allocating the bandwidth of the broadcast star between the L0s in order to minimize the average delay. We use both analytical and simulation techniques to solve the problems addressed in this thesis.

¹The notion of "traffic shape" will be formally defined in Chapter 2.

Chapter 2

Level 1 Network Model

The purpose of this chapter is to introduce the level 1 network model and to formally define the problem to be solved. We begin by introducing the model and the notation that is used in this thesis. We follow by formulating several sub problems and finish by providing a brief problem summary.

2.1 λ -Router/Transmissive Star Configuration

We consider each Level 0 network to be a single node attached to the L1. Recall that all data is broadcast within an L0. The individual Optical Terminals that are the actual origins and destinations of calls have no impact on L1 routing, and their respective host L0s may be treated as the origin and destination nodes. Thus, though the L1 is really a network of networks, it is justifiable to model it as a network with "black box" nodes. We assume that each message or call has a single origin and a single destination node. The Origin-Destination (O-D) addresses are used to identify traffic.

In the AON, there are F wavelength channels available in each fiber and each node (L0) has an input fiber and an output fiber. The F wavelength channels are divided into R sets of wavelengths called Free Spectral Ranges (FSRs)². A channel is defined to be a single wavelength³.

²Recall from Chapter 1 that each FSR behaves identically in the λ -router.

³In an AON with Time Division Multiplexing (TDM) capability, the actual number of channels would be $T * F$, where T is the number of TDM slots per channel (wavelength).

Let Λ be the number of wavelengths available for inter-L0 communication in each FSR and R be the number of FSRs. The total number of channels available for inter-L0 communication is:

$$F = \Lambda * R \quad (2.1)$$

We require that the filter which routes wavelengths to the transmissive star is identical at each input fiber. This is necessary in order to prevent a transmissive star signal from colliding with a λ -router signal when the two sets of frequencies are coupled after routing. We assume that each L0 has one wavelength in each FSR dedicated to every other L0. Then, if the total number of nodes (L0s) is M , the number of wavelengths in each FSR, Λ , must be equal to M .

$$\Lambda = M \quad (2.2)$$

Since the λ -router can be used to send a wavelength between every pair of nodes over R spectral ranges, each node has R potential channels over which it can communicate with every other L0 through the λ -router. Note that any number of FSRs can be filtered off before the λ -router and sent directly to the broadcast star. Let r^* be the number of FSRs sent to the star, and, without loss of generality, let them be the first r^* FSRs (see Figure 2.1). Since each FSR contains $\Lambda = M$ wavelengths, the total number of channels available in the transmissive star is $M r^*$. Let r^\wedge be the number of FSRs left to go through the λ -router (last r^\wedge FSRs). Then the number of channels each node has available for communicating with each other node through the λ -router is equal to r^\wedge .

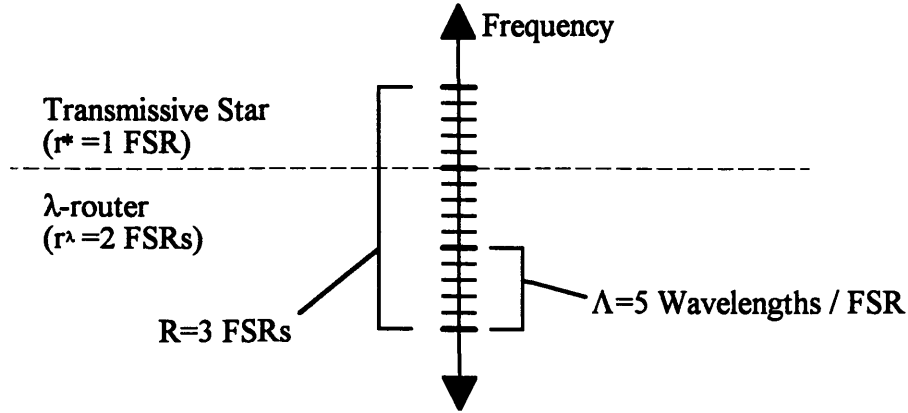


Figure 2.1: Illustrative Division of FSRs Between the λ -router and the Transmissive Star

In order to simplify the calculations, we from here on ignore the fact that r^λ and r^* must be integer valued. The relationship between r^λ and r^* is:

$$R = r^* + r^\lambda \quad (2.3)$$

$$0 \leq r^* \leq R$$

$$0 \leq r^\lambda \leq R$$

$$r^\lambda, r^* \in \mathfrak{R}$$

Note that each node has one wavelength in each FSR that simply returns to the same node (Figure 1.3). Also note that this wavelength is the same for all nodes. Since it is wasteful to send an inter-LO wavelength back inside the original node, it is logical that this wavelength should be filtered off and sent through the broadcast star. These "self-loop" wavelengths make up $1 / M$ of each FSR, and there are R FSRs. We have,

$$r^* \geq R / M \quad (2.4)$$

Furthermore, all the FSRs that were filtered off before the λ -router end up in the star, so:

$$\begin{aligned} r^* &= R / M + ((M-1) / M) (R - r^\lambda) \\ &= R - (1 - 1 / M) r^\lambda \end{aligned} \quad (2.5)$$

and

$$R = r^* + (1 - 1 / M) r^\lambda \quad (2.6)$$

Combining (2.4) with the physical bounds, we get,

$$\begin{aligned} R / M &\leq r^* \leq R \\ 0 &\leq r^\lambda \leq R \end{aligned} \quad (2.7)$$

While equations (2.6) and (2.7) give the exact relationships between r^λ and r^* , in order to simplify the math we will use the relationships described in (2.3), keeping in mind that we are ignoring some of the physical structure of the problem. As M gets large, the difference between the two becomes small.

2.2 Traffic Patterns

Let M be the number of nodes, or L0s, within the L1. We define a traffic matrix Π , where Π is an $M \times M$ matrix whose rows correspond to the origins and columns correspond to the destinations of traffic flow. The entries of Π correspond to the amount of traffic between corresponding O-D pairs, i.e. Π_{ij} is the amount of traffic flow from node i to node j . Π_{ij} is referred to as the traffic intensity on the O-D pair i,j . Traffic flow is measured in data per unit time. The traffic flow is initially assumed to be constant, i.e., node i is constantly sending Π_{ij} data units/time unit to node j . Later this assumption will be relaxed and Π_{ij}

will be interpreted to be the average number of data units or calls per unit time on the O-D pair i,j . Figure 2.2 shows a sample network and Figure 2.3 shows its associated traffic matrix.

The arrows in Figure 2.2 represent the direction of the traffic, their labels represent the traffic's intensity. Note that each entry in Π corresponds to an edge connecting two of the nodes in Figure 2.2.

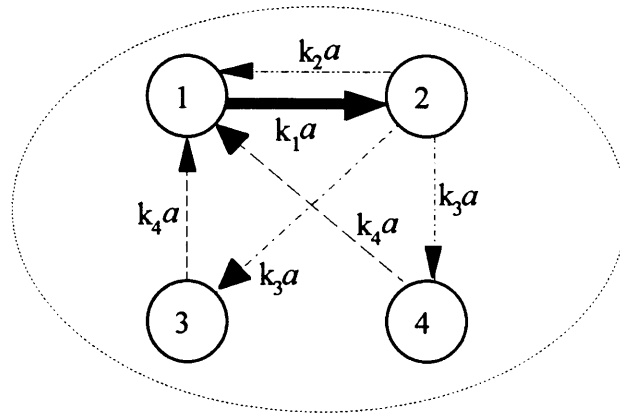


Figure 2.2: A Sample Server Network

We consider Π to be the product of a fixed "shape matrix" A with a variable scale factor a ,

$$\Pi = A a \tag{2.8}$$

It is the entries of A that determine the optimum network configuration. We assign each element of A to a value k_i , $i = 1, \dots, m$ where m is equal to the number of different numerical values in A (see Figure 2.3). We will consider all O-D pairs corresponding to

entries of A with value k_i to be the i th class, and we will consider this class to have intensity $k_i a$.

		Destination			
		1	2	3	4
Source	1	0	$k_1 a$	0	0
	2	$k_2 a$	0	$k_3 a$	$k_4 a$
	3	$k_4 a$	0	0	0
	4	$k_4 a$	0	0	0

Figure 2.3: The Traffic Matrix for the Network of Figure 2.2

Figures 2.2 and 2.3 depict a case where all nodes send traffic to node 1 which is presumably a server of some type. Node 1 then performs some calculations and sends a large amount of traffic to node 2, which distributes "responses" to all other nodes. The values of k_i are arbitrary, but from their definition $k_i \neq k_j$ for $i \neq j$.

Chapter 3

Maximum Throughput Analysis

This chapter involves the search for the values of r^λ and r^* which allow maximum network throughput for a given traffic shape (the maximum value of a for a given A matrix). We assume that the traffic flow is constant, i.e., node i is constantly sending Π_{ij} units of traffic to node j . We also assume that message flow is fluid and any fraction of it can be split off and sent through the transmissive star, leaving the remainder to be transmitted using the λ -router. These assumptions enable us to ignore the stochastics of the traffic and to focus solely on a single parameter for each O-D pair: the traffic intensity. The calculations are greatly simplified while the large-scale behavior characteristics are retained, leading to a simple and intuitive solution.

The analysis discussed above is based on a constant-shape traffic pattern. Real traffic generally changes with time. For example, the demands on a telephone network decrease and shift from the cities to the suburbs after the work day is over. In the last section of this chapter we discuss ways the network can dynamically adjust the value of r^λ in order to adapt to changing demands. We propose the use of a network controller to collect traffic data and discuss ways r^λ may be changed while the network is in use.

3.1 Constraints

Consider the general case with m classes of traffic intensity. Each class, i , has N_i member Origin-Destination (O-D) pairs and has intensity $k_i a$. In Figure 2.3, for example, $m = 4$,

$N_1 = 1, N_2 = 1, N_3 = 2,$ and $N_4 = 2$. We choose to number the classes so that

$$k_1 > k_2 > k_3 > \dots > k_m$$

We will temporarily ignore the special structure of the traffic shape matrix A , and treat all O-D pairs independently.

A fraction of the traffic from each O-D pair is sent through the transmissive star. Let this fraction be $x_{i,j}$, $i,j=1, 2, \dots, m$. The remainder, $1 - x_{i,j}$, goes to the λ -router. In order to maximize throughput, the sum of the traffic through the star and the λ -router must be maximized:

$$\max_{a, \bar{x}, r^\lambda} \sum_{i,j=1}^M A_{i,j} a$$

Subject to the constraints:

$$\sum_{i,j=1}^M A_{i,j} x_{i,j} a \leq Mr^* \quad (\text{star - bound})$$

$$A_{i,j} (1 - x_{i,j}) a \leq r^\lambda \quad (\lambda\text{-router bound}) \quad i, j = 1, 2, \dots, M$$

$$0 \leq r^\lambda \leq R \quad (\text{bandwidth constraint})$$

$$0 \leq x_{i,j} \leq 1 \quad i, j = 1, 2, \dots, M \quad (\text{routing constraint})$$

When one of the constraints (bounds) is satisfied with equality, we say that this bound is tight.

Equation (2.3) can be used to eliminate r from the above inequalities. Then, after some algebraic manipulation, the star bound reduces to

$$a \leq \frac{MR - Mr^\lambda}{\sum_{i,j=1}^M A_{i,j} x_{i,j}}$$

and the λ -router bound reduces to

$$a \leq \frac{r^\lambda}{A_{i,j}(1 - x_{i,j})}$$

The optimization problem is to maximize the traffic flow, which is proportional to a , by finding the optimal values of a , r^λ , and $x_{i,j}$, $i,j = 1, \dots, M$, for a given traffic shape matrix A .

Note that O-D pairs of the same class satisfy the same λ -router bound. If the fraction of traffic sent to the star is different for two O-D pairs belonging to the same class, at least one of the λ -router bounds is not tight. Then extra traffic belonging to that O-D pair could be taken from the transmissive star and sent through the λ -router. The space made available in the transmissive star could be used by all nodes and a could be increased, contradicting the optimality of the original configuration. Therefore, for optimal traffic flow, it is necessary to send the same fraction of traffic from each O-D pair belonging to a class to the star.

We can exploit the structure of the A matrix and reduce the dimension of the problem by using the above result. Since all $x_{i,j}$'s corresponding to O-D pairs of the same class are required to be the same in an optimal solution, we will assign a variable x_i to each class i . As with $x_{i,j}$ previously, x_i corresponds to the amount of class i 's traffic that will be sent to the transmissive star. The problem becomes:

$$\max_{a, \bar{x}, r^\lambda} \left(\sum_{i=1}^m N_i k_i \right) a \quad (3.1)$$

Subject to the constraints:

$$a \leq \frac{MR - Mr^\lambda}{\sum_{i=1}^m N_i k_i x_i} \quad (\text{star - bound}) \quad (3.2a)$$

$$a \leq \frac{r^\lambda}{k_i (1 - x_i)} \quad (\lambda\text{-router bound}) \quad i = 1, 2, \dots, m \quad (3.2b)$$

$$0 \leq r^\lambda \leq R \quad (\text{bandwidth constraint}) \quad (3.2c)$$

$$0 \leq x_i \leq 1 \quad i = 1, 2, \dots, m \quad (\text{routing constraint}) \quad (3.2d)$$

Note that a is the only variable term in (3.1). Thus the optimization problem is equivalent to maximizing a such that (3.2a) - (3.2d) are satisfied. We will refer to the maximum value of a as a_{\max} .

