ABSTRACT

MARTIN WILLIAM McKINNON, III. Performance Analysis of a Class of Photonic Interconnection Architectures. (Under the direction of Professor Harry G. Perros and Professor George N. Rouskas.)

In this work, we evaluate the performance of a series of related photonic switches which operate under schedules that mask the transceiver tuning latency. The performance of these devices is expressed in terms of the occupancy probabilities, loss, and delay. We develop queueing-based decompositions algorithm to obtain the queue-length distribution at the input and output ports of the switches. The analysis is carried out assuming source models which capture the notions of burstiness, correlation, and non-uniform destination probabilities. The results presented indicate that there exists a complex interaction among the various system components (e.g., the load balancing and scheduling algorithms) and the various system parameters (e.g., the number of available channels and the buffer capacity), and that the overall performance of such devices may not be predictable without an accurate analysis of both the traffic patterns and the switch itself.

PERFORMANCE ANALYSIS OF A CLASS OF PHOTONIC INTERCONNECTION ARCHITECTURES

by

Martin W. McKinnon, III

A thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Computer Engineering

Raleigh

1997

APPROVED BY:

Co-Chair of Advisory Committee

Co-Chair of Advisory Committee

To myself,

Bill McKinnon,

the only person worthy of my company.

BIOGRAPHY

I am what I am.

ACKNOWLEDGEMENTS

Thanks, everybody.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

One of the issues in evolving today's networks is that of developing device architectures that can effectively switch data at very high data rates (currently, data rates on the order of a few tens of Gigabits per second per port are envisioned). Over the last decade, a great deal of research has been devoted to the design of fast cell switches and high speed routers suitable to a broadband integrated services environment; surveys of some of these architectures may be found in [2, 66]. Mainstream research and development activities in the area of broadband switching are focused exclusively on electronics-based technologies which have attained a high level of maturity. On the other hand, the deployment of optics is limited to mere point-to-point transmission where the technology has proven successful in a short time span.

Given the continued rapid progress in lightwave technology (including the demonstration of fast tunable transceivers [32, 62], the development of erbium-doped fiber amplifiers [50], and guided-wave optical switching [65]), and the anticipated total dominance of optical fiber in the wired network, the issue of deeper penetration of optics naturally arises. Given the potential of optical solutions to cell switching and derivative solutions, the possibility of employing photonics to implement switching functions hitherto reserved for electronics is currently being explored (see [38] and references thereof). However, there remain at least two major technical challenges to be overcome before one can contemplate the design of *all-optical* switches. First, there is the difficulty of "controlling light by light", and secondly, the technologies for implementing buffering in the optical domain are not yet mature enough. Consequently, the most likely scenarios for near-term photonic cell switching will involve an optical switching fabric with electronic control and buffering.

It has long been recognized that Wavelength Division Multiplexing (WDM) will be instrumental in bridging the gap between the speed of electronics and the virtually unlimited bandwidth available within the optical medium. The wavelength domain adds a significant new degree of freedom to network design, allowing new network concepts to be developed. With a few exceptions (e.g., [45, 59, 60]), however, most broadcast WDM architectures that have appeared in the literature require a large number of wavelengths and/or very fast tunable transceivers [3, 4, 10, 18, 22, 29, 35, 44]. Furthermore, the performance analysis of these architectures has been typically carried out assuming uniform traffic and memoryless arrival processes (see most of the above references, as well as [8, 13, 14, 15, 17, 19, 25, 26, 27, 34, 39, 46, 47, 55, 67]). However, it has been shown that, in order to study correctly the performance of a switch, one needs to use traffic models that capture the notion of burstiness and correlation, and which permit non-uniform output port destinations [53, 56].

In this paper, a series of similar device architectures are analyzed using correlated traffic patterns. The performance of these devices is expressed in terms of the occupancy probabilities, loss, and, in one significant case, delay. In doing so, we develop a queueing-based decomposition algorithm to obtain the queue-length distribution at the input and output ports of the switches. The analysis is carried out assuming relevent source models which capture the notions of burstiness, correlation, and non-uniform destination probabilities.

Chapter 2

Photonic Switch Model Overview

Today, photonic technology primarily takes advantage of the low-loss characteristics of fiber optic media. Single channel digital signals transmitted over fiber optic cables show extremely low error rates. As bandwidth requirements increase beyond the capabilities of electronics, the multiplexing of channels which can be addressed by either electronics or a new technology will be necessary. “Wavelength Division Multiplexing” (WDM) appears to be a plausible technology for addressing this issue. With the realization of reliable tunable transmitters and diffraction gradients which can transmit and receive over specific frequencies, an increase in the amount of data which can be sustained over the medium by an order of magnitude becomes realistic. Current projections call for the ability to use the low-loss region of the lightwave spectrum to accommodate as many as hundreds of independent, non-interfering transmissions [23].

One significant issue associated with migrating the current information infrastructure to an all-optical switching system is the saturation of comparably low-speed electronic devices within the computing community. By simply attaching electro-optical interfaces to today's devices, the result will be one of two scenarios:

- individually allocated frequencies will be extremely under-utilized, or
- highly efficient media-access protocols implemented in the optical domain will regulate access to shared frequencies between sets of users.

Multiplexing electronic lines to near peak utilizations before translation to the optical domain will lessen the problem, but the incongruity between speeds (by orders of magnitude) will still demand a more satisfactory solution as individual bandwidth requirements increase.

Several switching paradigms have arisen as a result of research aimed at tapping this potential bandwidth, including

- "Single Hop Broadcast-and-Select",
- "Multiple Hop ("Multi-Hop") Broadcast-and-Select", and
- "Wavelength Routing" systems.

"Broadcast-and-select" systems are based on a passive star coupler receiving transmissions on several unique frequencies from a number of input ports and broadcasting the signals to all output ports. The output ports then use a diffraction grating or similar equipment to filter an individual signal (i.e., frequency). "Wavelength routing" systems relay groups of transmissions through intermediate nodes, assigning output wavelengths for each transmission based on route characteristics.

Single hop broadcast-and-select systems ("Single hop systems") are characterized by transmissions sent directly from the originating to the destination nodes. While these systems are relatively simple and easy to implement, the reliance on higher layer protocols (e.g, the ULPHA architecture of [64]) and the contention resolution algorithms for shared channels are significant issues. Applications for which this type of system is well suited include distributed processor/shared memory configurations, locally distributed database operations, local data transfers, and local video transmissions (see [33]).

Multi-hop broadcast-and-select systems (i.e., "multi-hop systems") are similar to single hop systems. In this type of system, however, every device attached to the switch can receive and transmit information on a sub-set of frequencies supported by the switch. A cell then may have to be sent to an attached device and returned to the switch repeatedly in order for it to be encoded on the appropriate frequency before continuing towards its destination. This type of system does have the distrinct advantages of scalability and modularity, but the devices for converting signals between frequencies are not currently financially feasible; therefore, this operation requires the incoming data to be converted and stored in a comparatively slow electronic form and then re-converted to an optical format on another frequency. Systems such as "Shuffle-net" ([1], [30], [49]) plan for extremely fast and low overhead dedicated network interface hardware components to perform these operations. Another disadvantage in these systems is the design of a media access protocol which can support the addition of new nodes to the network which may only be able to access a specific set of frequencies; adding the new node and then propagating its addition into the routing structure of the adjacent nodes is a significant issue.

The main switching component within "Wavelength routing" systems does not receive only individual signals per input port, but receives a full-spectrum WDM input signal and selectively transmits the individual frequencies from each of the input lines to only the appropriate output lines. Wavelength changing functionality may be included in this type of architecture, but it is not necessary and may be avoided due to currently limited capabilities and performance and requirements which may only today be fulfilled in the electronic domain. The Lightning architecture and prototype [20] is one architecture which uses this type of switch in a hierarchical and scalable wide-area network.

2.1 Characteristics of Single-Hop Photonic Switches

The key component of most single-hop photonic switching systems is a passive star coupler (see Figure 2.1). The star coupler consumes virtually no power in the retransmission of input signals. Signals are transmitted to the coupler, merged, and then all frequencies are broadcast to all output ports (at a reduced power level). The devices attached to the output ports then filter the signals of interest from the aggregate signal. The transmitters and receivers, generally located at intelligent servers or terminals, may be either tunable or fixed frequency devices. Fixed frequency laser transmitters and receivers have become reasonably inexpensive with the increased popularity of home electronics (e.g., compact disc and video disc players). Greene shows in [32] (page 149) active tunable filters (i.e., receivers) available with tuning times as low as 0.1 μsec, the ability to distinguish between at least 70 frequencies within a 200 nm frequency spectrum, and no more than 5 dB attenuation within the transmitted signal's strength. Greater numbers of frequencies and decreased signal attenuation may be achievable, but must be traded off against

- decreases in tuning time;
- higher tolerances for frequency drift due to temperature intolerance; and/or,
- increases in size, temperature, unreliability, & cost.

Tunable lasers (i.e., transmitters) have similar shortcomings. The development of lasers which can quickly tune between frequencies within a relatively wide band of the

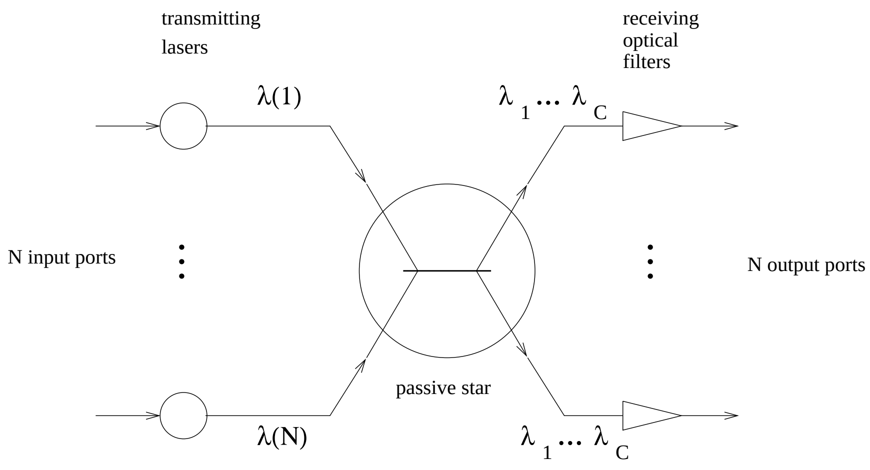

Figure 2.1: Abstract view of a passive star based switch

frequency spectrum is one of the main problems of photonic switches. Lasers currently generally operate reliably over a 35 nm band of the usable 240 nm spectrum, but higher numbers of frequencies and a broader usable spectrum generally results in lower reliability (with respect to temperature induced frequency drift), higher power consumption, and greater signal attenuation (page 204-205 of [32]).

Photonic switching systems are generally categorized by their tunability characteristics. For instance, a switch which uses tunable transmitters and fixed frequency receivers is referred to as a "TTFR" switch; similarly, a switch with fixed frequency transmitters and tunable receivers is termed "FTTR". Likewise, a switch in which both the input and output sides employ tunable components is a "TTTR" switch.

Most single hop systems currently under investigation must take advantage of either tunable transmitters or receivers, though many avoid using both due to prohibitive costs and the complexity of designing protocols which can (effectively) take advantage of the potential flexibility. Systems which would take advantage of both tunable transmitters and tunable receivers, however, should theoretically require fewer frequencies to deliver similar throughput measures, relaxing the tunability specifications for the components and possibly resulting in lower manufacturing costs.

The characteristics of systems with tunable transmitters differ significantly from those using tunable receivers. Tunable transmitters are more adept for point-to-point connections since the transmitter may simply tune itself to the receiver's frequency for communication. This type of system is ideal for transmissions which have significant constant bit rate (CBR) requirements (e.g., uncompressed non-interactive audio and video). Conversely, systems which use tunable receivers may be easily adapted to accommodate multicast requirements; these systems simply require that all recipients tune to the transmitter's frequency before transmission. The disadvantage to systems which use tunable receivers is that a parallel control network is necessary to inform the receiver to what frequency to tune.

Interest in the single hop photonic switch architecture discussed above arises for several reasons:

- it is highly modular, allowing the switch to grow relatively easily by adding ports and wavelengths;
- it is scalable, since the number of wavelengths need not be equal to the number of ports, and since the data rate within the switch needs only be $\frac{N}{C}$ times the rate of the input/output links;
- it provides end-to-end optical paths;
- its hardware requirements, in terms of the number of transceivers per port, is minimum;
- it can be reconfigured [9] to adapt to changing traffic patterns or to overcome failures of ports or transceivers; and,
- it does not require extremely fast tunable transmitters (as explained below), and thus can be built using *currently available* tunable optical devices.

The switch models which will be discussed in this work will be based on this architecture; specific characteristics (e.g., buffering paradigms, etc.) will be discussed as they become pertinent.

2.2 Transmission Schedules

One of the potentially difficult issues that arises in a WDM environment, is that of coordinating the various transmitters/receivers. Some form of coordination is necessary because (a) a transmitter and a receiver must both be tuned to the same channel for the duration of a cell's transmission, and (b) a simultaneous transmission by one or more input ports on the same channel will result in a *collision*. The issue of coordination is further complicated by the fact that tunable transceivers need a non-negligible amount of time to switch between wavelengths. For the Gigabit per second rates envisioned here, and for 53-byte ATM cells, the tuning latency of state-of-the-art tunable lasers or filters can be as long as several times the size of a service slot [32]. Consequently, approaches that require each tunable transmitter to send a single cell and then switch to a new wavelength, will suffer a high tuning overhead and will result in a very low throughput.

In a recent paper [60], it was shown that careful scheduling can mask the effects of arbitrarily long tuning latencies, making it possible to build high-throughput photonic ATM switches using *currently available* lightwave technology. The key idea is to have each tunable transmitter send a *block* of cells on a wavelength before switching to another one. The main result of [60] was a set of new algorithms for constructing near-optimal (and, under certain conditions, optimal) schedules for transmitting a set of traffic demands $\{a_{ic}\}$. Quantity a_{ic} represents the number of cells to be transmitted by input port i onto channel λ_c per frame. The schedules are such that no collisions occur. Furthermore, they are easy to implement in a high speed environment, since the order in which the various input ports transmit is the same for all channels [60].

Quantity a_{ic}, $i = 1, \cdots, N$, $c = 1, \cdots, C$, can be seen as the number of service slots per frame allocated to input port i, so that the port can satisfy the required quality of service of its incoming traffic intended for wavelength λ_c. By fixing a_{ic}, a certain amount of the bandwidth of wavelength λ_c is indirectly allocated to port i. This bandwidth could be equal to the effective bandwidth of the total traffic carried by input port i on wavelength λ_c. In general, the estimation of the quantities a_{ic}, $i = 1, \cdots, N$, $c = 1, \cdots, C$, is part of the call admission algorithm, and it is beyond the scope of this paper. Notice that as the traffic varies, a_{ic} may vary as well. In this paper, it is assumed that quantities a_{ic} are fixed, since this variation will more likely take place over larger scales in time.

It is assumed that transmissions by the input ports onto wavelength λ_c follow a schedule as shown in Figure 2.2. This schedule repeats over time. Each frame of the schedule consists of M arrival slots. Within each frame, input port i is assigned a_{ic} *contiguous* service slots for transmitting cells on channel λ_c. These a_{ic} slots are followed by a *gap* of $g_{ic} \geq 0$ slots during which no port can transmit on λ_c. This gap may be necessary to ensure that input port $i + 1$ has sufficient time to tune from wavelength λ_{c-1} to λ_c before it starts

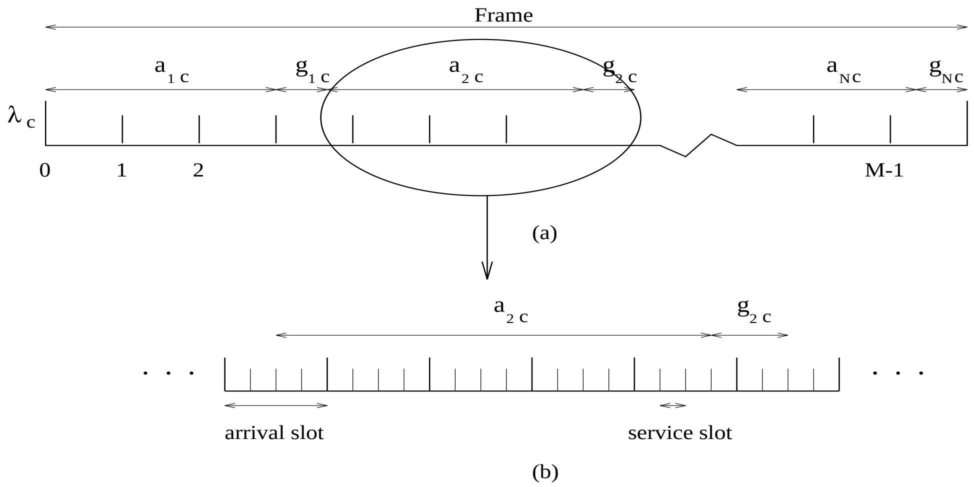

Figure 2.2: (a) Schedule for channel λ_c, and (b) detail corresponding to input port 2

transmission. The algorithms in [60] are such that the number of slots in most of the gaps is equal to either zero or a small integer. Thus, the length of the schedule is very close to the lower bound $\max_i\{\sum_{c=1}^{C} a_{ic}\}$. Note that in Figure 2.2 it is assumed that an arrival slot is an integer multiple of service slots. This may not be true in general, and it is not a necessary assumption for our model. Observe also that, although the frame begins and ends on *arrival* slot boundaries, the beginning or end of transmissions by a port does not necessarily coincide with the beginning or end of an *arrival* slot (although it is, obviously, synchronized with *service* slots).

Chapter 3

Previous Work in Photonic Switch Architectures

3.1 Photonic Knockout Architecture

This architecture is based on the Knockout Principle and this architecture's electronic implementation, the Knockout Switch ([40], [68]). In any slotted switching environment, fixed length data units (i.e., cells) may arrive from more than one input destined for the same output. In a typical slotted switching system, a potential worst-case situation may exist in which one cell may be received from every input and all are destined for the same output, resulting in extreme waiting times. An assumption and related approximation may be made, though, that arrivals from independent sources are independent and, therefore, the probability of receiving more than m cells all destined for the same output during a slot is negligible. Taking advantage of this assumption, the output lines of the switch may be "grouped" such that, if more than a fixed maximum number of cells arrives during a given slot bound for the same output, the excessive cells are "knocked out" of the switch and lost.

The Photonic Knockout Architecture ([22]) operates under a slotted timing scheme (appropriate to fixed length data packets, or "cells"). Transmission is accommodated in two stages: during each slot, the cells from the N input ports are transmitted by fixed frequency laser transmitters through a passive star coupler and buffered. Each of the N input ports are assigned to one of the "Output Packet Switch Modules". Based on the destinations of the buffered cells, the tunable receivers at the interface with the "Output Packet Switch

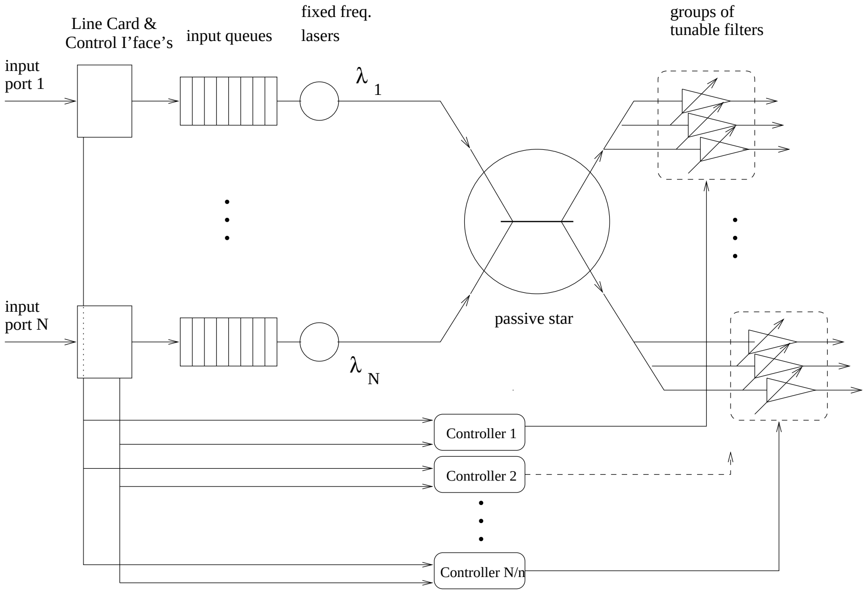

Figure 3.1: Photonic Knockout Architecture

Modules" are adjusted to the appropriate frequency for reception of a subset of the received cells with the remainder of the cells in each group "knocked out" of the switch. On the output side of the switch, one cell which is buffered at each of the output lines is be transmitted during each slot time, if available. Currently, the technology for performing this buffering efficiently (from both the technical and financial aspects) within the optical domain does not exist; as such, data translation via optoelectronic converters is a necessary part of the interface between the star coupler and the "Output Packet Switch Modules".

Based on the Knockout Principle and given assumptions of uniform random traffic on all of the input lines, ensuring low loss probabilities under heavy input loads with reasonably small values for m (defined above) is not a difficult task [42]. It has been shown that under as much as 90% uniform random loading, m need only be 8 cells (implemented by designing 8 input lines per Output Packet Switch Module) to ensure a loss probability of 10^{-6}; 10^{-10} can be assured with 12 cells ([23], [68]). These performance measures, however, may be improved by allowing for some number of cells which are "knocked out" to be retransmitted through the switch; In [22] it is shown that, for an $N \times 4N$ switch with $\frac{N}{2}$ possible retransmissions of blocked cells, m may be reduced to 4 input lines per Output Packet Switch Module in order to guarantee a loss probability of 10^{-6}. This architecture does, however, increase the number of fixed frequency lasers by 50% and tunable receivers by 12.5% ($\frac{N}{2}$ in both cases).

One advantage of this architecture in its electronic form is its modular (scalable) nature; stages of "Cell Distribution Networks" may be chained successively in order to increase the size of the switch. Therefore, the switch could effectively and efficiently "grow" to meet the demands of rising numbers of connections. In the optical domain, the star coupler can transmit as many frequencies as is necessary and physically feasible. Since this architecture uses fixed transmitters and tunable receivers, allocating new input lines to unique frequencies which are simply propagated by the coupler is not an issue, providing that the tunability characteristics of the receivers are not violated. Further, the designation of these new input lines and their corresponding frequencies and the communication of this information to the controllers for the tunable receivers is a minimal processing requirement.

As was mentioned previously, systems using tunable receivers must use some type of control mechanism in order to synchronize the laser transmitters and receivers. Tuning the receivers to access the correct frequency before the cell is transmitted through the Passive Star Coupler can be accommodated in any of several ways. One technique would involve buffering the cells for a fixed period of time (if necessary) while pipeline electronic processing of the headers of the incoming cells is performed. Buffering may or may not be necessary so long as the processing is completed in time for this control mechanism to notify the transmission fabric (i.e., the star coupler) which cells to transmit [68]. It should be noted that this task's demands, in conjunction with the transmission speed, will essentially dictate the required processor speed. Tuning this control mechanism for optimal performance in a given situation, however, may hamper the desirable scalability aspects of this architecture (i.e., once the controlling processor or processors are fully utilized, it will be a significant task to add additional input lines and frequencies). Sacrifices such as enqueueing input cells and transmitting cells in batches will degrade the desirable performance characteristics of this architecture. A second option would involve buffering groups of cells bound for the same destination and transmitting them as such. This option, however, would entail buffering the input lines which would again risk the performance characteristics of this paradigm. Finally, since this design does not require input buffering (all buffering is performed on output lines), ideal buffering delay can be attained. Karol, et al. show in [41] the advantages of output queueing as opposed to input queueing in discrete-time switching systems; output queueing systems saturate at significantly higher utilizations (potentially as the utilization of the switch becomes 100%) than input queueing systems (as the utilization of the switch

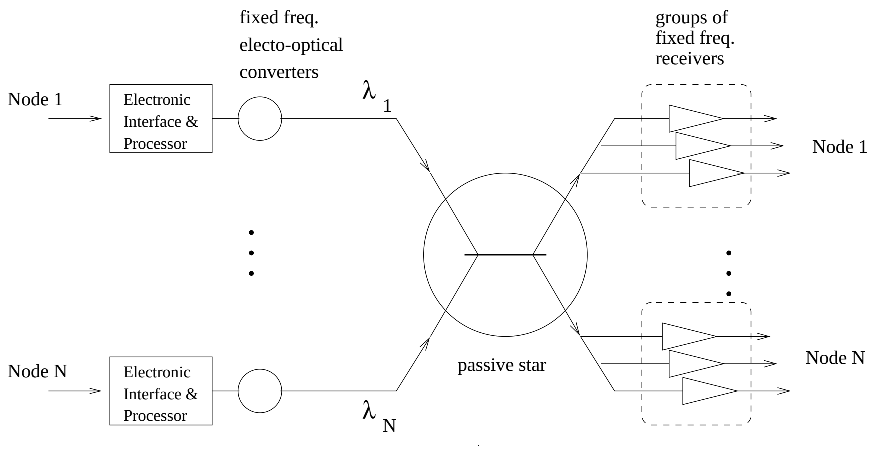

Figure 3.2: LAMBDANET Architecture

becomes 58.6%), as the switch grows infinitely large. Also, "head of line" ("HOL") buffering effects are not present in this architecture due to the output buffering strategy.

3.2 LAMBDANET Architecture

Bellcore's LAMBDANET Architecture and prototype is oriented toward a set of communicating nodes which process and maintain several inter-dependent streams of data. Each of the N nodes are allocated a single fixed unique frequency for transmission through the switch. Knowing the routing parameters, the receivers are equipped with as many as N fixed receivers which are accessed at the receiver's station as necessary[1]. Since LAMBDANET uses fixed frequency transmitters which do not necessarily require acknowledgment from a receiver before transmission, "head of line" blocking does not play a role in this architecture.

Transmission between the communicating nodes is performed via a star coupler. Since the passive star provides no amplification of the signals upon receipt, amplifiers are used at the receivers' stations. In order to prevent signal degradation due to an exponential number of signal taps (as many as N^2), the receiving stations are isolated from the remainder of the network using either optical amplifiers or wavelength demultiplexors.

LAMBDANET allows for point-to-point, multicast, or broadcast capabilities. Since

[1]The potential for replacing the fixed receivers with a tunable receiver exists, but the use of fixed frequency receivers was based not only on the ability to access the necessary frequencies in a timely fashion, but also the ability to access information from several frequencies simultaneously [12].

a node would have to explicitly monitor a frequency for the possibility of a multicast transmission, the synchronization mechanism for these capabilities could be moved up the protocol stack as a higher layer capability which may be managed by an operating system or application. Additionally, a synchronization frequency, common to all nodes in the network and slotted such that each station is granted a priori a known slot for its transmission(s), could be used to indicate broadcast, session setup or takedown, user-to-user signaling, or other intermittent states [31].