3.2 Necessary Conditions for Optimality

We will show that the following conditions are necessary for optimality:

1. The star bound (3.2a) is tight.
2. For each $i \in [1, \dots, m]$, if the λ -router bound (3.2b) is not tight then $x_i = 0$.
3. If $x_j = 0$ for some $j \in [1, \dots, m]$ then $x_i = 0$ for all $i > j$.

Corollary:

Let l be the minimum value of i such that $x_i = 0$. Then at the optimum solution $x_i = 0$ if and only if $i \geq l$.

Proof of necessary condition 1:

We prove necessary condition 1 by contradiction. Assume the star bound is not tight and $a = a_{\max}$. Let $a' = a + \varepsilon$, where $0 < \varepsilon \ll 1$. Now send all the new (ε) traffic to the star. None of the λ -router bounds change since we have not added any traffic there.

$$\begin{aligned} \text{if} \quad & \sum_{i=1}^m N_i k_i x_i a < MR - Mr^\lambda \\ \text{then} \quad & \sum_{i=1}^m N_i k_i x_i a + \left(\sum_{i=1}^m N_i k_i \right) \varepsilon < MR - Mr^\lambda \\ & \text{for } \varepsilon \text{ small enough} \end{aligned}$$

We just added traffic to the star. Because the star bound was not tight, for small enough ε it is still not violated. Thus the network can handle $a' = a + \varepsilon > a_{\max}$. This is a contradiction of the definition of a_{\max} which proves the star bound must be tight at $a = a_{\max}$.

Proof of necessary condition 2:

Proof of necessary condition 2 is also accomplished by contradiction. Assume that the λ -router bound is not tight and $x_i > 0$ for some i . Now take ε' of this class' traffic from the star and put it in the λ -router. This makes the star bound not tight, and for ε' small enough, the λ -router bound is not violated. By necessary condition 1, this configuration can not be optimal.

Proof of necessary condition 3:

To prove necessary condition 3, recall that by definition,

$$k_1 > k_2 > k_3 > \dots > k_m$$

Each wavelength channel must satisfy the λ -router bound. If $x_i = 0$, all the channel's traffic goes to the λ -router. Therefore, the flow must satisfy

$$k_i a_{\max} \leq r^\lambda$$

Now recall that for all $i > j$, $k_i < k_j$. We then have

$$k_i a_{\max} < r^\lambda \quad \text{all } i > j$$

and the λ -router bound is not tight for $i > j$. From 2 we know that if the λ -router bound is not tight for some $i \in [1, \dots, m]$ then x_i must equal 0. Thus, 3 is necessary for optimality.

Corollary to necessary condition 3:

Let l be the minimum value of i such that $x_i = 0$. Note that $0 \leq l \leq m+1$. We know from condition 3 that $x_i = 0$ for all $i > l$. By hypothesis, $x_j > 0$ for $j < l$. Thus $x_i = 0$ if and only if $i \geq l$ and l is the index of the heaviest traffic class that does not use the star. Figure 3.1 illustrates the result proven by the corollary through an example optimal solution. All optimal solutions must have the form of the solution illustrated in Figure 3.1.

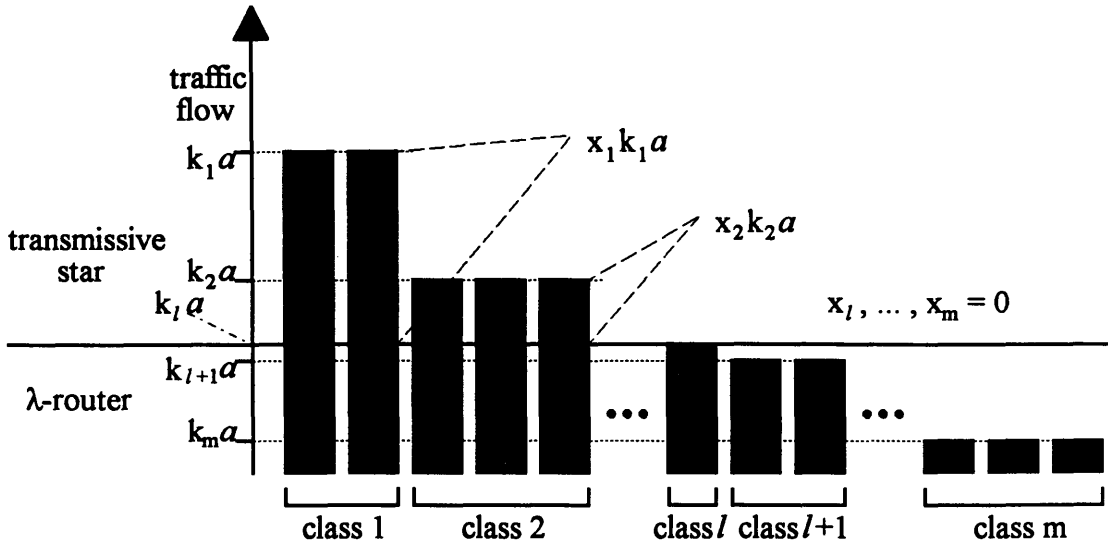


Figure 3.1: Example Optimal Solution as Dictated by Necessary Condition 3

3.3 Optimum Configuration

We use necessary conditions 1 - 3 to reduce the dimension of the feasible space. While the initial feasible space was parametrized by r^λ and x_i for $i = [1, \dots, m]$, the set satisfying the necessary conditions is fully parametrized by r^λ and the index l . We optimize over this two dimensional set and find the values of r^λ and l that allow for the largest value of a , and thus the maximum throughput.

Note that the following are consequences of the necessary conditions:

1. $x_i = 0$ for all $i \geq l$ (necessary condition 3).
2. The λ -router bound must be tight for each class that sends traffic to the star ($i < l$), and thus only $k_i a - r^\lambda$ traffic from each O-D pair in each of these classes uses the star (necessary condition 2).
3. The star traffic is $\sum_{i=1}^{l-1} (k_i a - r^\lambda) N_i$ (implied by 1 and 2 above).

The star's capacity is $Mr^* = MR - Mr^\lambda$. The star's traffic cannot exceed capacity, so by 3 above, the following star bound holds:

$$\sum_{i=1}^{l-1} (k_i a - r^\lambda) N_i \leq MR - Mr^\lambda$$

Solving for a , we find

$$a \leq \frac{MR + r^\lambda \left(\sum_{i=1}^{l-1} N_i - M \right)}{\sum_{i=1}^{l-1} k_i N_i} \quad (3.3)$$

Inequality (3.3) summarizes necessary conditions 1 - 3 as well as bounds (3.2a), (3.2b), and (3.2d). Because (3.3) arises from conditions necessary for optimality, together with bound (3.2c) and the limits on l ($0 \leq l \leq m+1$), it fully specifies the set of solutions that may achieve the optimum. Note that (3.3) is the star bound, and from necessary condition 1 we know that at the maximum value of a this bound is tight. Therefore, by maximizing the right hand side of (3.3) we also maximize a . The right hand side of (3.3) is parametrized only by r^λ and l , so the search for an optimum solution involves only varying r^λ and l .

We define

$$\sum_{i=1}^0 N_i \equiv 0$$

and consider three cases of l :

$$\sum_{i=1}^{l-1} N_i - M < 0 \quad (\text{Case 1})$$

$$\sum_{i=1}^{l-1} N_i - M > 0 \quad (\text{Case 2})$$

$$\sum_{i=1}^{l-1} N_i - M = 0 \quad (\text{Case 3})$$

Case 1:

If

$$\sum_{i=1}^{l-1} N_i - M < 0 \quad (3.4)$$

then, from (3.3), α is maximized when r^λ is minimized.

From the λ -router bound and the fact that by definition $x_l = 0$ we have

$$r^\lambda \geq k_l a$$

which implies

$$r_{\min}^\lambda = k_l a \quad (3.5)$$

and the λ -router bound is tight for $i = l$. Substituting (3.5) into (3.3), we get the relation

$$\alpha \leq \frac{MR}{\sum_{i=1}^{l-1} (k_i - k_l) N_i + k_l M}$$

The objective is to maximize this bound on a by varying l . The right hand side is maximized when its denominator is minimized. Consider decreasing l by 1. As long as l is positive (3.4) remains valid. The change in the denominator is

$$\begin{aligned}\Delta \text{ denominator} &= \sum_{i=1}^{l-2} (k_i - k_{l-1}) N_i + k_{l-1} M - \sum_{i=1}^{l-1} (k_i - k_l) N_i - k_l M \\ &= (k_{l-1} - k_l) \left(M - \sum_{i=1}^{l-1} N_i \right)\end{aligned}$$

From (3.4) and the fact that $k_{l-1} > k_l$, we see that the above quantity is positive. Therefore decreasing l increases the value of the denominator of the bound and thus decreases the maximum throughput. Conversely, as long as (3.4) holds, increasing l increases throughput. We conclude that l should be made as large as possible while (3.4) remains valid.

Case 2:

When

$$\sum_{i=1}^{l-1} N_i - M > 0 \quad (3.6)$$

then, from (3.3), a is maximized when r^λ is maximized. But because, by the definition of l , $x_{l-1} > 0$ the λ -router bound is tight for $i = l-1$ (necessary condition 2) and

$$\begin{aligned}r^\lambda &= (1 - x_{l-1}) k_{l-1} a && \text{(necessary condition 2 and (3.2b))} \\ r^\lambda &< k_{l-1} a && (x_{l-1} > 0)\end{aligned}$$

Again, using (3.3) and solving for a gives

$$a < \frac{MR}{\sum_{i=1}^{l-1} (k_i - k_{i-1})N_i + k_{l-1}M} = \frac{MR}{\sum_{i=1}^{l-2} (k_i - k_{i-1})N_i + k_{l-1}M}$$

Note that the upper bound on a here is strictly less than the one obtained by letting $l = l - 1$. In this case we decrease l and get a higher upper bound on a . We repeat this process until (3.6) no longer holds, (3.4) holds, and it ceases to be optimal to decrease l . Therefore if (3.6) holds, for optimality it is necessary to decrease l until (3.6) no longer holds. Putting this together with the situation where (3.4) holds, we see that when (3.4) can not be satisfied with equality, the optimal value of l is the maximum integer that satisfies (3.4). There is only one value of r^* that maximizes (3.3), and thus the value of r^* at $a = a_{\max}$ is unique.

Case 3:

It remains necessary to treat the final case:

$$\sum_{i=1}^{l-1} N_i - M = 0 \quad (3.7)$$

If l is increased, (3.6) will be satisfied -- a non-optimal situation. If it is decreased, (3.4) will be satisfied and the appropriate relations will hold. If l is not changed, (3.3) is optimized for any value of r^* satisfying

$$r^* \geq k_l a$$

and

$$r^* < k_{l-1} a$$

The bound on a then is, from (3.3),

$$a \leq \frac{MR}{\sum_{i=1}^{l-1} k_i N_i} \quad (3.8)$$

If l is decreased, the bound on a is

$$a \leq \frac{MR}{\sum_{i=1}^{l-2} (k_i - k_{l-1}) N_i + k_{l-1} M} \quad (3.9)$$

and

$$r^\lambda = k_{l-1} a$$

When (3.7) is true (3.8) and (3.9) are equal. Thus, when (3.7) holds, optimum throughput is achieved when

$$k_l a \leq r^\lambda \leq k_{l-1} a$$

And in particular, if

$$r^\lambda = k_l a$$

Therefore, if there exists an l that satisfies (3.7), a tight λ -router bound at $i = l$ yields one of many optimal values of r^λ , all of which allow a to reach a_{\max} . The index l has two optimal values in this case: one satisfies (3.7), and one is the maximum integer satisfying (3.4).

We conclude that the following relation, when satisfied, yields the optimum value of r^λ and a corresponding optimum value of a :

$$a_{\max} = \frac{MR}{\sum_{i=1}^{l^*-1} (k_i - k_{l^*}) N_i + k_{l^*} M} \quad (3.10)$$

$$\Gamma^\lambda = k_r \alpha_{\max} \quad (3.11)$$

for some l^* . From the discussion of cases 1-3 it is clear that if l^* is the maximum value of l satisfying

$$\sum_{i=1}^{l-1} N_i \leq M \quad (3.12)$$

then maximum throughput will be achieved. If l^* is one then the optimum configuration involves a pure λ -router. Likewise, if $l^* > m$ then the optimum configuration involves a pure transmissive star.

3.4 Application to an Example

The following example illustrates how the use of a transmissive star/ λ -router combination increases throughput of the L1 network for asymmetric traffic patterns over the use of either the star or the λ -router alone. Consider the simple example where $m = 2$, $M = 4$, $R = 3$, $N_1 = 1$, and $N_2 = 7$. Thus $l^* = 2$ and

$$\alpha_{\max} = \frac{MR}{\sum_{i=1}^1 (k_i - k_2) N_i + k_2 M}$$

$$\Gamma^\lambda = k_2 \alpha_{\max}$$

$$\Gamma^* = 3 - \Gamma^\lambda$$

The maximum throughput is

$$\sum_{i=1}^2 N_i k_i \alpha_{\max}$$

Figure 3.2 compares the maximum throughput allowed in the network using the star only, the λ -router only, and an optimal combination of the two as a function of k_1 for $k_2 = 1$ and $R = 3$. Note that the case when $k_1 = 1$ is equivalent to having one class of intensity 1 with $N_1 = 8$.

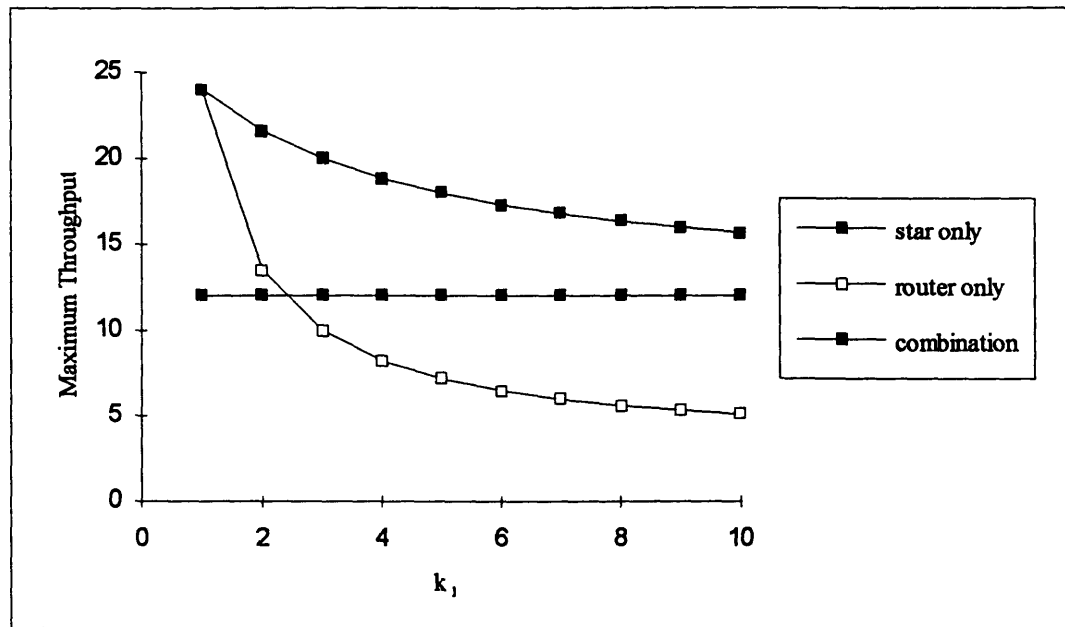


Figure 3.2: Maximum throughput allowed in a network using the star only, the λ -router only, and an optimal combination of the two.

$$(M = 4, R = 3, k_2 = 1, N_1 = 1, N_2 = 7)$$

3.5 Collection of Traffic Data and Configuration Changes

The maximum throughput configuration is based on a constant traffic pattern. Real traffic patterns change with time. It is therefore necessary to address the problem of dynamically adapting the L1 to real traffic changes. Changing the network configuration requires that a wavelength (or a number of wavelengths) be moved from the λ -router to the

transmissive star or vice versa. In order for this to be accomplished, all calls on this wavelength (or wavelengths) must be terminated. While the network is waiting for such calls to terminate it can not use these wavelengths and thus changes to the configuration carry with them an overhead efficiency loss. In order to minimize this overhead, frequent changes to the network configuration should be avoided.

One possible approach to collecting traffic data is to implement a network controller to monitor traffic. Because network configuration changes should be infrequent, a network controller should have sufficient time between changes to collect traffic data and update the traffic shape matrix. Thus it is reasonable for traffic shape data to be collected by polling the individual LOs about the number of call requests received since the previous poll. The actual method for collecting traffic data and the protocol for network configuration updates are left as an open problem for further research.

Chapter 4

Minimum Delay Analysis

An AON operating at loads approaching capacity will have finite queues only if r^λ is chosen to maximize throughput. Networks operating at lower loads can handle the traffic at multiple values of r^λ and other performance measures must be used to determine the optimum r^λ . In this chapter we analyze the use of two stochastically-based performance measures, blocking probability and queueing delay, and show queueing delay is the more appropriate measure. We then use analysis and simulation to find the value of r^λ that will result in minimum average queueing delay. We pay particular attention to the value of r^λ that results in maximum throughput using the model developed in Chapter 3.

4.1 Minimum Delay vs. Blocking Probability

Communication networks usually operate at loads that are below their capacity. This will be especially true for the All-Optical Network, as its capacity will eventually be measured in terabits and few applications require that much bandwidth. Therefore, in most instances the L1 will be able to support its traffic at many values of r^λ . It makes sense to set r^λ to optimize the network performance vis-à-vis some performance measures, such as blocking probability or delay.

Blocking probability is the probability that a call request is denied, i.e., the probability that the O-D pair that requests bandwidth finds no resources available. The common simplifying assumption used when analyzing networks from the standpoint of blocking

probability is that blocked calls are lost. A request that has been denied disappears as if it never happened. This assumption significantly simplifies the models used.

The delay is the time a call waits in queue before it is serviced. Delay is measured from the time the LO node requests bandwidth for a call until the time the call is given access to a server (either in the transmissive star or in the λ -router). Calls that can not be served are queued and wait for a server. We assume that calls are served using a First Come First Serve (FIFO) discipline throughout this work, but since we model all call service times as independent and identically distributed random variables, the service discipline does not affect average delay [BG92]. Each O-D pair has its own queue that is independent of the queues of all other O-D pairs. Average delay is calculated by averaging the delay experienced by all calls in the network, so each O-D pair's delay is effectively weighted by its arrival rate.

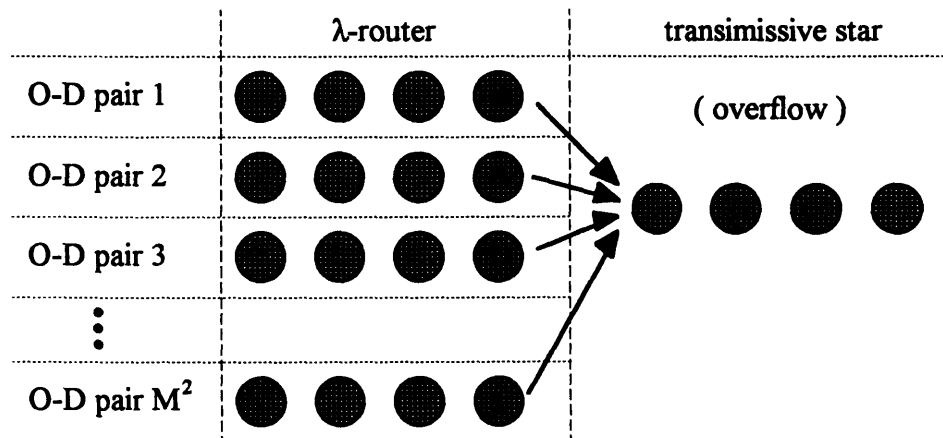


Figure 4.1: Network Model Using Independent Servers ($r^\lambda = 4$, $M r^* = 4$)

First, we analyze blocking probability as a performance measure. To simplify calculations assume that arrivals to the network are Poisson and that hold times are exponential. The network can then be modeled as groups of independent servers (λ -router channels) and a

group of shared overflow servers (transmissive star channels). Each O-D pair has access to r^{λ} proprietary servers and to Mr^{λ} common servers that it shares with all O-D pairs (see Figure 4.1).

Even with these simplifying assumptions the blocking probability calculations turn out to be intractable [Wol89]. Fortunately, a recursive method developed by R. Wilkinson to model telephone traffic called the Equivalent Queue Random Method (EQRM) [Wil56] gives a reliable approximation for blocking delay. Wilkinson notes that calls are blocked only when the transmissive star (which, in his analysis, is a group of secondary servers) is full and a call request is made. He models the first group of independent servers (primary servers) as a single random server which behaves similarly to the aggregate group of primary servers. He is then able to generate a group of recursive equations that define the stochastics of the secondary servers and produce an approximation to the blocking probability in the network.

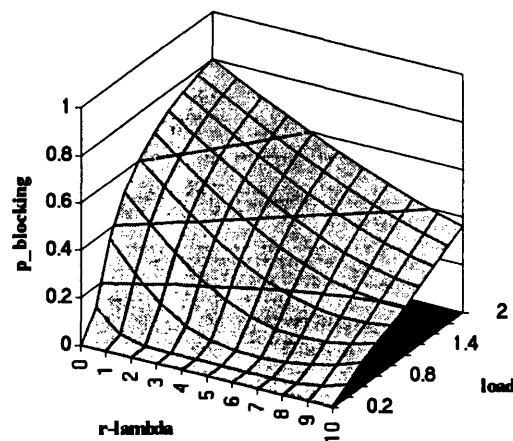


Figure 4.2: Approximate Blocking Probability for a Network with $M = 4$, $R = 10$, $N_1 = 1$, $k_1 = 5$, $N_2 = 15$, and $k_2 = 5$

The EQRN method, when applied to a simple network with $M = 4$, $R = 10$, $N_1 = 1$, $k_1 = 5$, $N_2 = 15$, and $k_2 = 5$, produced the graph of the blocking probability in the L1 as a function of r^λ and the traffic load shown in Figure 4.2. The traffic load ρ is defined to be the ratio of a to a_{\max} .

The maximum throughput value of r^λ for this network is five. Figure 4.2 illustrates that for low traffic intensity the blocking probability is minimized near $r^\lambda = 5$. However, as the traffic intensity increases, so does the value of r^λ that minimizes the blocking probability.

This result might suggest a simple solution to the heavy load scenario: regardless of the traffic shape, set r^λ to its maximum value and the blocking probability will be minimized. While the maximum value of r^λ will in fact minimize the blocking probability, the resulting dynamics are highly undesirable in that they violate the traffic shape.

Consider the example of Figure 4.2. The traffic intensity on one O-D pair is five times higher than that on all the other pairs. In a high load scenario the blocking probability is minimized when 4/5 of the traffic on the heavy-intensity O-D pair is blocked and thus the traffic is "uniformized." Uniform traffic is always best handled by a pure λ -router since the overall bandwidth is maximized and every O-D pair gets an equal share.

The example can be generalized to show that regardless of the traffic shape, under heavy loads blocking probability is minimized when the traffic is "uniformized" and a pure λ -router is used. This approach does not preserve the traffic shape. If the above example was a client-server communication network, 4/5 of the server traffic would be discarded. This is clearly not desirable in such a network. As a result, blocking probability does not appear to be an appropriate measure for network performance optimization.

The other network performance measure we consider is average network delay. In the delay model call requests are queued until they are served, and thus traffic shape is preserved.

At loads approaching capacity minimizing delay is consistent with maximizing the throughput. To see this, note that a traffic load near capacity can only be handled at one value of r^λ . At this value of r^λ the average delay is finite. The traffic can not be handled at any other values of r^λ and the average delay for these configurations will be infinite. As maximizing throughput will select the only value of r^λ that allows for finite delay, at heavy loads minimizing delay is equivalent to maximizing throughput. Thus, average delay appears to be viable for use as a network performance measure. Its use preserves the traffic shape.

4.2 Minimum Delay Configuration

Network delay is in general a function of the service strategy and the resources available for serving messages or calls. Total L1 bandwidth is maximized when a pure λ -router configuration is used, but this bandwidth can not be shared. The sharing of bandwidth in general reduces network delay. This is best demonstrated by the example of a communication channel with M users where users will experience lower delay using statistical multiplexing to share the channel than using frequency division multiplexing or time division multiplexing [BG92].

The introduction of stochastics into the network model creates two opposing forces acting to minimize average delay. One force calls for maximizing the bandwidth and pushes the L1 towards a pure λ -router. The other calls for maximizing the statistical multiplexing gain and pushes the L1 towards a pure transmissive star. In general, the maximum

throughput configuration will fall somewhere in the middle. Recall that at loads near capacity the maximum throughput configuration also results in minimum delay. At low loads it is unclear what configuration minimizes delay.