LAMBDANET was intended for transmitting circuit switched traffic (e.g., video and voice traffic). In reported experiments using unspecified video traffic, LAMBDANET was implemented with 16 frequencies and nodes; in these experiments, as much as 1.5 Gb/s were able to be transmitted over 57.8 km with a point-to-point bandwidth-distance product of 1.56 Tb-km/s. When the data was broadcasted to all 16 nodes, the network capacity for broadcasting was found to be 21.5 Tb-km/s [43]. Other experimental results for circuit switched experiments were reported in [11].

Packet switched data can also easily be transmitted via LAMBDANET. It is speculated that bandwidth capabilities in this architecture can exceed 100 Gb/s with the most significant drawback being the investment in the multiple fixed-frequency receivers [30]. In addition to this issue is the development of high speed electronic hardware and (possibly) complex software which can integrate and act on data from multiple receivers simultaneously to take advantage of the multiple available data streams.

3.3 The Fast Optical Crossconnect ("FOX")

The Fast Optical Crossconnect experiment, or FOX, addresses the issue of connectivity between shared processors and memory. In order to provide addressability between an arbitrary number of processors and an arbitrary number of shared memory segments, these items are treated as nodes within a slotted wave-division multiplexed (WDM) network. Two passive star couplers are used to transmit cells between the processors and the memories: one star for processor to memory cells and another star for the opposite direction of transmission ([1], [11], [36]).

Consider first the CPU to memory star coupler. Each memory is assigned a fixed frequency which is implemented with a fixed frequency filter at the receiver. The processors have tunable transmitters. When transmission between a given processor and a memory

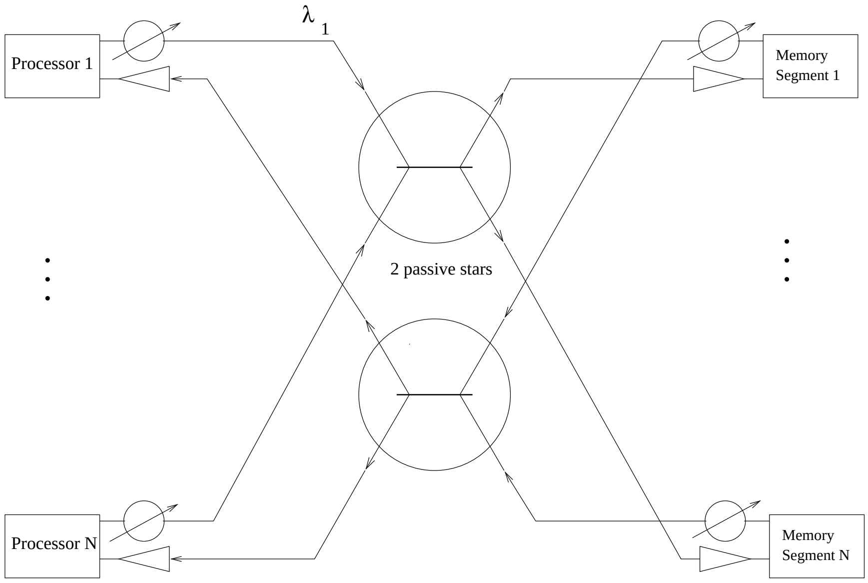

Figure 3.3: Fast Optical Crossconnect ("FOX") Architecture

segment is desired, the transmitter performs no scheduling; instead it simply transmits its request on the appropriate frequency. Given this type of system, collisions may occur if two CPU's attempt to access the same segment of memory simultaneously. Since the processors are not designed to monitor their own outgoing transmissions (although they could be in theory), and the memory segments can not necessarily accurately identify the colliding processors, the processors must retransmit their requests if after a given period expires without acknowledgment by the requested memory. If certain assumptions are made regarding traffic generation patterns and characteristics, this phenomena can be easily accommodated with an ALOHA network-type of exponential backoff period on the part of both transmitters before attempting retransmission. A similar network between the memories and the processors allow for communications in the reverse direction [1].

The actual performance of such a system will be highly dependent on the patterns of (non-cached) memory requests by the processors (i.e., the inter-request time and request length distributions, among others). This fact is extremely critical in that the probability of collisions and retransmissions affects this system just as the probability of collisions and retransmissions affect ALOHA-type systems. This scheme performs best when lightly loaded and when packet sizes are minimal (thereby lessening the probability of collision) [30]. Assuming that the switch grows arbitrarily large and all memory requests must be transmitted to the appropriate memory segment (as opposed to being managed by processor cache, i.e., "100% cache miss-rate" [1], or a worst case scenario), then the throughput for each of the processors will be approximately 58%. A more realistic situation would show approximately 10% of the memory requests being transmitted to the appropriate memory segments; assuming either "ideal" circumstances in which no collisions exist or an exponential backoff period collision resolution scheme, the throughput of the processors is estimated as being approximately 98%.

Since this architecture does not queue requests on the input side of either switch, blocking is not exhibited here. However, under heavy loading, the necessity for a node to wait for a response to a message and then potentially determine that retransmission is needed causes a significant degradation to the performance. Similarly, if the loading on this system becomes excessive, the tunable transmitters will not be able to retune themselves for transmissions before queueing does occur within the processor.

Proving that fast tunable laser transmitters were feasible with current technology within a parallel-processing application was the original motivation for FOX; in the experiments reported in [1] and [11], the FOX architecture was implemented with two frequencies separated by 0.06 nm (resulting in a tuning time less than 20 ns) transmitting 1 Gb/s using 100 bit packets. Mukherjee claims that tuning times on the order of a few tenths of the transmission time of a packet should provide adequate efficiency for throughput [48].

3.4 HYPASS/BHYPASS Architectures

Two architectures evolved from Bellcore's work on the FOX system: HYPASS and BHYPASS. These switches also use two uni-directional star couplers in parallel: one for data transmission, the other for feedback and control. In HYPASS, the input packets are received from optical lines, converted to the electrical domain via opto-electronic converters, and buffered while destination decoding and transmitter tuning occurs. The packet then waits until a "Request to Send" signal is sent by the intended recipient via the "control" path within the switch. The packet is then transmitted through the passive star coupler on the frequency designated for the destination port. The packet is received by a fixed frequency receiver, and, provided no collision has occurred, an acknowledgment is generated. Regardless of a collision, another "Request to Send" signal is transmitted on a fixed frequency allocated to the destination port through a second passive star coupler and

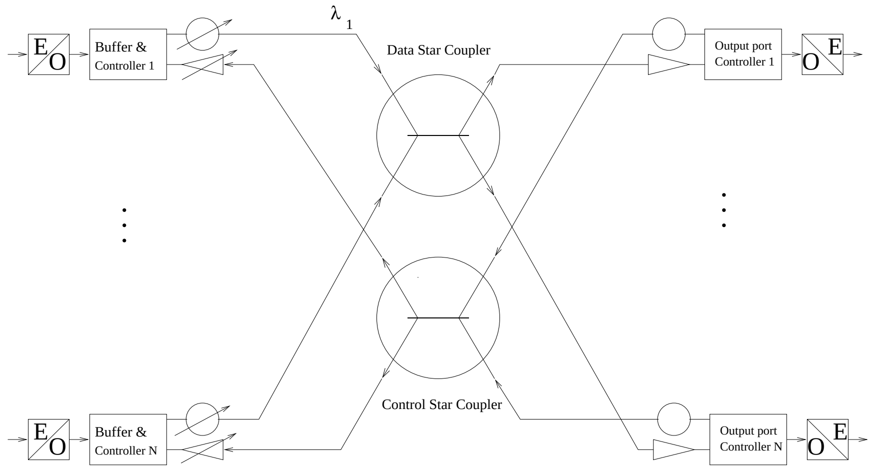

Figure 3.4: HYPASS Architecture

received by tunable receivers at the input ports ([7], [16], [30]).

Like the FOX architecture, this system's efficiency is primarily limited by the probability of collisions and/or retransmissions. Assuming independent "random" traffic, [7] shows that the switch can ideally achieve 58.6% input port utilization and, on average, achieves 31.8% utilization. The architecture contributes to these utilization values due to the "head of line" blocking effect intrinsic to the system. The derivation for these figures is provided in [7].

Additionally, performance may be significantly degraded depending on the specific implementation of electronics usedin the buffering mechanism. Since the optical input lines' data is assumed to be received serially, the serial-to-parallel processor adjacent to the optoelectronic converter and the parallel-to-serial processor between the packet buffer and the star coupler must both operate above the single connection data rate of the input and output lines; the HYPASS prototype operated at approximately 2Gb/s [7].

Arthurs, et al. proposes in [7] the contention resolution protocol described above, as well as modifications to support both CBR and variable or asnychronous bit rate (VBR or ABR) traffic requirements. High bandwidth CBR traffic may assigned to slots and frequencies by an output port upon call setup withrequirements transmitted via an indicative "Request to Send" signal from the output port which is awaiting the priority traffic. This signal may be used to identify the input port which is solicited for traffic, although it is proposed to only differentiate the preassigned CBR transmission slots from VBR or ABR

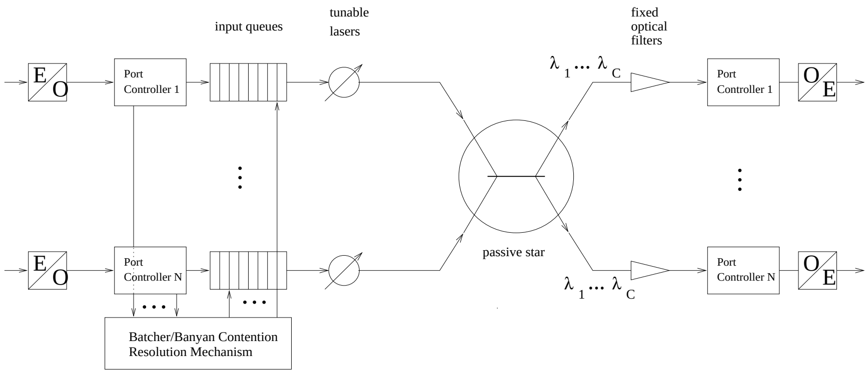

Figure 3.5: BHYPASS Architecture

cells or packets. Another mechanism which could potentially be used would be broadcasting per-call fixed bandwidth requirements or slot allocations per frequency via a shared clocking/synchronization channel to which all input and output nodes have access.

The BHYPASS architecture is similar to HYPASS, but the control/feedback network is replaced by an electronic Batcher-Banyan contention resolution mechanism. Each cell's destination address is transmitted to the network when the cell is buffered at the input port. The Banyan network resolves the contention for the output ports by processing the requested destination addresses in electronic domain. Those cells which are allowed to pass through the coupler are then transmitted as in the HYPASS system while the remaining cells may be either buffered for transmission during thenext slot or lost. Two significant advantages of the BHYPASS design over HYPASS are that the BHYPASS system does not require the tunable receivers of its predecessor and that cells transmitted in the BHYPASS switch architecture do not collide.

3.5 Star-Track Architecture

The Star-Track switching architecture is designed to specifically address multicast packet transmission. Input cells are buffered upon arrival to an input port. The cells remain in the buffer while a circulating token on a control network is received by the node's processor. The node indicates that a cell is waiting to be transmitted on the input node's dedicated frequency to one or more desired destinations by writing information to the token. Once all of the input nodes have been polled, the token traverses the output nodes' processing ports, informing these nodes of the awaiting cells. The output ports' tunable

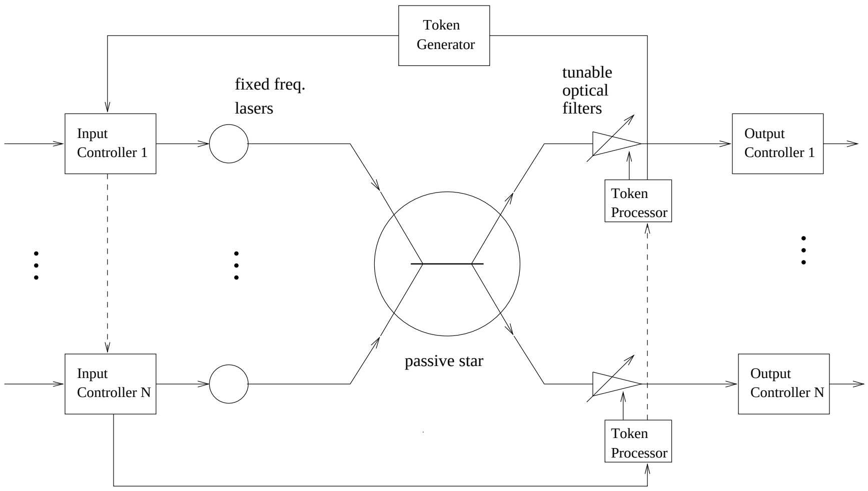

Figure 3.6: Star-Track Architecture

receivers are then tuned one-by-one to the appropriate frequencies, after which the token returns to the input ports which have since received and buffered new cells. The cells from the previous cycle are transmitted and information regarding the new arrivals' destinations are written to the token ([6], [11], [28]).