4.3 Simulation Results and Discussion

At loads approaching capacity only one value of r^λ results in finite average delay. As the load decreases, more values of r^λ become feasible and the maximum throughput r^λ is always one of them. The maximum throughput r^λ results in the network configuration with the most capacity for a particular traffic shape. It is therefore reasonable that the minimum delay r^λ would move away from the maximum throughput r^λ in continuous fashion as the load decreases (see Figure 4.3). Because at low loads the statistical multiplexing gain offered by the star promises to decrease delay, it also seems reasonable that as the load drops off the minimum delay r^λ might move from the maximum throughput r^λ towards zero.

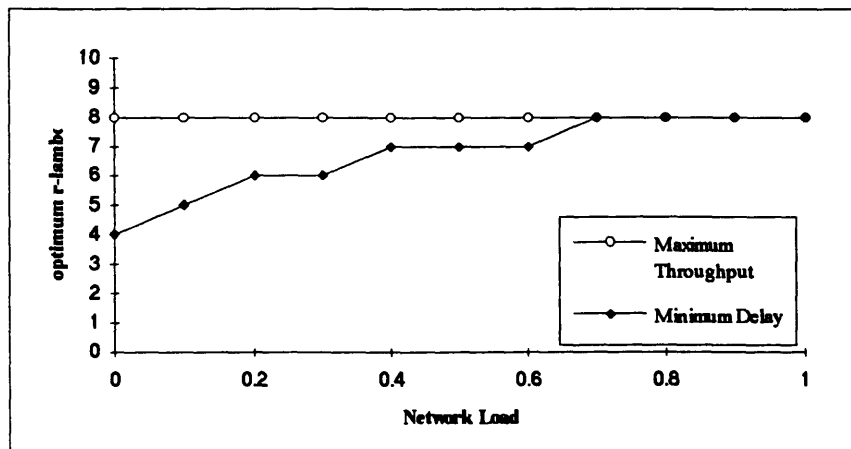


Figure 4.3: Behavior of Minimum Delay r^λ vs. Network Load for an Example Network with $R = 10$ and Maximum Throughput $r^\lambda = 8$

While the above argument would suggest that at low loads the r^λ that maximizes throughput analysis is unlikely to minimize delay, there is an additional dimension to the problem. When the load on the network is low, the value of the average delay is an insignificant number. For example, at $\rho = .5$, the average delay in the network illustrated in Figure 4.3 may be 2×10^{-7} s when $r^\lambda = 8$ and 1×10^{-7} s when $r^\lambda = 7$. Thus, one may set r^λ to 8 and optimize the network configuration for maximum traffic at a very low cost. This rationale would suggest that by selecting r^λ to maximize network throughput one can guarantee good delay performance for all values of ρ .

Simulation was used to explore whether the arguments above are justified. The simulation tool used was MIL3's Opnet (see Appendix A). In the interest of consistency we use a network of $M = 4$ and $R = 10$ with a traffic shape of $k_1 = 2$, $N_1 = 1$, $k_2 = 1$, $N_2 = 15$ for all simulations. The maximum throughput value of r^λ for this traffic shape is 8. We vary the value of a to control the traffic load. Figure 4.4 shows simulation results under three traffic loads, the data is listed in Table 4.1.

The simulation results described by Figure 4.4 and Table 4.1 show that at heavy and medium loads the maximum throughput r^λ is also the minimum delay r^λ . Furthermore, they show that at low loads, while the maximum throughput and minimum delay values of r^λ may be different, their difference in their respective delays is so small that it is negligible. The simulations support the earlier conclusion that good network performance in terms of minimum average delay is achieved when r^λ is chosen to maximize the potential throughput for a particular traffic shape.

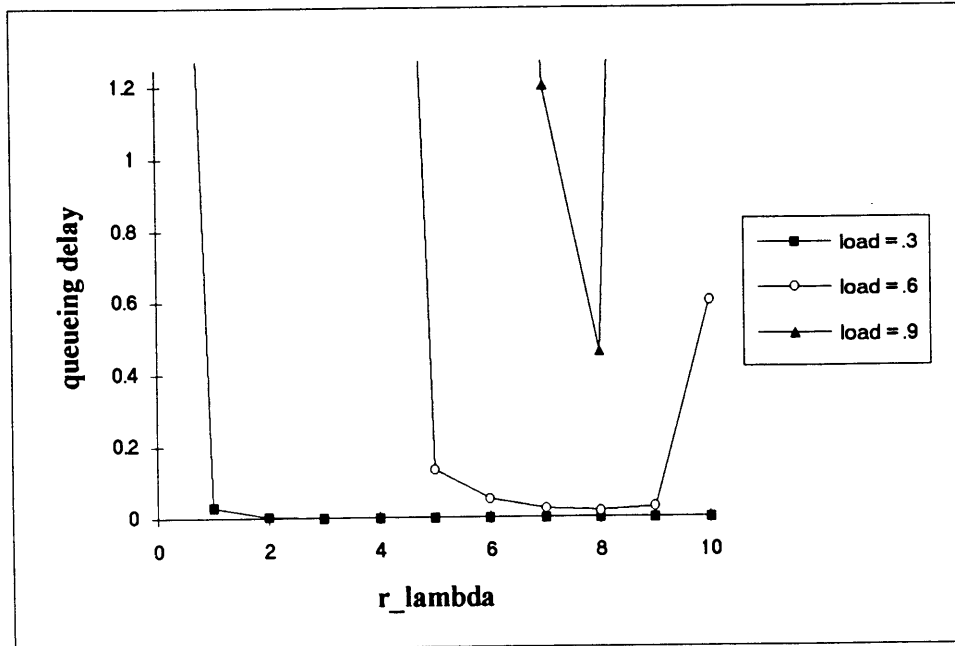


Figure 4.4: Average Queueing Delay for a Sample Network of $M = 4$ and $R = 10$ with traffic shape $k_1 = 2$, $N_1 = 1$, $k_2 = 1$, $N_2 = 15$ at $\alpha = 1, 2.5, 5$, and 7 . (Maximum Throughput $r^\lambda = 8$)

r^λ	$\rho \approx .1$	$\rho \approx .3$	$\rho \approx .6$	$\rho \approx .9$
0	2×10^{-8}	∞	∞	∞
1	0	.03	∞	∞
2	0	.0016	∞	∞
3	0	7×10^{-5}	∞	∞
4	0	2.5×10^{-6}	∞	∞
5	0	3×10^{-7}	.133	∞
6	0	5×10^{-9}	.05	∞
7	0	1.5×10^{-7}	.024	1.2
8	0	1.5×10^{-6}	.0175	.46
9	0	4.25×10^{-5}	.027	∞
10	5×10^{-7}	9.3×10^{-4}	.6	∞

Table 4.1: Data for Figure 4.4

Chapter 5

Scheduling

In this chapter we address the problem of scheduling the transmissive star. The star can be used by all the O-D pairs in the network. We propose several algorithms for allocating of the star's bandwidth and identify the most efficient one based on simulation results and analysis.

5.1 The Need for Efficient Scheduling of the Transmissive Star

Assume that the arrival traffic to the network is Poisson and that call hold times are exponential. Also assume that r^* has been selected to maximize throughput. In this scenario it might seem that any work conserving service policy should produce the minimal average queueing delay [BG92]. It therefore might seem that it is irrelevant how the transmissive star is scheduled as long as it is working whenever there is work to be done.

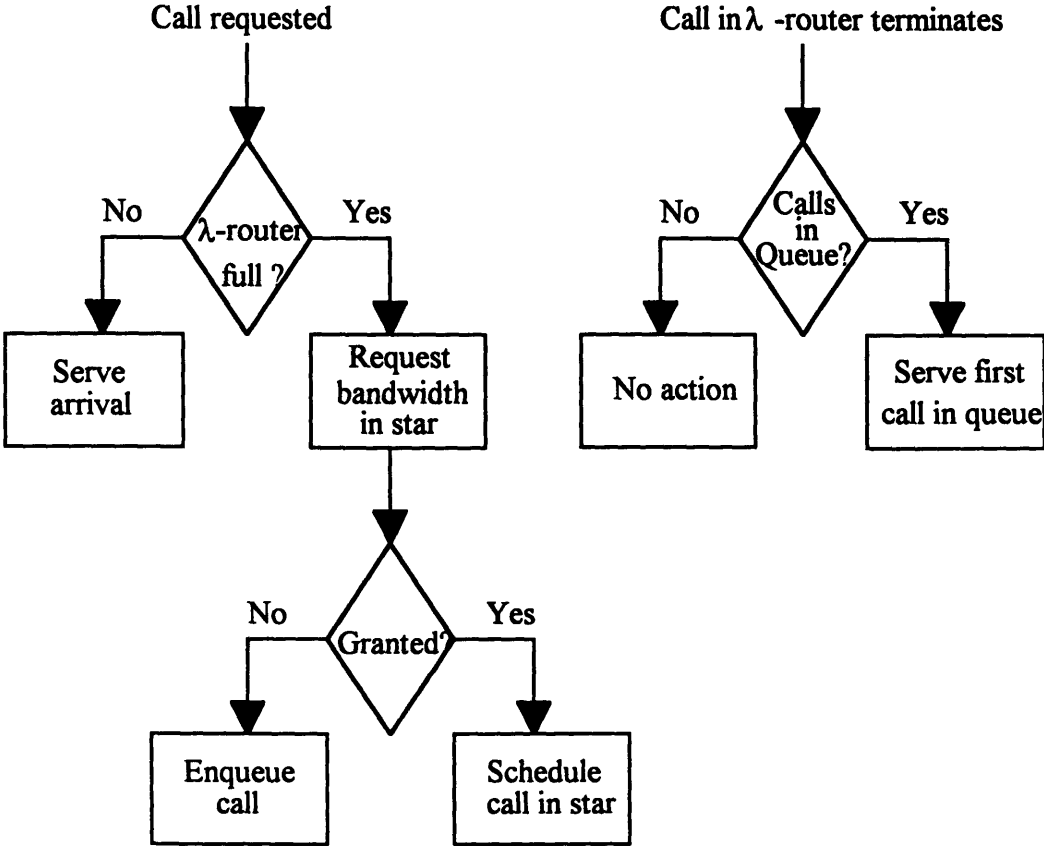
Unfortunately, the above argument is incorrect when applied to the Level 1 of the AON. The reason it is incorrect is that the AON Level 1 is intrinsically non work-conserving. A scenario that clearly demonstrates this involves two O-D pairs. One has many active users and needs all the resources it can get. The other has no active users. In a work-conserving network, the busy O-D pair would borrow λ -router channels from the idle one.

Such bandwidth sharing is impossible in the λ -router. Thus, even the best strategy for sharing the transmissive star in the L1 will not be work-conserving.

In a non work-conserving network the service strategy directly affects the queuing delay. As this effect can be very significant, we study several possible strategies for scheduling the use of the transmissive star in the L1.

5.2 λ -Router Scheduling

We assume that the λ -router follows the general strategy outlined below for all O-D pair call requests.



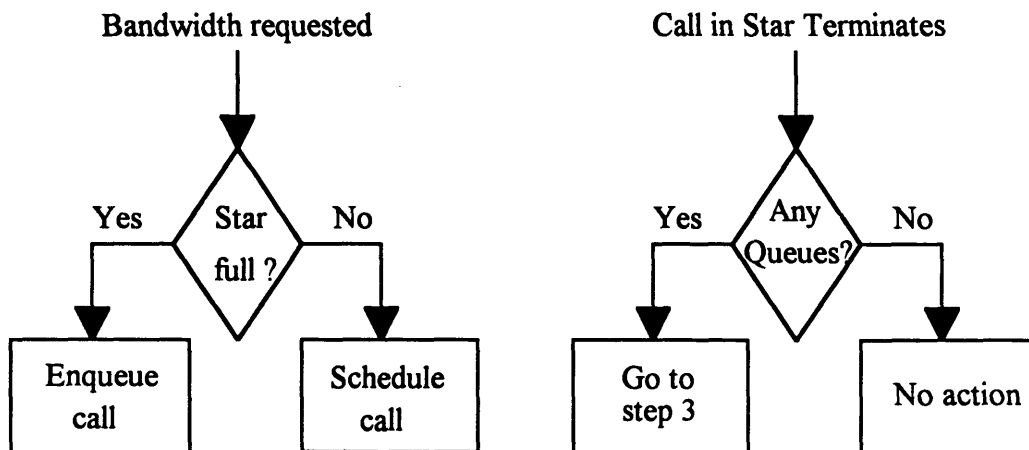
When the λ -router is full, the O-D pair requests bandwidth in the transmissive star. If it is granted bandwidth, the call is scheduled in the star. If it is denied, the call is enqueued. The decision is based on the availability of the star and the star sharing strategy in use. We make the assumption that if a call is scheduled in the star it will use the star until it terminates.

5.3 Star Scheduling Strategies Considered

Because of the beneficial effect statistical multiplexing gain usually has on average queueing delay, we focus on star sharing strategies that allow access to the star for all O-D pairs. The first strategy we consider is the round robin strategy. The round robin strategy is outlined below.

Round Robin Strategy

1. Let $i = 0$.
2. Follow the diagram below:



3. Let $j = i$.
4. Determine whether O-D pair j has calls in queue. If yes, let $i = (i + 1) \bmod M^2$, schedule first enqueued call from O-D pair j in star, and go to 2. If not, go to 5.

5. Let $j = (j + 1) \bmod M^2$. If $j = i$ then all queues empty, go to 2. Else go to 4.

The round robin strategy ensures that all active O-D pairs have equal opportunity to use the star. Thus, this strategy allows for fair sharing of the star as well as for bandwidth flexibility.

The round robin strategy has a critical problem. At loads near capacity, each O-D pair uses approximately $1/M^2$ of the capacity of the star. Each O-D pair uses approximately the same amount of bandwidth, and the effect is the "uniformization" of the traffic in a manner similar to that caused by the use of blocking probability as a network performance measure (Chapter 4). Thus, at loads near capacity the round robin strategy fails to preserve the traffic shape.

The bandwidth reservation strategy, outlined below, ensures that the traffic shape is preserved at all loads.

Bandwidth Reservation Strategy

1. Permanently divide up the bandwidth of the transmissive star in the following manner:
To each member of class i , assign $x_i k_i a_{\max}$ bandwidth in the star, where x_i is the optimal fraction of class i 's traffic assigned to the star under the fluid maximum throughput analysis, k_i is class i 's intensity, and a_{\max} is the maximum value of a (Chapter 3).
2. Do not allow any bandwidth sharing.

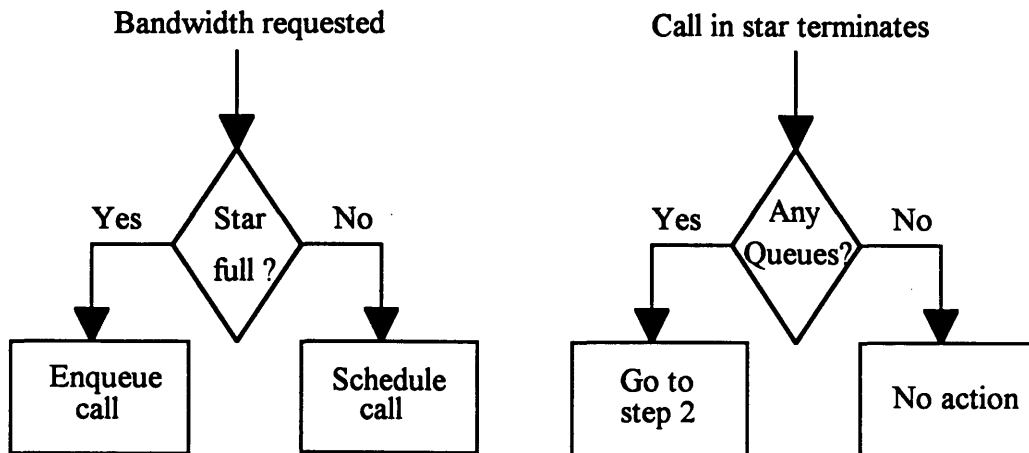
This strategy will in most cases call for fractional division of the channels available in the transmissive star. This can be accomplished either through TDM or through time sharing.

Unlike the round robin strategy, the bandwidth reservation strategy ensures that all O-D pairs have the bandwidth necessary for queue stability as long as $a < a_{\max}$. It does not operate well at low loads since it allows for no dynamic bandwidth sharing. However, the argument can be made that at low loads the delay is so small that the exact level of performance is negligible.

The round robin strategy should perform well at small values of ρ and the bandwidth reservation strategy should perform well at high values of ρ . It is desirable to find a strategy that operates well over the entire range of ρ . The final strategy analyzed has the potential of accomplishing this goal. We call this strategy the longest queue strategy and outline it below.

Longest Queue Strategy

1. Follow the diagram below:



2. Determine which O-D pair has the most calls in queue. Send first enqueued call from this O-D pair to star and go to 1.

Under this strategy, the O-D pairs with the heaviest loads will tend to gain the most access to the star because they will tend to have the longest queues. However, all users will be able to use the star if they need it. Thus, this strategy allows for full bandwidth sharing while ensuring that at heavy loads the star is largely reserved for the heaviest users.

We compare the three strategies in the following sections.

5.4 Delay Limits

An absolute bound on the delay achievable in the L1 would be very helpful with strategy comparisons. We calculate two bounds on the minimum queueing delay achievable.

Delay Bound 1

Note that the delay is bounded by that of a work-conserving network with the same number of servers and the same arrival rate as the L1. In such a network all system resources are shared and its delay is independent of the serving discipline. The delay of this network is the delay of an M/M/m network with the number of servers, m , equal to the total number of λ -router and transmissive star channels and arrival rate, λ , equal to the sum of the arrival rates of all the O-D pairs. The average queueing delay in an M/M/m network is given by W in (5.1) [BG92], where $\rho = \lambda/m\mu$, and, as throughout this thesis, μ is normalized to 1.

$$W = \frac{\rho P_0}{\lambda (1 - \rho)} \quad (5.1)$$

where

$$P_0 = \frac{\rho^m (m\rho)^m}{m!} \sum_{n=m}^{\infty} \rho^{n-m} \quad (5.2)$$

and

$$P_0 = \left[\sum_{n=0}^{m-1} \frac{(m\rho)^n}{n!} + \frac{(m\rho)^m}{m!(1-\rho)} \right]^{-1} \quad (5.3)$$

This delay bound is tight when r^λ is equal to zero. We expect it to be reasonably accurate when r^λ is small, and to be loose when r^λ is large.

Delay Bound 2

To get the second delay bound note that all O-D pairs in the network use at most $r^\lambda + Mr^\lambda$ servers. Define the arrival rate of calls on O-D pair i to be λ_i and the wait in queue for O-D pair i 's calls to be W_i . Then, from M/M/m queueing theory, the queueing delay W obeys the bound in (5.4),

$$W \leq \frac{1}{\sum_{i=0}^{M^2-1} \lambda_i} \sum_{i=0}^{M^2-1} \lambda_i W_i \quad (5.4)$$

where W_i is the delay of an M/M/m system with $\lambda = \lambda_i$ and $m = r^\lambda + Mr^\lambda$. The bound is exact at $r^\lambda = R$. We expect it to perform relatively well at large values of r^λ and to be very loose at small values of r^λ .

Queueing delay is lower bounded by both of these values. Thus, for any algorithm X, we can say that

$$\begin{aligned} & \max (\text{delay bound 1, delay bound 2}) \\ & \leq \text{delay using algorithm X} \end{aligned} \quad (5.5)$$

5.5 Delay Calculations

In certain cases the average queueing delay for the star sharing strategies can be calculated. In certain cases the delay for the bandwidth reservation system is easily calculable. Each O-D pair operates like an independent M/M/m system, where m

corresponds to the number of λ -router servers it has access to (r^λ) plus the number of servers it has reserved in the transmissive star (this number is not necessarily an integer). The overall average delay is the average of the individual session delays weighted by the session arrival rates.

In the general case where m does not turn out to be an integer, three options are available for approximating the delay. One method is to realize m by alternating between two integer values with weighted average equal to m . A rough approximation of the delay in this case can be calculated by using a weighted average of the delays calculated by using the largest integer less than m and the smallest integer greater than m . Because the delay of an $M/M/m$ system is nonlinear in m , this approximation may be poor.

The second method to approximate the delay is to note that the delay of an $M/M/m$ system with arrival rate λ is lower bounded by k times the delay of an $M/M/km$ system with an arrival rate $k\lambda$. This bound works well at low loads and small values of k , but as load and k grow it becomes very loose.

The third method is to lower bound the delay of the $M/M/m$ system by using an $M/M/c$ system where c is the smallest integer greater than m . Note that while the first method is a crude approximation, the second and third methods produce strict lower bounds. Thus, the delay of a bandwidth reservation system can be lower bounded by using the maximum of the delays produced by the second and third delay approximation methods.

The following example illustrates the use of the bandwidth reservation strategy delay approximation. Since m turns out to be an integer in this case, all three approximation methods yield the same result. Consider a heavy load system with $k_1 = 2$, $N_1 = 1$, $k_2 = 1$, $N_2 = 15$, and $R = 10$ at $r^\lambda = 8$ and $a = 7$ (thus $\rho \approx .9$). When the bandwidth is divided up

according to the bandwidth reservation strategy, the heavy user O-D pair is allowed 16 servers and the other O-D pairs get 8 servers each. The delay is the weighted average of an M/M/16 system with $\lambda = 14$ and fifteen M/M/8 systems with $\lambda = 7$. The resulting theoretical delay is .59, a value that has been confirmed by simulation.

The delay for the round-robin system is very difficult to calculate in general, but it satisfies an easily calculable upper bound. Each O-D pair has access to at least the λ -router channels and $1/M^2$ of the transmissive star's channels for a total of r^λ plus Mr^λ/M^2 channels. Thus, each O-D pair's delay is upper bounded by the delay of an M/M/(r^λ plus r^λ/M) system. The overall system delay is the average of the individual O-D pairs' average delays weighted by their arrival rates. As the load approaches capacity, the system delay approaches the bound.

The same scenario as above is used to illustrate the use of this bound in the heavy load situation. When $k_1 = 2$, $N_1 = 1$, $k_2 = 1$, $N_2 = 15$, $R = 10$, $r^\lambda = 8$ and $\alpha = 7$ the delay can be approximated by the weighted average of the delays of an M/M/($8 + 8/16$) system with $\lambda = 14$ and fifteen M/M/($8 + 8/16$) systems with $\lambda = 7$. The 8.5 servers available can not handle traffic with $\lambda = 14$ and the approximation produces infinite delay, as was again confirmed by simulation.

The delay of the longest queue strategy is very difficult to calculate, although at loads near capacity it should approach that of the bandwidth reservation strategy. We do not attempt to approximate this delay and instead rely on simulation for numerical results.

5.6 Simulation Results

Tables 5.1 - 5.3 show the simulated queueing delay for the round robin and longest queue strategies and calculated lower bounds on the delay for the bandwidth reservation strategy for a network with $k_1 = 2$, $N_1 = 1$, $k_2 = 1$, $N_2 = 15$ and $R = 10$. The maximum throughput value of r^λ for this network and traffic pattern is 8.

r^λ	Round Robin	BW Res	Longest Q
1 - 6	∞	∞	∞
7	∞	2.15 s	1.2 s
8	∞	.59 s	.46 s
9 - 10	∞	∞	∞

Table 5.1: Comparison of the Average Delays vs. r^λ Generated by the Three Strategies under Heavy Load when $k_1 = 2$, $N_1 = 1$, $k_2 = 1$, and $N_2 = 15$ at $a = 7$ ($\rho \approx .9$)

r^λ	Round Robin	BW Res	Longest Q
0-3	∞	∞	∞
4	∞	2.21	.72
5	.7	.54	.133
6	.08	.15	.05
7	.032	.052	.024
8	.025	.05	.0175
9	.035	.029	.027
10	.6	.6	.6

Table 5.2: Comparison of the Average Delays vs. r^λ Generated by the Three Strategies under Medium Load when $k_1 = 2$, $N_1 = 1$, $k_2 = 1$, and $N_2 = 15$ at $a = 5$ ($\rho \approx .6$)

r^{λ}	Round Robin	BW Res	Longest Q
0	∞	∞	∞
1	.035	.36	.03
2	.0018	.19	.0016
3	8×10^{-5}	.05	7×10^{-5}
4	8×10^{-6}	.012	2.5×10^{-6}
5	3×10^{-7}	.012	3×10^{-7}
6	5×10^{-9}	.003	5×10^{-9}
7	1.5×10^{-7}	7.27×10^{-4}	1.5×10^{-7}
8	1.5×10^{-6}	7.25×10^{-4}	1.5×10^{-6}
9	6×10^{-5}	1.94×10^{-4}	4.25×10^{-5}
10	9.3×10^{-4}	9.3×10^{-4}	9.3×10^{-4}

Table 5.3: Comparison of the Average Delays vs. r^{λ} Generated by the Three Strategies under Low Load when $k_1 = 2$, $N_1 = 1$, $k_2 = 1$, and $N_2 = 15$ at $a = 2.5$ ($\rho \approx .3$)

Figure 5.1 illustrates the behavior of the three strategies as a function of the network load. It shows that the longest queue strategy gives the lowest delay for all values of ρ . It also

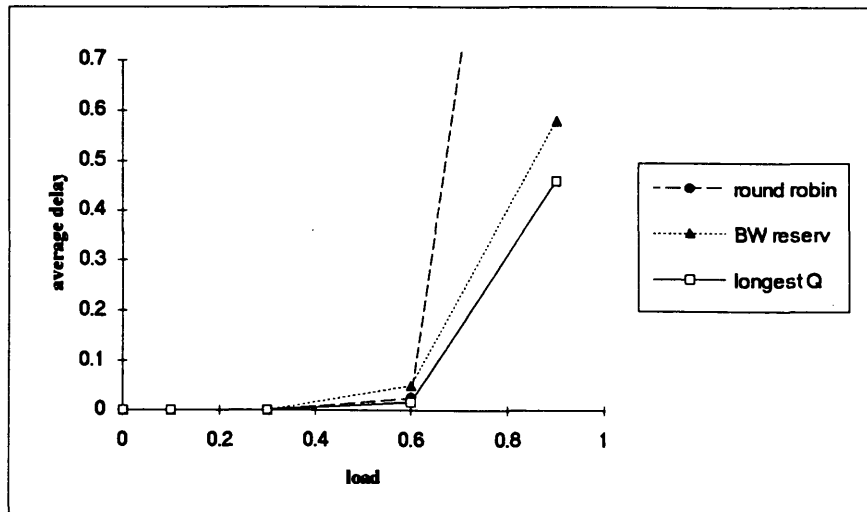


Figure 5.1: Average Delay Using Three Star Sharing Strategies at the maximum throughput value of r^{λ} for a Network with $k_1 = 2$, $N_1 = 1$, $k_2 = 1$, $N_2 = 15$ and $R = 10$

shows that at low values of ρ , the round robin strategy outperforms the bandwidth reservation strategy. At high values of ρ the bandwidth reservation strategy outperforms the round robin strategy.