The architecture described in [30] states that the processing involved in the generation and maintenance of the token is currently only feasible in the electronic domain. This point allows for the possibility of tuning times comparable to the electrical transmission times and processing times (which are currently available on the order of tens and hundreds of nanoseconds). However, the utilization of the passive star coupler on which the switch architecture is built will be extremely low. As a result, extensive buffering at the input ports may be necessary if the switch is not lightly loaded; the storage of the cells in the electronic domain, is not a concern in this architecture, though, since the time to access the electronic memory is dominated by the token transmission time. Alternatively, the processing time required at idle stations in a scenario in which the input lines (and, hence, the switch) are underutilized will similarly constrict the performance. Goodman reports in [30] projections that the Star-Track architecture can achieve 40-60 Gb/s for a 64 port switching fabric.

One scheme for improving the performance of this architecture is designing the control network to contain multiple tokens, each circulating on a parallel control network [5]. "The Multiple Major Track Switch" therefore lessens the time per cycle of the electronic token (which still traverses all addressable output ports). The performance of such a system is shown in [44] to allow for approximately 60% throughput with only one track (given randomly uniform traffic and that the number of input ports is substantially larger than the number of tracks) and increases to 95% when the number of tracks increases to 10. Furthermore, knowing that a certain number of tracks can support a maximum traffic rate, it has been shown that buffer sizes of 16 cells can support loss probabilities of 10^{-9} when 70% of the maximum traffic rate is offered [44].

Multicasting is an obvious advantage to this architecture. For a station to perform such an operation, the token need only reflect that the necessary set of output ports tune their receivers to the input and the token itself will initiate the transmission when that task is complete. The delay involved in accessing the stations may, again, have ramifications in the buffering of new cells since all of the output ports may not be available during the first requested slot. Allowing the frequency allocated to the input channel to remain idle due to "HOL" blocking, however, may create undesirable performance measures.

With certain guarantees or assumptions regarding the input traffic, this restriction may be able to be circumvented. For instance, if a fixed maximum number of cells may be received and buffered at an input port (under an approach similar to that for the Photonic Knockout Architecture), the requirements of all of the cells may be able to be transmitted to the token (provided the bandwidth consumption of the token does not become extreme) and then processed by a contention resolution mechanism. This mechanism would then generate the token again which would in turn inform the output ports to which frequency to tune and then notify the input ports of which cell to transmit. Provided an acceptable loss rate could be achieved over long periods, this scheme may be an acceptable compromise. Prioritization of traffic may also be easily implemented in a distributed fashion between the input nodes: one station may request an output port and, subsequently, may be overridden by another station by simply rewriting the token appropriately. Designing the network with the electronic being cycled between the input ports may implement prioritization schemes with several gradations of priority ("minor loops") [44]. Reserved traffic bandwidth may be also be implemented [11].

Continual resynchronization and broadcasting of small amounts of information via the token also may be implemented within this architecture. Monitoring the regularity of the token to dynamically determine input cell buffering capabilities may be possible. The opposing view of this point, though, is the fault-tolerance of the architecture: severing the control network would result in the system's complete failure. As the length of the

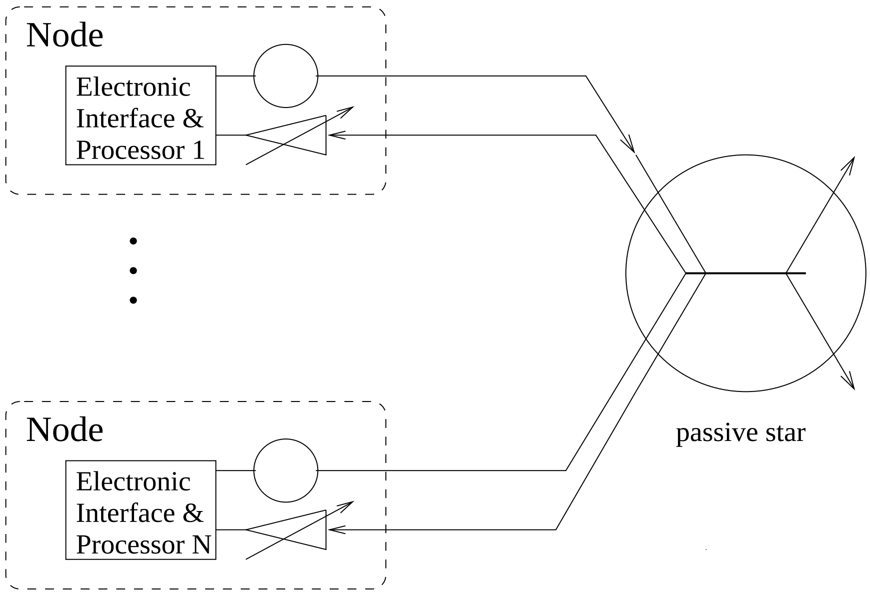

Figure 3.7: Rainbow Architecture

control network's implementation (i.e., the distance between the star coupler and the input & output port controllers) increase, the vulnerability of the system would also increase.

3.6 Rainbow

Developed by IBM, Rainbow is an architecture which focuses on circuit-switched traffic. The system uses fixed frequency transmitters and tunable receivers. The receivers continually scan the transmitters' frequencies for requests to send information to an identified station. Once the identified output port receives the request and identifies itself as the recipient of the data, it transmits an acknowledgment to the input port which then initiates the transmission [57]. Since the overhead required for the polling by the output ports is significant (the maximum tuning time within the prototype was on the order to 25 ms), this system is probably not appropriate for transmitting packet switched traffic; the overhead, however, does serve the purpose of dominating (i.e., masking) the electronic processing time required to determine the required recipient of the data. Multicasting (for applications such as scheduled broadcast video and tele- or video-conferencing) can be accommodated by this architecture by the transmitting input port simply holding its initial transmissions until acknowledgments have been received from all destinations.

Regarding issues of practicality, the system was implemented using 32 workstations with interface cards which allowed for each to transmit and receive information at 300 Mb/s. The transmitters and receivers, however, were not co-located with the traffic generators, but were centrally housed about the switch itself [37]. Since the frequencies used were adjacent to one another in the 1.5um range (separated by about 1nm each), scanning the frequencies was fairly simple. Scalability to a larger number of frequencies could become an issue, however, if the frequencies are not contiguous. To wit, the dynamic introduction of new frequencies which do not exist within a currently scanned band of the spectrum would require either a more complex protocol (to allow the new station to notify peers of its existence) and/or a shared control channel (to perform a similar task).

The peak performance of the switch was found to be approximately 45% for 32 stations. Either increasing the number of stations or decreasing the message size increased contention within the system, thereby decreasing the peak utilization of the switch; the reverse had the expected opposite effects. Additionally, the extrapolated and simulated delays in the system rose extremely slowly as the throughput rose to its peak value, and then increased dramatically, indicating that the stability of the system beyond its peak operating point is extremely unstable [39].

A new version of this network, Rainbow–II, has been implemented in a testbed at Los Alamos National Laboratory. This system allows for 32 stations to communcate at 1 Gbps speeds, compared to the 300 Mbps data rate of its predecessor. These speeds were found to be incompatible with the "off–the–shelf" protocols which were used in the Rainbow system. The architecture (both hardware and software) are described in [24]; no delay or loss measures have yet been reported for Rainbow–II.

Chapter 4

Analysis of a TTFR Photonic Switch Supporting Fixed Length Transmission Units

In this chapter we present for analysis a single-hop broadcast-and-select WDM architecture [48]. The next section, in conjunction with Chapter 2, describes our system and traffic models and provides some background information. We then develop a queueing-based decomposition algorithm to study the performance of a single-hop switch architecture in terms of its occupancy distribution on both the input and output sides of the switch. After summarizing the algorithm, we then derive loss probability expressions for both sides of the switch in Section 4.3. The delay distribution for traversing the switch is presented in Section 4.4, followed by numerical results and interpretations.

4.1 The Switch Architecture

Consider an optical switch architecture with N input ports and N output ports interconnected through a broadcast passive star (the switch fabric) supporting $C \leq N$ wavelengths $\lambda_1, \cdots, \lambda_C$ (see Figure 4.1). Each input port is equipped with a laser that enables it to inject signals into the optical medium. Similarly, each output port is capable of receiving optical signals through an optical filter. The laser at each input port is assumed to be tunable over all available wavelengths. The optical filters, on the other hand, are

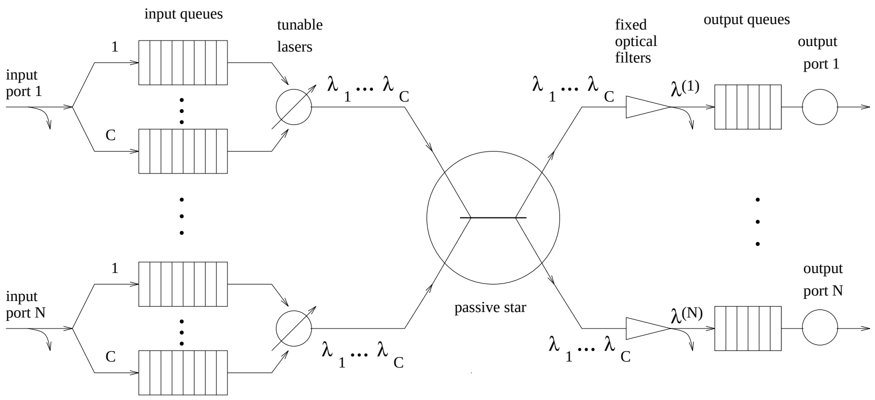

Figure 4.1: Queueing model of a switch architecture with N ports and C wavelengths

fixed to a given wavelength. Let $\lambda(j)$ denote the receive wavelength of output port j. Since $C \leq N$, a set $\mathcal{R}_c$ of output ports may be sharing a single receive wavelength λ_c:

$$\mathcal{R}_c \quad = \quad \{j \ : \ \lambda(j) = \lambda_c\}, \qquad c = 1, \cdots, C \tag{4.1}$$

Sets $\mathcal{R}_c$ will typically be obtained by running a load balancing algorithm [9].

The switch operates in a slotted mode. Since there are N ports but $C \leq N$ channels, each channel must run at a rate $\frac{N}{C}$ times faster than the rate of the input links ($\frac{N}{C}$ need not be an integer). The rate of an output link is equal to the rate of an input link. Thus, *arrival* slots (which correspond to the ATM cell transmission time at the input/output link rate) and *service* slots (which are equal to the cell transmission time at the channel rate within the switch) are distinguished as different units of time. Obviously, the duration of a service slot is equal to $\frac{C}{N}$ times that of an arrival slot. Without loss of generality, assume that all input links are synchronized at arrival slot boundaries; similarly for output links. On the other hand, all C channels internal to the switch are synchronized at service slot boundaries.

The switch employs electronic queueing at both the input and output ports, as Figure 4.1 illustrates. Cells arrive at an input port i and are buffered at a finite capacity queue, if the queue is not full. Otherwise, they are dropped. As Figure 4.1 indicates, the buffer space at each input port is assumed to be partitioned into C independent queues. Each queue c at input port i contains cells destined for the output ports which listen to a particular wavelength λ_c, $c = 1, \cdots, C$. This arrangement eliminates the head-of-line problem, and permits an input port to send a number of cells back-to-back when tuned to a certain wavelength. We let $B_{ic}^{(in)}$ denote the capacity of the queue at input port i corresponding to wavelength λ_c.

Cells buffered at an input port are transmitted on a FIFO basis onto the optical medium by the port's laser. This transmission takes place on an appropriate service slot which guarantees that the cell will be correctly received by its destination output port. Upon arriving at the output port, the cell is once again placed in a finite capacity buffer. Let $B_j^{(out)}$ denote the buffer capacity of output port j. Cells arriving at an output port to find a full buffer are lost. Cells in an output buffer are also served on a FIFO basis.

4.2 Traffic Model

The arrival process to each input port of the switch is characterized by a two-state Markov Modulated Bernoulli Process (MMBP), hereafter referred to as 2-MMBP. A 2-MMBP is a Bernoulli process whose arrival rate varies according to a two-state Markov chain. It captures the notion of burstiness and the correlation of successive interarrival times, two important characteristics of ATM type of traffic. For details on the properties of the 2-MMBP, the reader is referred to [52]. (We note that the algorithm for analyzing the switch was developed so that it can be readily extended to MMBPs with more than two states.)

We assume that the arrival process to port i, $i = 1, \cdots, N$, is given by a 2-MMBP characterized by the transition probability matrix $\mathbf{Q}_i$, and by $\mathbf{A}_i$ as follows:

$$\mathbf{Q}_i = \begin{bmatrix} q_i^{(00)} & q_i^{(01)} \\ q_i^{(10)} & q_i^{(11)} \end{bmatrix} \quad \text{and} \quad \mathbf{A}_i = \begin{bmatrix} \alpha_i^{(0)} & 0 \\ 0 & \alpha_i^{(1)} \end{bmatrix} \tag{4.2}$$

In (4.2), $q_i^{(kl)}, k, l = 0, 1$, is the probability that the 2-MMBP will make a transition to state l, given that it is currently at state k. Obviously, $q_i^{(k0)} + q_i^{(k1)} = 1,\ k = 0, 1$. Also, $\alpha_i^{(0)}$ ($\alpha_i^{(1)}$) is the probability that an arrival will occur in a slot at state 0 (1). Transitions between states of the 2-MMBP occur only at the boundaries of *arrival* slots. We assume that the arrival process to each input port is given by a different 2-MMBP.