5.7 Discussion

The simulation results show that the longest queue strategy always outperforms the bandwidth reservation and round robin strategy. In this section we compare the longest queue strategy to the other two and attempt to explain the performance differences. We also address an inefficiency inherent in the longest queue strategy.

5.7.1 Longest Queue vs. Bandwidth Reservation

At heavy loads, these two strategies perform very similarly. To see this, consider the very simplified scenario of $r^{\lambda} = 0$, $R = 3$, and two active sessions. Session one has $\lambda = 1$ and session two has $\lambda = 2$. The bandwidth reservation strategy will assign 1 server to session 1 and 2 servers to session 2. The longest queue strategy will assign the servers based on the length of the two queues. Each second, session one will have an average of one arrival, and session two will have an average of two arrivals. Assuming that the queues are equal at the beginning of this second (since the strategy acts to equalize the queues, most of the time they will be approximately equal), session two will be given the first available server, and each session will receive one of the next two available servers. On the average, three servers become available every second. Thus, session one will average one server per second and session two will average two servers per second -- the same result that was achieved with bandwidth reservation.

At lower loads, the bandwidth reservation strategy behaves the same as it does at heavy loads but the longest queue strategy allows for more dynamic bandwidth sharing. Under the longest queue strategy each session will get access to at least as much of the transmissive star's bandwidth as it would have using the bandwidth reservation strategy. Thus, the longest queue strategy should perform at least as well as the bandwidth reservation strategy at all loads, and should perform better at lower loads. The simulations support this hypothesis, and we conclude that use of the longest queue strategy produces smaller average queueing delay than use of the bandwidth reservation strategy.

5.7.2 Longest Queue vs. Round Robin

At low loads, the longest queue and round robin star sharing strategies produce approximately the same average delays. Both allow bandwidth sharing, and if there is any work that can be done in the star, both enable this work to be performed. The two strategies do differ, however, under heavier loads when $r^\lambda > 0$. To illustrate this case, consider two O-D pairs, one with a low load and one with a high load (Figure 5.2). The high load pair's queue will usually be non-empty. The low-load pair's queue will usually be empty, but due to statistical fluctuations, it will at times have a number of calls waiting for service. Assume the low load queue has a small number of messages waiting for connection (Step 1). Under the round robin strategy, one of them will eventually enter the star (Step 2). Because the queue is small and the load is low, there is a high probability that the queue will empty and some λ -router session will complete before the star session completes (Step 3). At this point, the low-load O-D pair is using bandwidth in the star though it could be sending a message via its λ -router. Star bandwidth is unnecessarily being used by the low-load O-D pair and the system is operating in an inefficient manner.

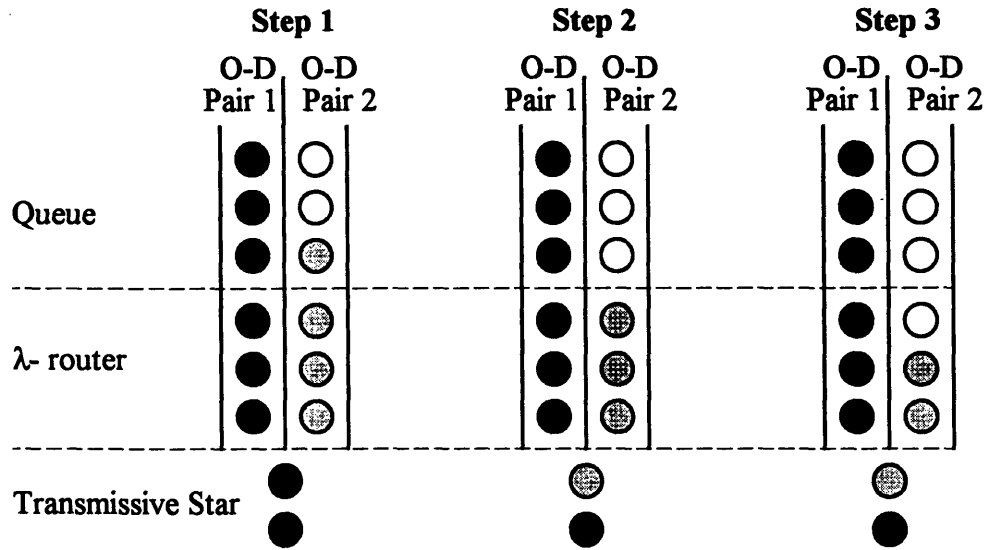


Figure 5.2: Round Robin Strategy

Note that if the longest queue strategy was being used, the low load session would rarely get service in the star. A high load O-D pair's queue is unlikely to empty before its messages in the transmissive star end, and therefore the probability of system inefficiency is small. In this situation the longest queue strategy is more efficient than the round robin strategy. The scenario illustrated above would happen with high frequency if a round robin strategy was being used. For this reason the longest queue strategy uses network resources more efficiently than the round robin strategy and should produce a shorter average delay. This result was confirmed by simulation.

5.8 Call Redirection and the Longest Queue Strategy

The longest queue strategy outperforms the other two strategies considered under all network loads, and it accomplishes two goals that an optimum strategy should accomplish. First, it allows for bandwidth sharing at all network loads and second, it ensures that all nodes will have finite queues if possible. However, the situation illustrated in Step 3 of Figure 5.2, which results in inefficiencies and thus longer waiting times, is still

possible with the longest queue strategy. The following modification to the strategy would eliminate this inefficient use of λ -router bandwidth.

1. Allocate resources in such a manner that all possible λ -router channels are used.
Send rest of traffic to the transmissive star, using the longest queue strategy.
2. Redirect calls to the λ -router at any time to ensure (1) is satisfied.

Call redirection will be invoked only when an O-D pair has a session in the star, one of its λ -router sessions terminates, and it has no sessions in the queue. This strategy allows for the redirection of a call from the transmissive star to an idle λ -router channel. We now argue that call redirection should not significantly decrease average delay over the longest queue strategy.

Under low loads, the transmissive star is usually not full and any allocation strategy that allows bandwidth sharing will perform similarly. Under heavy loads, using the longest queue strategy ensures that all sessions in the transmissive star belong to O-D pairs with long queues. Thus, if one of their λ -router channels becomes available before the star session finishes service, one of the queued sessions will take its place without any inefficiency. Because the sessions in the transmissive star belong to O-D pairs with long queues, it is very unlikely that a λ -router channel opens up before a star session finishes service for an O-D pair that has no queue, which is the only time call redirection would be invoked. Thus, under the longest queue strategy, it is very unlikely that call redirection would be necessary and the efficiency of this strategy approaches that of a strategy that allows for call redirection.

In a real system we would have to consider the fact that redirecting calls carries with it overhead costs. These costs depend on the system, but regardless of the system they further reduce the benefits that would be gained by call redirection. It is even possible that redirecting calls would add to the overall average delay as a result of the switching overhead. For this reason, it appears that the longest queue strategy is a very good strategy when the goal is the minimization of overall delay.

Chapter 6

Conclusion

In this thesis we addressed many problems relating to the efficient use of resources in the Level 1 of the AON. We developed a method for optimally allocating the bandwidth in the AON Level 1 between the λ -router and the transmissive star in order to maximize network throughput. The method is applicable to arbitrary traffic patterns. We have explored the performance of the L1 under Poisson arrivals and exponential service and have shown that when the network is configured for maximum throughput the queueing delay in the L1 approaches the minimum delay possible. Finally, we have considered several scheduling strategies for sharing the resources of the transmissive star and concluded that a longest queue strategy results in the most efficient use of the L1 resources.

Appendix A

Simulation

In this appendix we describe the simulation that was used to support the results of Chapters 4 and 5.

A.1 Simulation Tool

The simulation tool used in this thesis was Opnet, version 2.4c. Produced by MIL3, Inc.⁴, Opnet is a package designed to simulate communication networks. For further information on Opnet, consult the Opnet manuals.

A.2 Simulation Model

The simulation involves a network of four nodes (LOs) and thus sixteen O-D pairs. A total of ten FSRs are divided between the λ -router and the transmissive star. Each wavelength is modeled as an independent server. Each O-D pair has access to r^λ proprietary servers (wavelengths in the λ -router). In addition, M_r servers (transmissive star wavelengths) are shared by all the O-D pairs. Figure 4.1 in Chapter 4 illustrates the server model used.

⁴Mil3, Inc., The INTELSAT Building, 3400 International Dr. NW, Washington, DC 20008.

In all cases calls are served by the λ -router if possible. If the λ -router is busy calls are either queued or routed to the transmissive star. The scheduling of the transmissive star is determined by the star sharing strategy as determined in Chapter 5.

All call hold times are exponential with parameter one. Arrivals to the O-D pairs are independent Poisson processes. Each O-D pair has a specific arrival rate. This rate is set to yield different simulation scenarios.

The performance parameter measured is average queuing delay. Queuing delay is measured from the time a call is requested to the time it is serviced. The average is calculated over all messages in the network.

A.3 Validation Using Theoretical Results

In general, queuing delay in the simulation model can not be calculated theoretically. However, in special cases the simulation model can be treated like an M/M/m queue. In Table A.1, we compare the theoretical results to the simulation results for several of these special cases in order to ensure that the simulation model is correct.

R	r^λ	Arrival rate (each O-D pair)	Theoretical Average Delay	Measured Average Delay
10	10	5	.0072 s	.0072 s
10	0	2	.0015 s	.0015 s
10	5	20 (node 0) ~0 (all others)	.042 s	.043 s

Table A.1: Comparison of Simulated and Theoretical Results

The measured results agree with the theoretical results and therefore there is significant evidence that the simulation produces valid results.

A.4 Simulation Code

Figure A.1 illustrates the organization of the network model. The solid lines denote message paths and the icons denote nodes (or devices). Each node has a corresponding process model. Figure A.2 illustrates the organization of a λ -router node. Figure A.3 illustrates the organization of the transmissive star node. The node model attributes and the code for the process models follow the figures.