Let r_{ij} denote the probability that a cell arriving to input port i will have j as its destination output port; $\{r_{ij}\}$ is referred to as the *routing* probabilities. This description implies that the routing probabilities can be input port dependent and non-uniformly distributed. The destination probabilities of successive cells are not correlated. That is, in an input port, the destination of one cell does not affect the destination of the cell behind it. This assumption is reasonable when the switch is used as part of a backbone network. Given these assumptions, the probability that a cell arriving to port i will have to be transmitted on wavelength λ_c is:

$$r_{ic} = \sum_{j \in \mathcal{R}_c} r_{ij}, \qquad i = 1, \cdots, N \tag{4.3}$$

4.3 Queueing Analysis

In this section, the queueing network shown in Figure 4.1 is analyzed. This queueing network represents the tunable-transmitter, fixed-receiver switch under study. The arrival process to each input port is assumed to be a 2-MMBP, and the access of the input ports to the wavelengths is governed by the schedule described in Section 2.2. The objective of the analysis of this queueing network is to obtain the queue-length distribution in an input or output port, from which performance measures such as the cell-loss probability can be obtained.

4.3.1 Input Side Analysis

In this section, we obtain the queue length distribution of an input queue. We first sketch an exact decomposition of the corresponding queueing network which, however, is not scalable to large systems. Then, we present in detail an approximation method which, as we will show later, gives accurate results.

Exact Queueing Analysis

We first observe that we can analyze the input side of th switch by decomposing it into N sub-systems, each corresponding to an input port, and analyzing each sub-system in isolation. Because of the fact that (a) the arrival processes to the various input queues are independent, (b) the way the schedule is constructed (i.e., that different inputs transmit to the same wavelength at different times), and (c) the operation of the input ports is independent of the operation of output ports, this decomposition is exact. Furthermore, we can analyze the sub-system corresponding to input queue i by defining a $(C+2)$-dimensional stochastic process $(x, y_1, \cdots, y_C, z)$, where

- x represents the arrival slot number within a frame ($x = 0, 1, \cdots, M-1$),

- y_c indicates the number of cells in the input queue servicing λ_c ($y_c = 0, 1, \cdots, B_{ic}^{(in)}$; $c = 1, \cdots, C$), and
- z indicates the state of the 2-MMBP describing the arrival process to this queue, that is, $z = 0, 1$.

It is easy to verify that this process defines a Markov chain and, thus, the steady state joint occupancy distribution of the C queues of input port i can be obtained. Unfortunately, the state space of the Markov chain grows in size as $O(2M \prod_{c=1}^{C} B_{ic}^{(in)})$. As a result, this analysis can only be applied to trivial systems. In the next subsection, we describe an approximate decomposition that can be applied to large systems.

Approximate Queueing Analysis

Our main approximation is to assume that arrivals to each queue of a given input port are independent and are generated by the original 2-MMBP (which characterizes the arrival process to the input port) appropriately thinned using the routing probabilities r_{ic}.

Assuming independence of arrivals among the queues of each input port, the original queueing network can now be decomposed into C sub-networks, one per wavelength, as in Figure 4.2. For each wavelength λ_c, the corresponding sub-network consists of N input queues, and all the output queues that listen to wavelength λ_c. Each input queue of the sub-network is the queue associated with wavelength λ_c in each input port of the switch. That is, the i-th input queue of this sub-network is the c-th queue of input port i. Since throughout this section we only consider the sub-network corresponding to λ_c, we will simply refer to this queue as "input queue i". These input queues will transmit to the output queues of the sub-network over wavelength λ_c. In view of this decomposition, it suffices to analyze a single sub-network, since the same analysis can be applied to all other sub-networks.

Consider now the sub-network for wavelength λ_c. We will analyze this sub-network by decomposing it into individual input and output ports. As discussed in the previous section, each input queue i of the sub-network is only served for a_{ic} consecutive service slots per frame. During that time, no other input port is served. Input queue i is not served in the remaining slots of the frame. In view of this, there is no dependence among the input queues of the sub-network, and consequently each one can be analyzed in isolation in order to obtain its queue-length distribution.

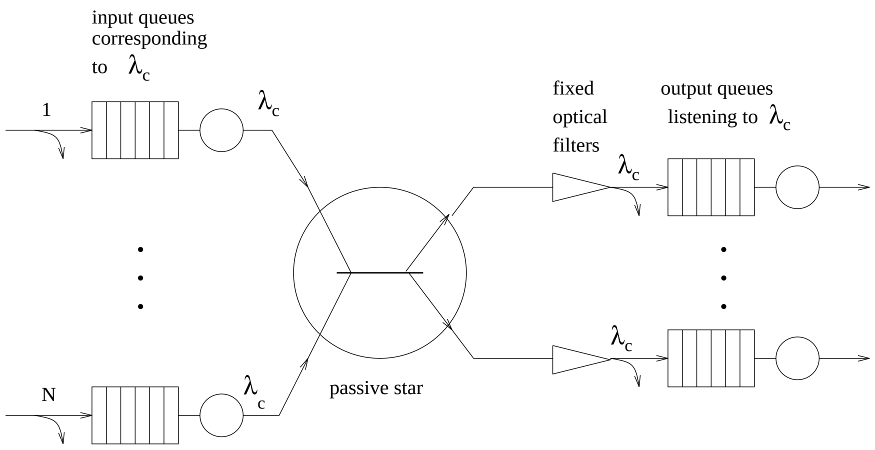

Figure 4.2: Queueing sub-network for wavelength λ_c

From the queueing point of view, the queueing network shown in Figure 4.2 can be seen as a polling system in discrete time. Despite the fact that polling systems have been extensively analyzed, we note that very little work has been done within the context of discrete time (see, for example, [69]). In addition, this particular problem differs from the typical polling system since we consider output queues, which are not typically analyzed in polling systems.

The Queue-Length Distribution of an Input Queue

Consider input queue i of the sub-network for λ_c in isolation. This input queue receives exactly a_{ic} service slots on wavelength λ_c, as shown in Figure 4.3(a). The block of a_{ic} service slots may not be aligned with the boundaries of the arrival slots. For instance, in the example shown in Figure 4.3(a), the block of a_{ic} service slots begins at the second service slot of arrival slot $x-1$, and it ends at the end of the second service slot in arrival slot $x+1$. Here, $x-1$, x, and $x+1$ represent the arrival slot number within a frame.

For each arrival slot, define $v_{ic}(x)$ as the number of service slots allocated to input queue i, that lie within arrival slot x [1]. Then, in the example in Figure 4.3(a), we have:

[1]In Figure 4.3, we assume that each arrival slot contains an integral number of service slots. If this is not the case, $v_{ic}(x)$ is defined as the number of service slots that are concluded within arrival slot x (i.e., if there is a service slot that lies partially within arrival slots x and $x+1$, it will be counted in $v_{ic}(x+1)$).

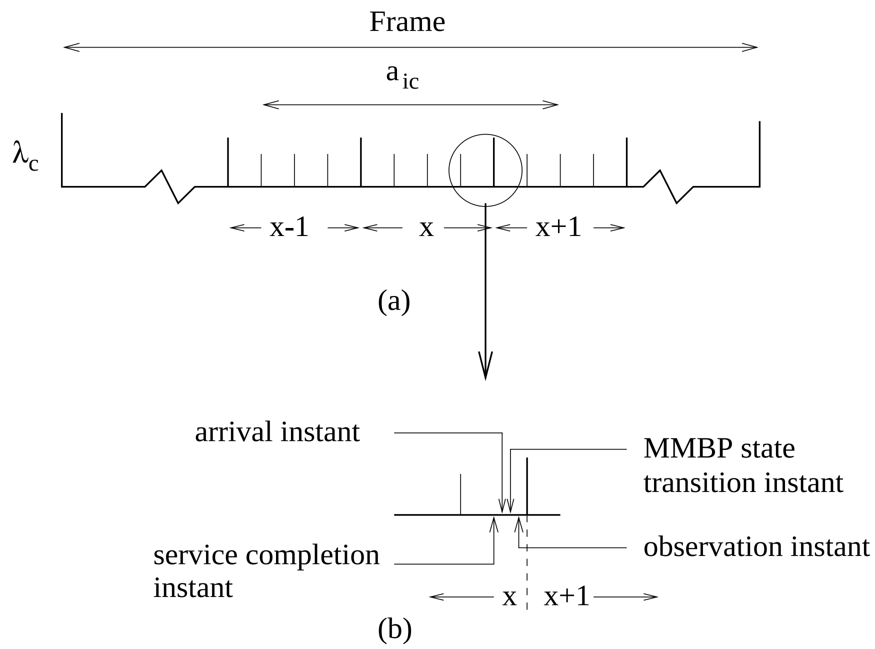

Figure 4.3: (a) Service period of input port i on channel λ_c, and (b) detail showing the relationship among service completion, arrival, 2-MMBP state transition, and observation instants within a service and an arrival slot

$v_{ic}(x-1) = 3, v_{ic}(x) = 4, v_{ic}(x+1) = 2$, and $v_{ic}(x') = 0$ for all other x'. Obviously we have,

$$\sum_{x=0}^{M-1} v_{ic}(x) = a_{ic} \tag{4.4}$$

We analyze input queue i by constructing its underlying Markov chain embedded at arrival slot boundaries. The order of events is as follows. The service (i.e., transmission) completion of a cell occurs at an instant just before the end of a service slot. An arrival may occur at an instant just before the end of an arrival slot, but after the service completion instant of a service slot whose end is aligned with the end of an arrival slot. The 2-MMBP describing the arrival process to the queue makes a state transition immediately after the arrival instant. Finally, the Markov chain is observed at the boundary of each arrival slot, *after* the state transition by the 2-MMBP. The order of these events is shown in Figure 4.3(b).

The state of the input queue is described by the tuple (x, y, z), where:

- x represents the arrival slot number within a frame $(x = 0, 1, \cdots, M-1)$,
- y indicates the number of cells in the input queue $(y = 0, 1, \cdots, B_{ic}^{(in)})$, and

Table 4.1: Transition probabilities out of state (x, y, z) of the Markov chain

Current State	Next State	Transition Probability
(x, y, z)	$(x \oplus 1,$ $\max\{0, y - v_{ic}(x \oplus 1)\}, z')$	$q_i^{(zz')}(1 - \alpha_i^{(z)} r_{ic})$
(x, y, z)	$(x \oplus 1,$ $\min\{B_{ic}^{(in)}, \max\{0, y - v_{ic}(x \oplus 1)\} + 1\}, z')$	$q_i^{(zz')} \alpha_i^{(z)} r_{ic}$

- z indicates the state of the 2-MMBP describing the arrival process to this queue, that is, $z = 0, 1$.

It is straightforward to verify that, as the state of the queue evolves in time, it defines a Markov chain. Let $\oplus$ denote modulo-M addition, where M is the number of arrival slots per frame. Then, the transition probabilities out of state (x, y, z) are given in Table 4.1. Note that, the next state after (x, y, z) always has an arrival slot number equal to $x \oplus 1$. In the first row of Table 4.1 we assume that the 2-MMBP makes a transition from state z to state z' (from (4.2), this event has a probability $q_i^{(zz')}$ of occurring), and that no cell arrives to this queue during the current slot (from (4.2) and (4.3), this occurs with probability $1 - \alpha_i^{(z)} r_{ic}$). Since at most $v_{ic}(x \oplus 1)$ cells are serviced during arrival slot $x \oplus 1$, and since no cell arrives, the queue length at the end of the slot is equal to $\max\{0, y - v_{ic}(x \oplus 1)\}$. In the second row of Table 4.1 we assume that the 2-MMBP makes a transition from state z to state z' and a cell arrives to the queue. This arriving cell cannot be serviced during this slot, and has to be added to the queue. Finally, the expression for the new queue length ensures that it will not exceed the capacity $B_{ic}^{(in)}$ of the input queue.

We observe that the probability transition matrix of this Markov chain has the following block form:

$$
\mathbf{S}_{ic} \quad = \quad \begin{array}{c} \left[\begin{array}{cccccc}
0 & \mathbf{R}_{ic}(0) & 0 & 0 & \cdots & 0 \\
0 & 0 & \mathbf{R}_{ic}(1) & 0 & \cdots & 0 \\
0 & 0 & 0 & \mathbf{R}_{ic}(2) & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \cdots & \mathbf{R}_{ic}(M-2) \\
\mathbf{R}_{ic}(M-1) & 0 & 0 & 0 & \cdots & 0
\end{array} \right] \end{array}
\begin{array}{c} 0 \\ 1 \\ 2 \\ \vdots \\ M-2 \\ M-1 \end{array}
\tag{4.5}
$$

This block form is due to the fact that at each transition instant (i.e., at each arrival slot boundary), the random variable x changes to $x \oplus 1$. Changes in the other two random variables, y and z, of the state of the queue are governed by the matrices $\mathbf{R}_{ic}(x)$. There are M different $\mathbf{R}_{ic}$ matrices, one for each arrival slot x in the frame. Let us define matrices $\mathbf{X}_{ic}$ and $\mathbf{Y}_{ic}$ as follows:

$$\mathbf{X}_{ic} \quad = \quad r_{ic}\,\mathbf{A}_i\,\mathbf{Q}_i \qquad \text{and} \qquad \mathbf{Y}_{ic} \quad = \quad (\mathbf{I} - r_{ic}\,\mathbf{A}_i)\,\mathbf{Q}_i, \tag{4.6}$$

where $\mathbf{I}$ is the identity matrix. Then, the transition matrix $\mathbf{R}_{ic}(x)$ associated with arrival slot x can be written as:

$$\mathbf{R}_{ic}(x) \quad = \quad \begin{array}{c} \left[\begin{array}{ccccccccc} \mathbf{Y}_{ic} & \mathbf{X}_{ic} & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{Y}_{ic} & \mathbf{X}_{ic} & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & \mathbf{Y}_{ic} & \mathbf{X}_{ic} & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \mathbf{Y}_{ic} & \mathbf{X}_{ic} & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & \mathbf{Y}_{ic} & \mathbf{X}_{ic} & 0 & \cdots & 0 \end{array}\right] \end{array} \begin{array}{c} 0 \\ \vdots \\ v_{ic}(x \oplus 1) \\ v_{ic}(x \oplus 1) + 1 \\ v_{ic}(x \oplus 1) + 2 \\ \vdots \\ B_{ic}^{(in)} \end{array} \tag{4.7}$$

The structure of matrix $\mathbf{R}_{ic}(x)$ given in (4.7) can be explained as follows. Suppose that the number of cells y in the queue at the end of slot x is at most $v_{ic}(x \oplus 1)$. Since up to $v_{ic}(x \oplus 1)$ cells can be served within slot $x \oplus 1$, the number in the queue at the end of that slot will be 1 or 0, depending on whether an arrival occurred or not. This is indicated by the transitions in rows 0 through $v_{ic}(x \oplus 1)$ of matrix $\mathbf{R}_{ic}(x)$. However, if at the end of slot x we have $y > v_{ic}(x \oplus 1)$, then the number in the queue at the next transition will be $y - v_{ic}(x \oplus 1)$ (plus one if an arrival occurred). This is indicated by the transitions in rows $v_{ic}(x \oplus 1) + 1$ through B_{ic} of $\mathbf{R}_{ic}(x)$. Of course, y cannot exceed the queue capacity $B_{ic}^{(in)}$. Since the number of service slots $v_{ic}(x \oplus 1)$ depends on the particular slot $x \oplus 1$ within the frame, $\mathbf{R}_{ic}(x)$ is a function of x.

Matrix $\mathbf{R}_{ic}(x)$ is slightly different when $v_{ic}(x \oplus 1) = 0$. This is because, in this case, if the state of the input queue is such that $y = B_{ic}^{(in)}$, a new arrival will be discarded. So when $y = B_{ic}^{(in)}$, the 2-MMBP is allowed to make a transition, but regardless of whether or not an arrival is generated, the number of cells in the queue will remain equal to $B_{ic}^{(in)}$. Thus, the the last row of $\mathbf{R}_{ic}(x)$ will be: $[0\ 0\ \cdots\ 0\ \mathbf{Q}_i]$.

It is now straightforward to verify that the Markov chain with transition matrix $\mathbf{S}_{ic}$ is irreducible, and therefore a steady-state distribution exists. Transition matrix $\mathbf{S}_{ic}$ defines a *p-cyclic* Markov chain [63], and therefore it can be solved using any of the techniques for p-cyclic Markov chains in [63, ch. 7]. We have used the LU decomposition method in [63] to obtain the steady state probability $\pi_{ic}(x, y, z)$ that at the end of arrival slot x, the 2-MMBP is in state z and the input queue has y cells. The steady-state probability that the queue has y cells at the end of slot x, independent of the state of the 2-MMBP is:

$$\pi_{ic}(x, y) \quad = \quad \sum_{z=0,1} \pi_{ic}(x, y, z) \tag{4.8}$$

Finally, we note that all of the results obtained in this subsection can be readily extended to MMBP-type arrival processes with more than two states. For this, it would suffice to appropriately modify matrices $\mathbf{X}_{ic}$ and $\mathbf{Y}_{ic}$.

4.3.2 Output Side Analysis

We now obtain the queue-length distribution of an output queue. Our analysis follows steps similar to the input side case.

Exact Queueing Analysis

Let us suppose that the (exact or approximate) queue-length distribution of the input queues is known. Given that transmissions on different channels are independent, and that output queue receivers operate on one frequence, the output side of the switch may be decomposed into C independent sub-systems, one per wavelength. Let us consider the sub-network corresponding to channel λ_c, and let k_c be the number of output ports sharing this channel. We can then define a $(k_c + 1)$-dimensional Markov chain, where

- x indicates the arrival slot number within the frame ($x = 0, 1, \cdots, M - 1$), and
- w_n $\forall\, n = 1, \cdots, k_c$ indicates the number of cells at the n^{th} output queue which shares λ_c ($w_n = 0, 1, \cdots, B_j^{(out)}$).

The transitions out of state $(x, w_1, \cdots, w_{k_c})$ can be computed given the schedule and the queue length distribution of the input ports. However, for realistic switch dimensions, this method will lead to a state space explosion since the total number of states is of the order of $M \times \prod_{j \in \mathcal{R}_c} B_j^{(out)}$. We now proceed to describe an approximation method that can be used for large systems.

Approximate Queueing Analysis

Consider the sub-network for wavelength λ_c, and observe that the arrival process to the output queues sharing λ_c is the combination of the departure processes from the input queues corresponding to λ_c. An interesting aspect of the departure process from the input queues is that for each frame, during the sub-period a_{ic} we only have departures from the i-th input queue. This period is then followed by a gap g_{ic} during which no departure occurs. This cycle repeats for the next input queue. Thus, in order to characterize the overall departure process offered as the arrival process to these output queues, it suffices to characterize the departure process from each input queue, and then combine them. (We note that this overall departure process is quite different from the typical superposition of a number of departure processess into a single stream, where, at each slot, more than one cell may be departing.)

However, the arrival processes to the output queues listening on λ_c are not independent. Specifically, if j and j' are two output ports on λ_c, and there is a transmission from input port i to output port j in a given service slot, then there can be no arrival to output port j' in the same service slot. As in the input side case, we will nevertheless make the assumption that these arrival processes are indeed independent, and that each is an appropriately thinned (based on the routing probabilities) version of the departure process from the input queues. Note that this is an approximation only when there are multiple output ports with receivers fixed on channel λ_c.

The Queue-Length Distribution of an Output Queue

As in the previous section, we obtain the queue-length distribution of output port j at arrival slot boundaries. Recall that an arrival slot to an input queue is equal to a departure slot from an output queue. Also, arrival and departure slots are synchronized. Therefore, during an arrival slot x a cell may be transmitted to the outgoing link from the output queue. However, during slot x, there may be several arrivals to the output queue from the input queues.

Let (x, w) be the state associated with output port j, where

- x indicates the arrival slot number within the frame ($x = 0, 1, \cdots, M-1$), and
- w indicates the number of cells at the output queue ($w = 0, 1, \cdots, B_j^{(out)}$).

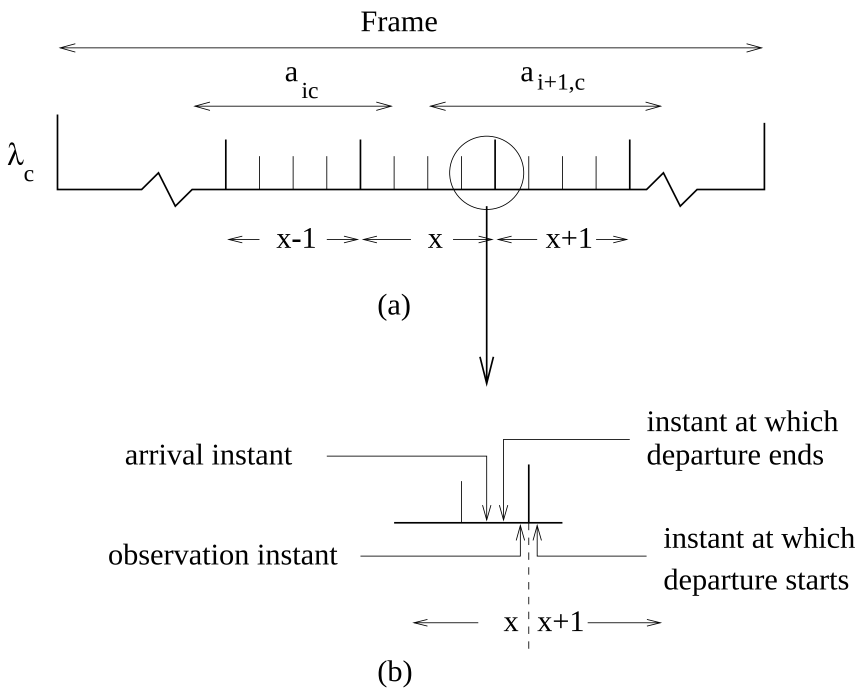

Figure 4.4: (a) Arrivals to output port j from input ports i and $i+1$, and (b) detail showing the relationship of departure, arrival, and observation instants

We assume the following order of events. A cell will begin to depart from the output queue at an instant immediately after the beginning of an arrival slot and the departure will be completed just before the end of the slot. A cell from an input port arrives at an instant just before the end of a service slot, but before the end-of-departure instant of an arrival slot whose end is aligned with the end of the service slot. Finally, the state of the queue is observed just before the end of an arrival slot and after the arrival associated with the last service slot has occurred (see Figure 4.4(b)).

Let $u_j(x)$ be the number of service slots of any input queue on wavelength λ_c within arrival slot x. We have that

$$u_j(x) \quad = \quad \sum_{i=1}^{N} v_{ic}(x) \tag{4.9}$$

where $v_{ic}(x)$ is as defined in (4.4). Quantity $u_j(x)$ represents the maximum number of cells that may arrive to output port j within slot x. In the example of Figure 4.4(a) where we show the arrival slots during which cells from input ports i and $i+1$ may arrive to output port j, we have: $u_j(x-1) = v_{ic}(x-1) = 4$, $u_j(x) = v_{ic}(x) + v_{i+1,c}(x) = 1 + 2 = 3$, and $u_j(x+1) = v_{i+1,c}(x+1) = 4$.

Table 4.2: Transition probabilities out of state (x, w) of the Markov chain

Current State	**Next State**	**Transition Probability**
$(x, 0)$	$(x \oplus 1, \min\{B_j^{(out)}, s\})$, $0 \leq s \leq u_j(x \oplus 1)$	$\sum_{s_1+\cdots+s_N=s} \prod_{i=1}^{N} L_i(s_i \mid x)$
$(x, w), w > 0$	$(x \oplus 1, \min\{B_j^{(out)}, w + s\} - 1)$, $0 \leq s \leq u_j(x \oplus 1)$	$\sum_{s_1+\cdots+s_N=s} \prod_{i=1}^{N} L_i(s_i \mid x)$

Observe now that (a) at each state transition x advances by one (modulo-M), (b) exactly one cell departs from the queue as long as the queue is not empty, (c) a number $s \leq u_j(x \oplus 1)$ of cells may be transmitted from the input ports to output port j within arrival slot $x \oplus 1$, and that (d) the queue capacity is $B_j^{(out)}$. Then, the transition probabilities out of state (x, w) for this Markov chain are given in Table 4.2.

In Table 4.2, $L_i(s_i \mid x)$ is the probability that input port i transmits s_i cells to output port j given that the system is at the end of arrival slot x (in other words, it is the probability that s_i cells are transmitted within slot $x \oplus 1$) [2]. To obtain $L_i(s_i \mid x)$, define r'_{ij} as the conditional probability that a cell is destined for output port j, given that the cell is destined to be transmitted on λ_c, the receive wavelength of output port j:

$$r'_{ij} \quad = \quad \frac{r_{ij}}{\sum_{k \in \mathcal{R}_c} r_{ik}} \quad = \quad \frac{r_{ij}}{r_{ic}} \tag{4.10}$$

This "thinning" of the arrival processes using the r'_{ij} routing probabilities discounts the correlation between arrival streams and is the crux of the approximation for the output side of the switch. The error introduced by this approximation will be discussed later in this work.

Define $\pi_{ic}(y \mid x)$ as the conditional probability of having y cells at the i-th input queue given that the system is observed at the end of slot x:

$$\pi_{ic}(y \mid x) \quad = \quad \frac{\pi_{ic}(x, y)}{\pi_{ic}(x)} \quad = \quad M \; \pi_{ic}(x, y) \tag{4.11}$$

[2] Since in most cases only one or two input ports will transmit to the same channel within an arrival slot (refer also to Figure 2.2), the summation and product in the expression in the last column of Table 4.2 does not necessarily run over all N values of i, only over one or two values of i. Thus, this expression can be computed very fast, not in exponential time as implied by the general form presented in the table.

Then, for $r'_{ij} < 1$, the probability $L_i(s_i \mid x)$ is given by

$$L_i(s_i \mid x) = \begin{cases} 0, & s_i > v_{ic}(x \oplus 1) \\ \sum_{y=s_i}^{B_{ic}^{(in)}} \pi_{ic}(y \mid x) \begin{pmatrix} \min\{y, v_{ic}(x \oplus 1)\} \\ s_i \end{pmatrix} \times & s_i \leq v_{ic}(x \oplus 1) \\ \qquad (r'_{ij})^{s_i}(1 - r'_{ij})^{\min\{y, v_{ic}(x \oplus 1)\} - s_i}, & \end{cases} \tag{4.12}$$

Expression (4.12) can be explained by noting that input port i will transmit s_i cells to output port j during arrival slot $x \oplus 1$ if (a) $v_{ic}(x \oplus 1) \geq s_i$, (b) input port i has $y \geq s_i$ cells in its queue for λ_c at the beginning of the slot (equivalently, at the end of slot x), and (c) exactly s_i of $\min\{y, v_{ic}(x \oplus 1)\}$ cells that will be transmitted by this queue in this arrival slot are for output j.