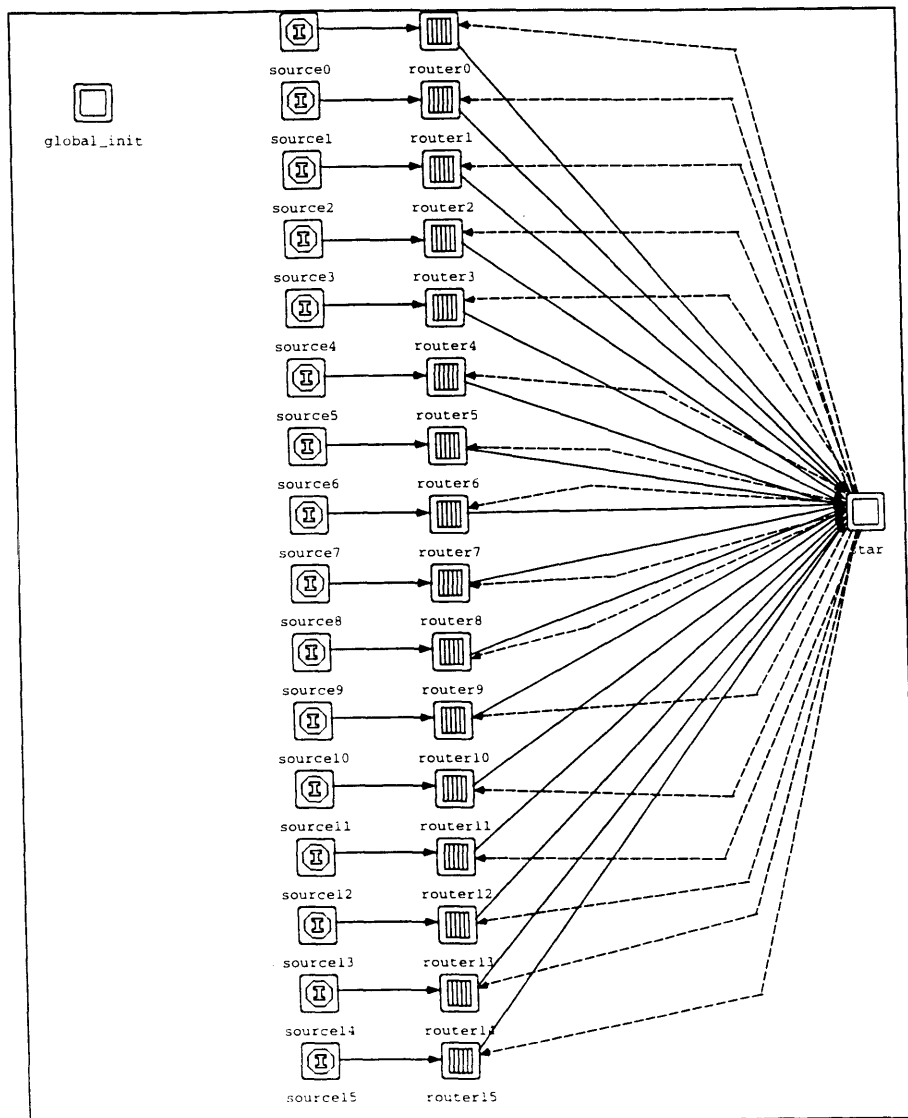


Figure A.1: Network Model

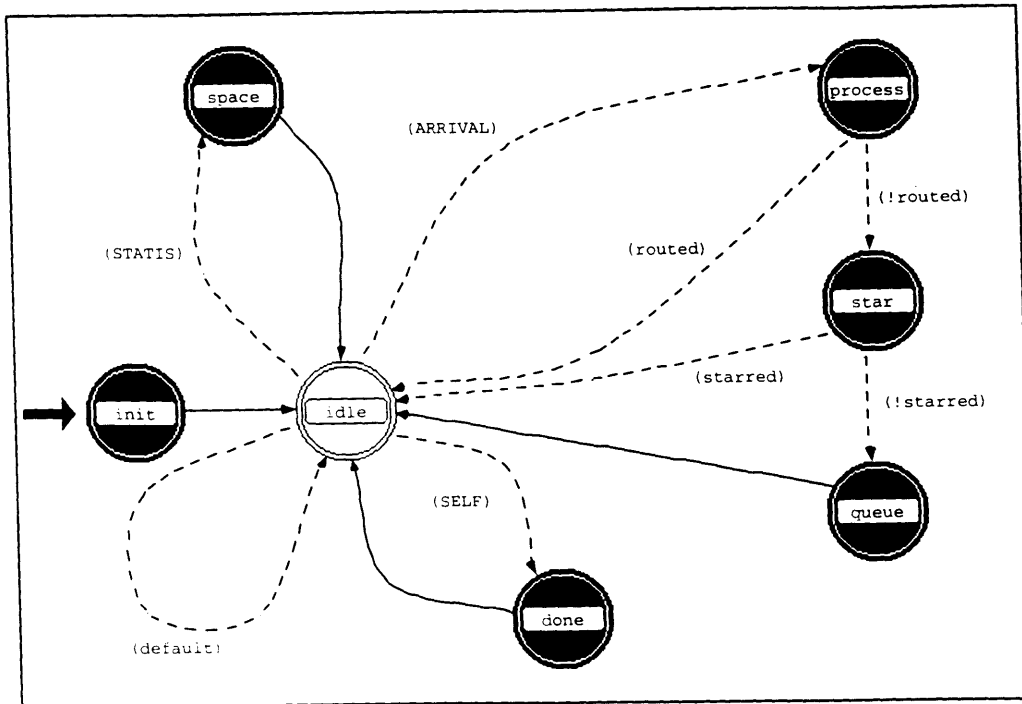


Figure A.2: λ -Router Node Process Model

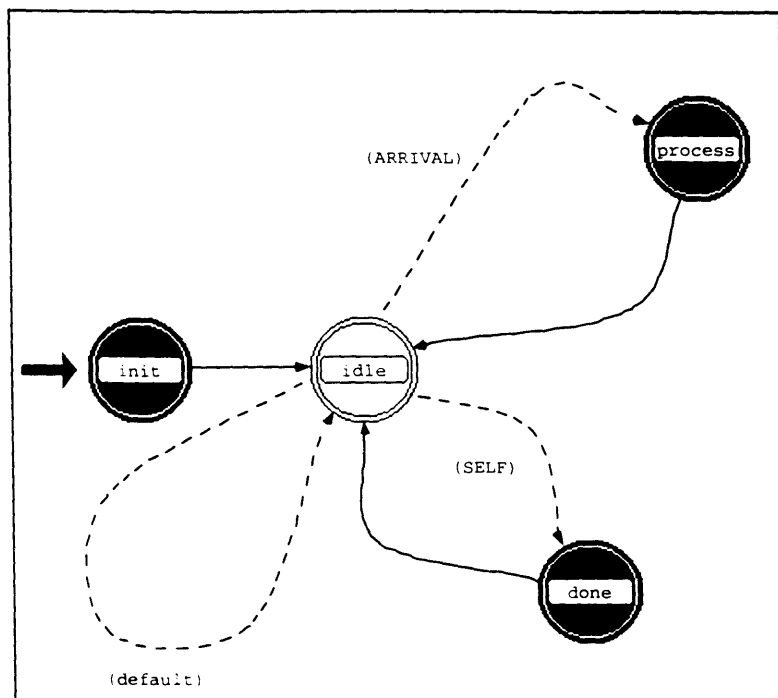


Figure A.3: Transmissive Star Process Model

A.4.1 Node Model Attributes

Node Model Attributes			
attribute	value	type	default value
R	promoted	integer	10
r_lambda	promoted	integer	5
packets (thousands)	promoted	integer	100

Ideal generator source0			
attribute	value	type	default value
name	source0	string	g
interarrival pdf	exponential	typed file	constant
interarrival args	promoted	string list	1.0
pk size pdf	constant	typed file	constant
pk size args	10	string list	1024,0
field (0) pdf	constant	typed file	constant
field (0) args	0.0	string list	0.0
packet format	NONE	typed file	NONE
start time	0.0 (sec.)	double	0.0 (sec.)
stop time	infinite (sec.)	double	infinite (sec.)
icon name	ideal_gen	icon	ideal_gen

packet stream source0 [0] -> router0 [0]			
attribute	value	type	default value
name	stm_14	string	stm
src stream	0	enumerated	0
dest stream	0	enumerated	0
intrpt method	scheduled	integer	scheduled
delay	0.0 (sec.)	double	0.0 (sec.)
color	RGB030	color	RGB030

queue router0			
attribute	value	type	default value
name	router0	string	p
process model	_router_new2	typed file	pc_fifo
subqueue count	1	integer	1
subqueue	(See below.)	object list	(See below.)
intrpt interval	disabled	toggle double	disabled
begsim intrpt	enabled	toggle	disabled
endsim intrpt	disabled	toggle	disabled
failure intrpts	disabled	enumerated	disabled
recovery intrpts	disabled	enumerated	disabled
priority	0	integer	0
super priority	disabled	toggle	disabled

subqueue router0.subqueue [0]			
attribute	value	type	default value
bit capacity	infinite (bits)	double	infinite (bits)
pk capacity	infinite (pkts)	double	infinite (pkts)

packet stream router0 [0] -> star [0]			
attribute	value	type	default value
name	stm_15	string	stm
src stream	0	enumerated	0
dest stream	0	enumerated	0
intrpt method	scheduled	integer	scheduled
delay	0.0 (sec.)	double	0.0 (sec.)
color	RGB030	color	RGB030

icon name			
id	queue	icon	queue
	0	integer	0

A.4.2 λ -Router Node Process Code

<pre> delay_handle = op_stat_global_req('Queueing Delay'); /* get own id number */ op_lms_obj_attr_get(own_id, id, &my_id); 20 </pre>		<pre> transition init -> idle attribute value type default value name tr_0 string tr condition string executive RGB333 color RGB333 drawing style spline toggle spline </pre>
<pre> /* forced state idle attribute value type default value name idle string st enter execs (empty) textlist (empty) exit execs (empty) textlist (empty) status unforced toggle unforced </pre>		<pre> transition idle -> done attribute value type default value name tr_1 string tr condition SELF string executive RGB333 color RGB333 drawing style spline toggle spline </pre>
<pre> transition idle -> process attribute value type default value name tr_13 string tr condition ARRIVAL string executive RGB333 color RGB333 drawing style spline toggle spline </pre>		<pre> transition idle -> space attribute value type default value name tr_14 string tr condition STATUS string executive RGB333 color RGB333 drawing style spline toggle spline </pre>
<pre> transition idle -> idle attribute value type default value name tr_16 string tr condition default string </pre>		<pre> transition idle -> idle attribute value type default value name tr_16 string tr condition default string </pre>

<pre> Header Block Define ARRIVAL (op_intrpt_type) == OPC_INTRPT_STRM) Define SELF (op_intrpt_type) == OPC_INTRPT_SELF) Define STATUS (op_intrpt_type) == OPC_INTRPT_STAT) Define M 4 5 extern int packet_count; extern double wait_time; extern int queues[16]; extern int stat_count; extern int term_count; </pre>	<pre> State Variable Block int vrouter_count; int vrouter_max; int v_lambda; int V; 5 Object own_id; Distribution* hold_dist; Handle delay_handle; int vny_id; int vfinal_count; </pre>
<pre> Temporary Variable Block int reused; int started; Packet* pcp; double hold_time; </pre>	<pre> forced state init attribute value type default value name init string st enter execs (See below.) textlist (See below.) exit execs (empty) textlist (empty) status forced toggle unforced </pre>
<pre> enter execs init /* set count to 0 */ router_count = 0; /* get R, lambda */ own_id = op_id_self(); parent = op_lms_getparent(own_id); op_lms_obj_attr_get(parent, 'R', &R); op_lms_obj_attr_get(parent, 'lambda', &lambda); 10 /* get termination count */ op_lms_obj_attr_getparent('packets (thousands)', &final_count); /* load hold distribution */ hold_dist = op_dist_load('exponential', 1.0, 0.0); 15 /* register delay statistic */ </pre>	<pre> </pre>

executive			
color	RGB333	string	RGB333
drawing style	spline	toggle	spline

forced state done	value	type	default value
name	done	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

```

enter execs done
/* just freed up a circuit */
/* decrement the router count, and if someone is waiting */
/* for service, let them use the router */
5
--router_count;
if (top_subq_empty(0))
    pkt = op_subq_pk_remove(0, OPC_OPOS_HEAD);
    questatim_id = op_subq_stat(0, OPC_OSTAT_PKSIZE);
/* record waiting time */
++packet_count;
wait_time = wait_time + op_sim_time() - op_pk_creation_time_get(pkt);
if (packet_count == 1000)
    {
/* write global statistic and reset count */
op_stat_global_write(delay_handle, wait_time/double(packet_count);
packet_count = 0;
wait_time = 0.0;
++term_count;
if (term_count == final_count)
    op_sim_end(".....");
    }
++router_count;
op_pk_destroy(pkt);
hold_time = op_dist_outcomethold_dist;
op_intrpt_schedule_self(op_sim_time() + hold_time, 1);
}
op_stat_local_write(0, double(router_count);
30

```

transition done -> idle	value	type	default value
name	tr_2	string	tr
condition		string	
executive	RGB333	string	RGB333
color	spline	color	spline
drawing style		toggle	

forced state process	value	type	default value
name	process	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

```

enter execs process
/* process an arriving user */
/* end in t_router if space permits */
5
pkt = op_pk_get(top_intrpt_strm);
if (router_count < t_lambda)
    {
++router_count;
op_stat_local_write(0, double(router_count);
/* record waiting time - same as before since no delay */
++packet_count;
if (packet_count == 1000)
    {
/* write global statistic and reset count */
op_stat_global_write(delay_handle, wait_time/double(packet_count);
packet_count = 0;
wait_time = 0.0;
++term_count;
if (term_count == final_count)
    op_sim_end(".....");
    }
op_pk_destroy(pkt);
hold_time = op_dist_outcomethold_dist;
op_intrpt_schedule_self(op_sim_time() + hold_time, 1);
routed = 1;
}
else
    routed = 0;