If $r'_{ij} = 1$, in which case j is the only port listening on wavelength λ_c, the expression for $L_i(s_i \mid x)$ must be modified as follows:

$$L_i(s_i \mid x) = \begin{cases} 0, & s_i > v_{ic}(x \oplus 1) \\ \pi_{ic}(s_i \mid x), & s_i < v_{ic}(x \oplus 1) \\ \sum_{y=s_i}^{B_{ic}^{(in)}} \pi_{ic}(y \mid x), & s_i = v_{ic}(x \oplus 1) \end{cases} \tag{4.13}$$

Expressions (4.12) and (4.13) are based on the assumption that $v_{ic}(x \oplus 1) < B_{ic}^{(in)}$ which we believe is a reasonable one. In the general case, quantity $v_{ic}(x \oplus 1)$ in both expressions must be replaced by $\min\{v_{ic}(x \oplus 1), B_{ic}^{(in)}\}$.

The transition matrix $\mathbf{T}_j$ of the Markov chain defined by the evolution of the state (x, w) of output queue j has the following form, which is similar to that of matrix $\mathbf{S}_{ic}$ given by (4.5):

$$\mathbf{T}_j = \begin{array}{c} \left[\begin{array}{cccccc} 0 & \mathbf{U}_j(0) & 0 & 0 & \cdots & 0 \\ 0 & 0 & \mathbf{U}_j(1) & 0 & \cdots & 0 \\ 0 & 0 & 0 & \mathbf{U}_j(2) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \mathbf{U}_j(M-2) \\ \mathbf{U}_j(M-1) & 0 & 0 & 0 & \cdots & 0 \end{array} \right] \end{array} \begin{array}{c} 0 \\ 1 \\ 2 \\ \vdots \\ M-2 \\ M-1 \end{array} \tag{4.14}$$

$\mathbf{U}_j(x)$ is a $(B_j^{(out)}+1) \times (B_j^{(out)}+1)$ matrix that governs changes in random variable w of the state of the output queue. The elements of this matrix can be determined using

Table 4.2 and expression (4.12) or (4.13). Since $L_i(s_i \mid x)$ depends on $v_{ic}(x)$ and $u_j(x)$, $\mathbf{U}_j(x)$ also depends on x, the slot number within the frame.

We observe that $\mathbf{T}_j$ also defines a *p-cyclic* Markov chain. We have used the LU decomposition method as prescribed in [63] to obtain $\pi_j(x,w)$, the steady-state probability that output queue j has w cells at the end of slot x.

4.3.3 Summary of the Decomposition Algorithm

Below we summarize our approach to analyzing the sub-network of Figure 4.2 corresponding to wavelength λ_c. We assume that quantities $\{a_{ic}\}$ and the corresponding schedule (see [60]) are given.

1. For each arrival slot x, use the schedule and expressions (4.4) and (4.9) to compute the quantities $v_{ic}(x)$ and $u_j(x)$, $i = 1, \cdots, N,\ j : \lambda(j) = \lambda_c$.

2. For each input queue i, construct the transition probability matrix $\mathbf{S}_{ic}$ from (4.2), (4.3), (4.5), (4.6), and (4.7). Solve this matrix and use (4.8) to obtain the steady-state probability $\pi_{ic}(x,y)$ that input queue i has y cells at the end of the x-th slot of the frame.

3. For each output port $j \in \mathcal{R}_c$, use $\pi_{ic}(x,y)$ derived in Step 2, and (4.12)-(4.13) to construct the transition matrix $\mathbf{T}_j$ given by (4.14). Solve the matrix as in Step 2 to obtain $\pi_j(x,w)$, the steady-state probability that port j has w cells in its queue at the end of slot x.

Note that the complexity of this approach is dominated by Step 2. For each of the N input queues we have to solve a matrix of dimensions $\left[2M(B_{ic}^{(in)}+1)\right] \times \left[2M(B_{ic}^{(in)}+1)\right]$, where M is the length of the schedule (in arrival slots) and $B_{ic}^{(in)}$ is the capacity of the respective queue. (Inverting a $K \times K$ matrix takes time $O(K^3)$, although we can take advantage of the fact that the matrix is sparse to solve for the queue-length distributions at a significantly faster rate.) Thus, in the worst case, the overall complexity of our algorithm is $O(NM^3B^3)$, where $B = \max_i\{B_{ic}^{(in)}\}$.

4.4 Cell-Loss Probability

We now use the queue-length distributions for the input and output ports, $\pi_{ic}(x, y)$ and $\pi_j(x, w)$, respectively, derived in the previous section, to obtain the cell-loss probability at the input and output ports.

4.4.1 The Cell-Loss Probability at an Input Port

Let Ω_{ic} be the cell-loss probability at the c-th queue of input port i, i.e., the probability that a cell arriving to that queue will be lost. Ω_{ic} can be expressed as:

$$\Omega_{ic} = \frac{E[\text{number of cells lost per frame at queue } c \text{ of port } i]}{E[\text{number of arrivals per frame at queue } c \text{ of port } i]} \tag{4.15}$$

Obtaining the expectation in the denominator is easy. From (4.2) and [52], the steady-state arrival probability for the arrival process to this queue is

$$\gamma_i = \frac{q_i^{(10)}\alpha_i^{(0)} + q_i^{(01)}\alpha_i^{(1)}}{q_i^{(01)} + q_i^{(10)}} \tag{4.16}$$

Then,

$$E[\text{number of arrivals per frame at queue } c \text{ of port } i] = M\ \gamma_i\ r_{ic} \tag{4.17}$$

To obtain the expectation in the numerator, let us refer to Figure 4.3(b) which shows the service completion, arrival, and observation instants within slot x. We observe that, due to the fact that at most one cell may arrive in slot x, if the number $v_{ic}(x)$ of slots during which this queue is serviced within arrival slot x is not zero (i.e., $v_{ic}(x) > 0$), no arriving cell will be lost. Even if the c-th queue at input port i is full at the beginning of slot x, $v_{ic}(x) \geq 1$ cells will be serviced during this slot, and the order of service completion and arrival instants in Figure 4.3(b) guarantees that an arriving cell will be accepted. On the other hand, if $v_{ic}(x) = 0$ for slot x, then an arriving cell will be discarded if and only if the queue is full at the beginning of x (equivalently, at the end of the slot before x). Since the 2-MMBP can be in one of two states, we have that

$$\begin{aligned} &E[\text{number of cells lost per frame at queue } c \text{ of port } i] = \\ &\textstyle\sum_{x:v_{ic}(x)=0} \sum_{z=0}^{1} \alpha_i^{(z)} r_{ic} \pi_{ic}(B_{ic}, z \mid x \ominus 1) \end{aligned} \tag{4.18}$$

In (4.18), $\ominus$ denotes regular subtraction with the exception that, if $x = 0$, then $x \ominus 1 = M - 1$, and the summation runs over all x for which $v_{ic}(x) = 0$. Using (4.15), (4.17), (4.18), and the fact that $\pi_{ic}(x) = \frac{1}{M}$ for all x, we obtain an expression for Ω_{ic} as follows:

$$\Omega_{ic} \quad = \quad \frac{\sum_{x:v_{ic}(x)=0} \sum_{z=0}^{1} \alpha_i^{(z)} \pi_{ic}(x \ominus 1, B_{ic}^{(in)}, z)}{\gamma_i} \tag{4.19}$$

4.4.2 The Cell-Loss Probability at an Output Port

The cell-loss probability at an output port is more complicated to calculate, since we may have multiple cell arrivals to the given output port within a single arrival slot (refer to Figure 4.4(a)). Let us define $\Omega_j(n \mid x)$ as the conditional probability that n cells will be lost at output queue j given that the current arrival slot is x. An output port will lose n cells in slot x if (a) the port had $w, 0 \leq w \leq B_j^{(out)}$ cells at the beginning of slot x, and (b) exactly $B_j^{(out)} - w + n$ cells arrived during slot x. We can then write:

$$\Omega_j(n \mid x) \quad = \quad \sum_{w=0}^{B_j^{(out)}} \pi_j(w \mid x \ominus 1) \; Pr[B_j^{(out)} - w + n \text{ cells arrive to } \; j \mid x] \tag{4.20}$$

where $\pi_j(w \mid x \ominus 1) = M\pi(x \ominus 1, w)$ similar to (4.11). The last probability in (4.20) can be obtained using (4.12) or (4.13), as in Table 4.2:

$$Pr[s \text{ cells arrive to output port } j \mid x] \quad = \quad \sum_{s_1+\cdots+s_N=s} \prod_{i=1}^{N} L_i(s_i \mid x \ominus 1) \tag{4.21}$$

Note that at most $u_j(x)$ cells may arrive (and get lost) in arrival slot x. Using (4.20), we can then compute the expected number of cells lost in slot x as:

$$E[\text{number of cells lost at } j \mid x] \quad = \quad \sum_{n=1}^{u_j(x)} n\Omega_j(n \mid x) \tag{4.22}$$

The expected number of arrivals to port j in slot x can be computed using (4.21):

$$E[\text{number of arrivals to } j \mid x] \quad = \quad \sum_{s=1}^{u_j(x)} s \; Pr[s \text{ cells arrive to } j \mid x] \tag{4.23}$$

Finally, the probability Ω_j that an arriving cell to output port j will be lost regardless of the arrival slot x can be found as follows:

$$\Omega_j \quad = \frac{E[\text{number of lost cells in a frame}]}{E[\text{number of arrivals in a frame}]} \quad = \quad \frac{\sum_{x=0}^{M-1} E[\text{number of lost cells at } j \mid x]}{\sum_{x=0}^{M-1} E[\text{number of arrivals to } j \mid x]} \tag{4.24}$$

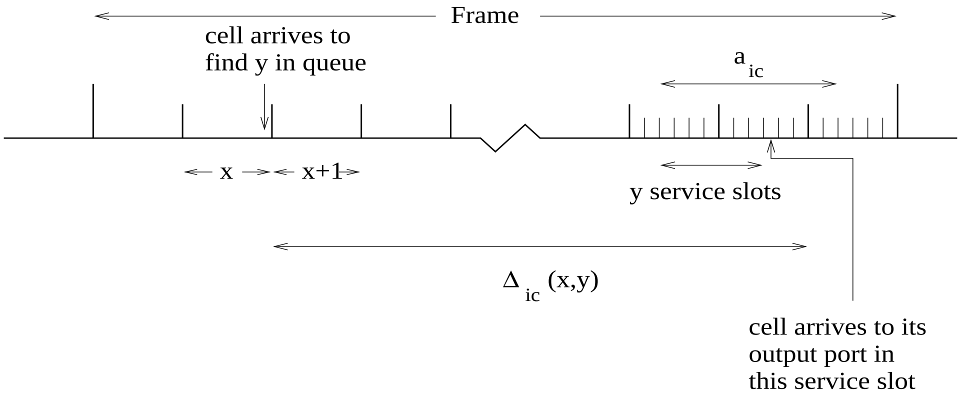

Figure 4.5: Definition of $\Delta_{ic}(x,y)$ for $y < a_{ic}$

4.5 The Delay Distribution

In this section we calculate the distribution of the number of arrival slots that elapse from the instant that a cell arrives to an input queue to the instant that the cell departs from an output queue. We note that the cell arrives and departs from the switch on arrival slot boundaries.

Let us tag a cell arriving to the c-th queue of input port i in arrival slot x. Let j be the destination output port of this cell, and let λ_c be the wavelength of j. We assume that the tagged cell sees y cells in the input queue, where $y < B_{ic}^{(in)}$. Define $\Delta_{ic}(x,y)$ to be the number of arrival slots between the end of slot x and the end of the arrival slot during which the tagged cell is transmitted to the output queue j on wavelength λ_c. If $y < a_{ic}$, then $\Delta_{ic}(x,y)$ can be calculated very easily, as shown in Figure 4.5. (Since x and y are given, we can calculate how many arrival slots will elapse until the input queue is served by wavelength λ_c, and subsequently we can calculate the number of arrival slots required to transmit y cells on this wavelength.) We note that for $y < a_{ic}$ we have $\Delta_{ic}(x,y) < M$, where M is the length of the schedule in arrival slots.

Now, let us assume that $y \geq a_{ic}$. Then, y can be written as $y = ka_{ic} + y'$, where k is an integer $k \geq 1$, and $y' < a_{ic}$. In this case, we have $\Delta_{ic}(x,y) = kM + \Delta_{ic}(x,y')$.

Finally, let w be the number of cells that the tagged cell will find in the output queue j upon arrival to this queue, where $w < B_j^{(out)}$. Therefore, the tagged cell will wait exactly w arrival slots efore it is transmitted out of the switch, for a total of $w+1$ slots.

We can now compute $\Xi_{ij}(m)$, the probability that a cell with destination j arriving to port i will spend m arrival slots in the switch as the convolution of the probability that the cell will spend $l < m$ slots in its input queue, and the probability that the cell will spend exactly $m - l$ slots in the output queue. We have:

$$\Xi_{ij}(m) = \sum_{l=1}^{m-1} Pr[l \text{ slots in input queue } i] \; Pr[m-l \text{ slots in output queue } j] \qquad (4.25)$$

Using the same reasoning as in (4.19) we can write:

$$Pr[l \text{ slots in input queue } i] = \sum_{x,y:\Delta_{ic}(x,y)=l} \pi_{ic}(x,y) \qquad (4.26)$$

where the sum is over all states (x, y) of the input queue such that the cell will spend exactly l arrival slots in the queue.