```

transition process -> idle	value	type	default value
name	tr_7	string	tr
condition	routed	string	
executive	RGB333	string	RGB333
color	spline	color	spline
drawing style		toggle	

transition process -> star	value	type	default value
name	tr_8	string	tr
condition	routed	string	
executive		string	

color	RGB333	color	RGB333
drawing style	spline	toggle	spline

forced state	queue	value	type	default value
attribute	queue	queue	string	st
name	(See below.)	(See below.)	textlist	(See below.)
enter execs	(empty)	(empty)	textlist	(empty)
exit execs	forced	forced	toggle	unforced
status				

```

ENTER EXEC QUEUE
/* since user fits neither in star nor in router, place */
/* user in waiting queue */
op_subq_pk_insert(0,pkptr.OPC_QPOS_TAIL);
quest(my_id)=op_subq_stat(0,OPC_QSTAT_PKSIZE);
5

```

transition	queue -> idle	value	type	default value
attribute	tr_11	tr_11	string	tr
name			string	
condition			string	
executive			string	
color	RGB333	RGB333	color	RGB333
drawing style	spline	spline	toggle	spline

forced state	space	value	type	default value
attribute	space	space	string	st
name	(See below.)	(See below.)	textlist	(See below.)
enter execs	(empty)	(empty)	textlist	(empty)
exit execs	forced	forced	toggle	unforced
status				

```

ENTER EXEC SPACE
/* check if anyone waiting */
/* if yes, check if star full (should never be here */
/* since we were just notified of a decrease in star */
/* use), if not full, send first person in queue to star */
/* The packet arrival is forced so that star_count may */
/* be incremented before the next person tries to send */
if (op_subq_empty(0))
{
pkptr = op_subq_pk_remove(0,OPC_QPOS_HEAD);
quest(my_id)=op_subq_stat(0,OPC_QSTAT_PKSIZE);
10
/* record waiting time */
++packet_count;
wait_time = wait_time + op_sim_timer() - op_creation_time_get(pkptr);
if (packet_count == 1000)
15

```

color	RGB333	color	RGB333
drawing style	spline	toggle	spline

forced state	star	value	type	default value
attribute	star	star	string	st
name	(See below.)	(See below.)	textlist	(See below.)
enter execs	(empty)	(empty)	textlist	(empty)
exit execs	forced	forced	toggle	unforced
status				

```

ENTER EXEC STAR
/* Check if star full. If not, send waiting user to star */
/* Send forced so that the next user knows the star */
/* is full */
5
if (star_count < M*(R - r_lambdai))
{
/* record waiting time - no delay */
++packet_count;
if (packet_count == 1000)
10
{
/* write global statistic and reset count */
op_star_global_writedelay_handle_wait_time(double(packet_count));
packet_count = 0;
wait_time = 0;
++term_count;
if (term_count == final_count)
15
{
op_star_send(.....);
}
/* send immediately to the star */
op_pk_send(pkptr(0));
++star_count;
started = 1;
}
else
20
started = 0;
25

```

transition	star -> idle	value	type	default value
attribute	tr_9	tr_9	string	tr
name	starred	starred	string	
condition			string	
executive			color	
color	RGB333	RGB333	color	RGB333
drawing style	spline	spline	toggle	spline

transition	star -> queue	value	type	default value
attribute	tr_10	tr_10	string	tr
name	istarred	istarred	string	
condition			string	
executive			string	

```

20 | /* write global statistic and reset count */
    | op_stat_global_write(delay_handle, wait_time/(double)packet_count);
    | packet_count = 0;
    | wait_time = 0.0;
    | ++term_count;
    | if (term_count == final_count)
    |     op_stat_end(" ", " ", " ");
25 | }
    |
    | /* send packet to star */
    | op_pk_send(pkt, 0);
    | ++star_count;
30 | }

```

transition	space	→	idle
attribute		value	default value
name		tr_15	tr
condition			string
executive			string
color		RGB333	RGB333
drawing style		spline	spline

A.4.3 Transmissive Star Process Code Using Longest Queue Strategy

enforced state	idle	value	type	default value
attribute				
name	idle		string	st
enter execs	(empty)		textlist	(empty)
exit execs	(empty)		textlist	(empty)
status	unforced		toggle	unforced

transition	idle -> done	value	type	default value
attribute				
name	tr_1		string	tr
condition	SELF		string	
executive			string	
color	RGB333		color	RGB333
drawing style	spline		toggle	spline

transition	idle -> process	value	type	default value
attribute				
name	tr_13		string	tr
condition	ARRIVAL		string	
executive			string	
color	RGB333		color	RGB333
drawing style	spline		toggle	spline

transition	idle -> idle	value	type	default value
attribute				
name	tr_14		string	tr
condition	default		string	
executive			string	
color	RGB333		color	RGB333
drawing style	spline		toggle	spline

forced state	done	value	type	default value
attribute				
name	done		string	st
enter execs	(See below.)		textlist	(See below.)
exit execs	(empty)		textlist	(empty)
status	forced		toggle	unforced


```

ENRPT_RESET done
/* Notify the next person in line to get service */
/* This is the node with the longest queue */
--star_count;
notify = -1;
masq = 0;
for (i = 0; i < 4*M; i++)
    if (queue[i] > masq)
        notify = i;
10

```


transition	init -> idle	value	type	default value
attribute				
name	tr_0		string	tr
condition			string	
executive			string	
color	RGB333		color	RGB333
drawing style	spline		toggle	spline

```

Header Block
/*define ARRIVAL (ep_inprt_type) == OPC_INTRPT_STRM)
/*define SELF (ep_inprt_type) == OPC_INTRPT_SELF && ep_inprt_code() == 1)
/*define NOTICE (ep_inprt_type) == OPC_INTRPT_SELF && ep_inprt_code() == 2)
5
extern int star_count;
extern int queue[16];

```

```

State Variable Block
int v_lambda;
int UC;
Objid town_id;
Distribution *hold_dist;
3
int v_star;

```

```

Temporary Variable Block
Packer *pkt;
double hold_time;
int i;
int j;
int masq;
int notify;

```

```

forced state init
attribute
name
value
init
(See below.)
string
st
enter execs
(See below.)
textlist
(exit execs)
(empty)
textlist
status
forced
toggle
unforced

```

```

ENRPT_RESET init
/* get R_r_lambda */
own_id = ep_id_self;
ep_line_obj_ptr_get(ep_param(own_id), "r", AR);
ep_line_obj_ptr_get(ep_param(own_id), "r_lambda", &r_lambda);
3
/* load hold distribution */
hold_dist = ep_dist_load(exponential, 1.0, 0);

```

```

transition init -> idle
attribute
name
value
tr_0
string
condition
string
executive
string
color
RGB333
drawing style
spline
toggle

```

```

15      maxq = queue();
      }
      if (notify >= 0)
        op_sia_local_write(notify, 1.0);

```

transition	done -> idle
attribute	value
name	tr_42
condition	
executive	
color	RGB333
drawing style	spline
	tr
	RGB333
	spline

forced state	process
attribute	value
name	process
enter execs	(See below.)
exit execs	(empty)
status	forced
	forced
	st
	(See below.)
	(empty)
	unforced

```

enter execs process
/* receive request for service, service person */
/* for an exponential amount of time */
/* schedule an interrupt for the time of completion */
/* notify everyone simultaneously of increase in number */
/* currently being served */
5
pkpr = op_pk_get(op_intrpt_stm1());
op_pk_destroy(pkpr);
hold_time = op_dist_outcome(hold_dist);
10 op_intrpt_schedule_self(op_sia_time) * hold_time.1;

```

transition	process -> idle
attribute	value
name	tr_7
condition	
executive	
color	RGB333
drawing style	spline
	tr
	RGB333
	spline

A.4.4 Transmissive Star Process Code Using Round Robin Strategy

<pre> Header Block define ARRIVAL (op, intrpt, type) == OPC_INTRPT_STRM define SELF (op, intrpt, type) == OPC_INTRPT_SELF && op_intrpt_code() == 1 define NOTICE (op, intrpt, type) == OPC_INTRPT_SELF && op_intrpt_code() == 2 define M4 extern int star_count; extern int queues[16]; </pre>	<pre> unforced state idle attribute name value type default value enter execs idle string st exit execs (empty) textlist (empty) status (empty) textlist (empty) unforced unforced toggle unforced </pre>	<pre> unforced state idle attribute name value type default value enter execs idle string st exit execs (empty) textlist (empty) status (empty) textlist (empty) unforced unforced toggle unforced </pre>
<pre> State Variable Block int V_lambda; int U; Objid town_id; Distribution* hold_dist; int V_star; int last_notified; int notify; </pre>	<pre> transition idle -> done attribute name value type default value tr_1 tr string tr SELF SELF string tr RGB333 RGB333 color RGB333 spline spline toggle spline </pre>	<pre> transition idle -> process attribute name value type default value tr_13 ARRIVAL string tr RGB333 RGB333 color RGB333 spline spline toggle spline </pre>
<pre> Temporary Variable Block Packer* p; double hold_time; int i; int j; </pre>	<pre> forced state idle attribute name value type default value tr_14 tr string tr default default string tr RGB333 RGB333 color RGB333 spline spline toggle spline </pre>	<pre> forced state idle -> idle attribute name value type default value tr_14 tr string tr default default string tr RGB333 RGB333 color RGB333 spline spline toggle spline </pre>
<pre> enter execs init /* get R, r, lambda */ own_id = op_id_self(); op_ina_obj_ptr_get(op_topo_parent(own_id), "r", &R); op_ina_obj_ptr_get(op_topo_parent(own_id), "r_lambda", &r_lambda); /* load hold distribution */ hold_dist = op_dist_load("exponential", 1.0, 0.0); /* set up round-robin notification */ last_notified = 0; </pre>	<pre> forced state done attribute name value type default value done done string done (See below.) textlist (See below.) (See below.) textlist (See below.) forced forced toggle unforced </pre>	<pre> forced state done attribute name value type default value done done string done (See below.) textlist (See below.) (See below.) textlist (See below.) forced forced toggle unforced </pre>
<pre> transition init -> idle attribute name value type default value tr_0 tr string tr RGB333 RGB333 color RGB333 spline spline toggle spline </pre>	<pre> enter execs done attribute name value type default value done done string done (See below.) textlist (See below.) (See below.) textlist (See below.) forced forced toggle unforced </pre>	<pre> enter execs done attribute name value type default value done done string done (See below.) textlist (See below.) (See below.) textlist (See below.) forced forced toggle unforced </pre>
<pre> 5 10 </pre>	<pre> 5 10 </pre>	<pre> 5 10 </pre>


```

notify = -1;
}
--notify;
}
15

```

transition done -> idle			
attribute	value	type	default value
name	tr_42	string	tr
condition		string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

forced state process			
attribute	value	type	default value
name	process	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

```

enter execs process
/* receive request for service, service person */
/* for an exponential amount of time */
/* schedule an interrupt for the time of completion */
/* notify everyone simultaneously of increase in number */
/* currently being served */
5
pkcr = op_pk_get(op_intrpt_strm1);
op_pk_destroy(pkcr);
hold_time = op_diar_outcome(hold_dist);
10 op_intrpt_schedule_self(op_sma_timer, hold_time, 1);

```

transition process -> idle			
attribute	value	type	default value
name	tr_7	string	tr
condition		string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

A.4.5 Variable Initiation Node Process Code

Header Block			
	int packet_count;		
	double wait_time;		
	int queues[5];		
	int star_count;		
	int term_count;		
5			
Temporary Variable Block			
	int j;		
unforced state init			
attribute	value	type	default value
name	init	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced
enter execs init			
	/* initialize global variables */		
	packet_count = 0;		
	wait_time = 0.0;		
	for (j=0; j++ <= 5)		
	queues[j] = 0;		
	star_count = 0;		
	term_count = 0;		
5			

Bibliography

- [A+93] Alexander, Steven B., et al. A Precompetitive Consortium on Wide-Band All-Optical Networks. *Journal of Lightwave Technology*, 11(5/6):714-735, May/June 1993.
- [Aca93] Acampora, Anthony S. Intelligent optical networks: Research, education, and industrial programs at the center for telecommunications research. *Proceedings of the IEEE*, 81(1)111-131, January 1993.
- [Ale92] Alexander, Steven B. Minutes of the DARPA-2 test-bed meeting of November 19, 1992, Cambridge, MA.
- [Bar93] Barry, Richard A. *Wavelength Routing for All-Optical Networks*. Ph.D. thesis, Department of Electrical Engineering and Computer Science, MIT. 1993.
- [BG92] Bertsekas, Dimitri and Gallager, Robert. *Data Networks*. Prentice Hall, New Jersey, 1992.
- [Gre93] Green, Paul E. Jr. *Fiber Optic Networks*. Prentice Hall, New Jersey, 1993.
- [Fin92] Finn, Steven G. *Level 2 Architecture Issues*. MIT, May 26, 1992.
- [Wil56] Wilkinson, R. I. Theories for Toll Traffic Engineering in the U.S.A. *Bell Systems Technical Journal*, 35:421-514, March 1956.
- [Wol89] Wolff, Ronald W. *Stochastic Modeling and the Theory of Queues*. Prentice Hall, New Jersey, 1989.