In order to compute $Pr[m - l$ slots in output queue $j]$, let us return to the tagged cell arriving to the c-th queue of input port i in slot x. Suppose that, at the time of its arrival to the switch, its input and output queues have y and w cells, respectively ($y < B_{ic}^{(in)}$). Then, the amount of time that the tagged cell spends in the switch is a function of x, y, and w. Note that the cell will arrive to its output port in arrival slot $x' = x \oplus \Delta_{ic}(x, y)$ (refer also to Figure 4.5). If it finds $w' < B_j^{(out)}$ cells in its output port at that time, it will spend another $w' + 1$ slots in the switch, for a total of $\Delta_{ic}(x, y) + w' + 1$ slots. To obtain an exact expression for $Pr[m - l$ slots in output queue $j]$, we must compute the conditional probability that the cell will find w' cells in its output queue in slot x', given that there were w cells in that same queue in slot x. This conditional probability, however, is difficult to calculate, as it depends on (a) the schedule, (b) the occupancy of the c-th queue at all other input ports, and (c) the routing probabilities.

Alternatively, we can make the simplifying assumption that, when a cell is transmitted to its output queue, the probability that it will find w cells in this queue is equal to the steady-state probability of having w cells in the queue $\pi_j(w) = \sum_{x=0}^{M-1} \pi_j(x, w)$. This is a reasonable approximation when (a) there is a relatively large number of input ports, and (b) the destination of one cell does not affect the destination of the cell behind it (as we assumed earlier). Then, we can write:

$$Pr[m - l \text{ slots in output queue } j] = \pi_j(m - l - 1) \qquad (4.27)$$

Finally, we can use (4.26) and (4.27) to rewrite the probability (4.25) that a cell will spend

Table 4.3: Channel sharing for $C = 4, 6, 8$

	$C = 4$	$C = 6$	$C = 8$
$\mathcal{R}_1$	$\{1\}$	$\{1\}$	$\{1\}$
$\mathcal{R}_2$	$\{2,5,8,11,14\}$	$\{2,7,12\}$	$\{2,9,16\}$
$\mathcal{R}_3$	$\{3,6,9,12,15\}$	$\{3,8,13\}$	$\{3,10\}$
$\mathcal{R}_4$	$\{4,7,10,13,16\}$	$\{4,9,14\}$	$\{4,11\}$
$\mathcal{R}_5$		$\{5,10,15\}$	$\{5,12\}$
$\mathcal{R}_6$		$\{6,11,16\}$	$\{6,13\}$
$\mathcal{R}_7$			$\{7,14\}$
$\mathcal{R}_8$			$\{8,15\}$

m arrival slots in the switch as

$$\Xi_{ij}(m) = \sum_{l=1}^{m-1} \left\{ \left[\sum_{x,y:\Delta_{ic}(x,y)=l} \pi_{ic}(x,y) \right] \pi_j(m-l-1) \right\} \tag{4.28}$$

4.6 Numerical Results

We now demonstrate the accuracy of our analysis by applying the decomposition algorithm to a 16×16 switch and comparing it to simulation results.

We have selected the following set of parameters for our study case. The arrival process to each of the ports of the switch is described by a different 2-MMBP. In Figure 4.6, we plot two important parameters of each of the 16 2-MMBPs we have used: the mean interarrival time in slots (γ_i^{-1} in (4.16)), and the squared coefficient of variation of the interarrival time given in [52]. As can be seen, the arrival processes exhibit a wide range of behavior in terms of these two parameters. The routing probabilities used are:

$$r_{ij} = \begin{cases} 0.10, & i = 1, \cdots, 16, \ j = 1 \\ 0.06, & i = 1, \cdots, 16, \ j = 2, \cdots, 16 \end{cases} \tag{4.29}$$

That is, output port 1 is a hot spot, receiving 10% of the total traffic, while the remaining traffic is evenly distributed to the other 15 ports. The total arrival rate to the network is $\gamma = \sum_i \gamma_i = 1.98$ cells per arrival slot. Most of the traffic enters the switch at input port 1, as the arrival rate to that port is $\gamma_1 = 0.59$ cells per arrival slot.

For load balancing purposes, we have allocated one of the C channels exclusively to output port 1, since this port receives a considerable fraction of the total traffic. The

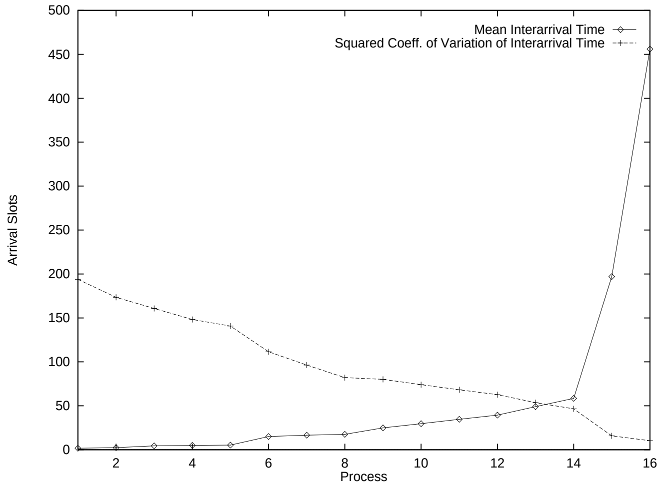

Figure 4.6: Mean arrival rate and squared coefficient of variation of the interarrival time for the arrival processes to the 16 input ports of the switch

remaining $C-1$ channels are shared by the other 15 output ports. The allocation of the output ports to the remaining frequencies was performed in a round-robin fashion, and is given in Table 4.3 for $C = 4, 6, 8$. The quantities a_{ic} of the schedule, i.e., the number of cells to be transmitted by input port i onto channel λ_c per frame (refer to Section 2.2 and Figure 2.2) were fixed to be as close to 0.5 arrival slots as possible. Recall that, while the length of an arrival slot is independent of C and is taken as our unit of time, the length of a service slot depends on the number of channels. In cases in which this value was not an integral number of service slots, the value a_{ic} was rounded up to ensure that every queue was granted at least 0.5 arrival slots of service during each frame [3] (i.e., $a_{ic} = \lceil \frac{N}{2C} \rceil \;\; \forall\, i, c$). In constructing the schedules, we have assumed that the time it takes a laser to tune from one channel to another is equal to one arrival slot [4]. Finally, for all the results we present in this section we have let all input and output queues have the same buffer capacity B (i.e., $B_{ic}^{(in)} = B_j^{(out)} = B$) so that we can vary a single parameter.

Figures 4.8, 4.9, 4.10, and 4.11 show the cell loss probability (CLP), obtained

[3]Other schemes for allocating a_{ic} have been implemented, including setting a_{ic} proportional to r_{ic}, setting a_{ic} proportional to $\max_z \alpha_i^{(z)}$, and setting a_{ic} to the effective bandwidth [54] of port i's total traffic carried on channel λ_c. Although the cell loss probability results do depend on how a_{ic} is determined, the overall conclusions drawn regarding our analysis are very similar. Thus, we have decided to include only the simplest case.

[4]Again, due to the synchronous nature of this switch, if one arrival slot is not an integral number of service slots, the number of service slots for which a transmitter cannot transmit is rounded up, thereby setting the required time for tuning to some value slightly greater than one arrival slot. In fact, the tuning time will be $\lceil \frac{N}{C} \rceil$ service slots.

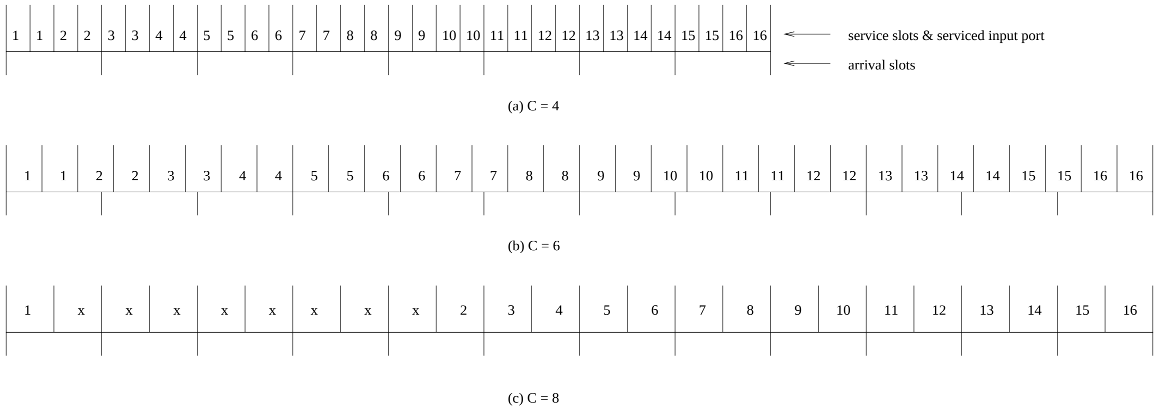

Figure 4.7: Transmission schedules for λ_1 and λ_2 for $C = 4, 6, 8$

through the analysis, at the input queues as a function of the buffer size B for $C = 4, 6, 8$. The simulation results and confidence intervals are not given as they do not provide a meaningful comparison for the scale given. The difference between analysis and simulation is discussed later in this section. We only present two cases, namely, the highest intensity port (input port 1) in Figures 4.8 and 4.10 and a representative intermediate port (input port 8) in Figures 4.9 and 4.11. We also consider only queues 1 and 2 (out of C) at each port. Queue 1 at each input port is for traffic to be carried on wavelength λ_1, which is dedicated to output port 1 (the hot spot). Thus, the amount of traffic received by this queue does not change as we vary the number of channels. Queue 2 at each input port is for traffic to be carried on wavelength λ_2. The amount of traffic received by this queue will decrease as the number of channels increases, since channel λ_2 will be shared by fewer output ports. The behavior of queue 2 is representative of the behavior of the other $C - 2$ queues, 3 through C. The portions of the schedules corresponding to λ_1 and λ_2 for $C = 4, 6,$ and 8 are shown in Figure 4.7.

In Figure 4.8, the $\Omega_{1,1}$ decreases predictably as the buffer size, B, increases. The cell loss across frequencies, however, changes due to the changing ratio of arrival slots per frame to service slots for this queue per frame. In the case in which $C = 4$, $M = 8$ and $a_{1,1} = 2$, therefore, as many as 8 cells may arrive during a frame while 2 will be serviced. For $C = 6$, $M = 12$ and $a_{1,1} = 2$, indicating an increase in the offered traffic (which is fixed per slot) between service periods. Similarly, for $C = 8$, $M = 12$ and $a_{1,1} = 1$, and the decrease in available service per frame for this queue increases the CLP. Similar behavior

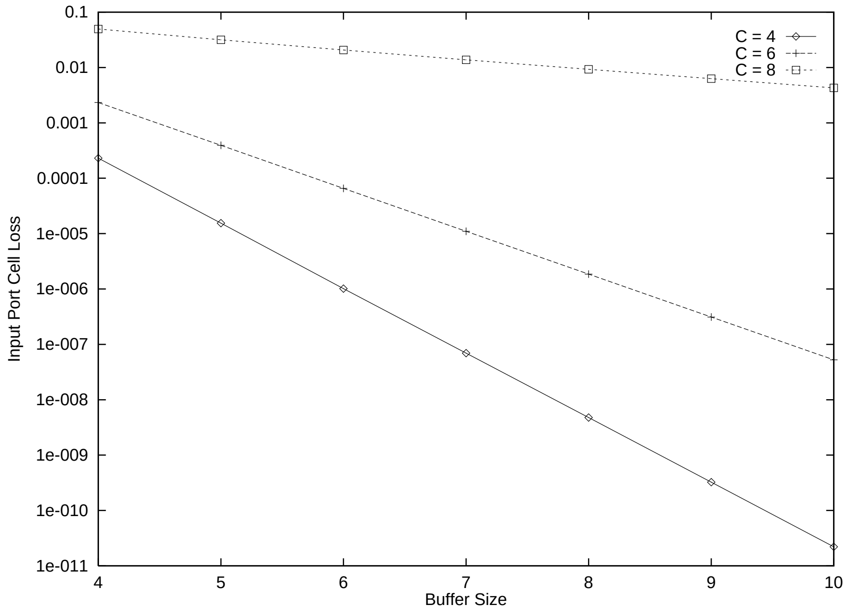

Figure 4.8: Input port cell loss probability $\Omega_{1,1}$ for $C = 4, 6, 8$ as a function of buffer size B

is observed in Figure 4.9, where we plot $\Omega_{8,1}$. We were not able to establish a reliable comparison between the analytic and simulated results for queue 1 of input port 8: the cell loss calculated analytically for queue 1 of input port 8 was 6×10^{-6}, whereas the simulation, after 30 replications of $100,000$ service slots, showed no loss.

It is obvious that queue 2 at input port 1 will be subject to not only the interactions discussed above, but also the effects of the changing traffic patterns due to the reallocation of output ports with the addition of new channels (see Table 4.3). In Figure 4.10, the traffic allocated to this queue decreases from 30% of that offered to input port 1 to 18% as C increases from 4 to 6. This change in the traffic arrival rate (from a mean normalized offered load of 0.175 to 0.105 per arrival slot) is enough to offset the change in the service rate. The CLP values for $C = 6$ are less than that for $C = 4$ in Figure 4.11 due to the fact that the change in the mean normalized offered load (from 0.121 to 0.0724 per arrival slot) is not substantial enough to offset the change in the service rate. As C increases to 8, however, there is no change in the offered traffic for either queue; as expected, the CLP rises with the decrease in the normalized service rate. The maximum absolute error observed between the anlaytic results and the mean simulated results for queue 2 of input port 1 was 17.6×10^{-3}; the loss for input queue 2 of port 8 could not be estimated for the same reason

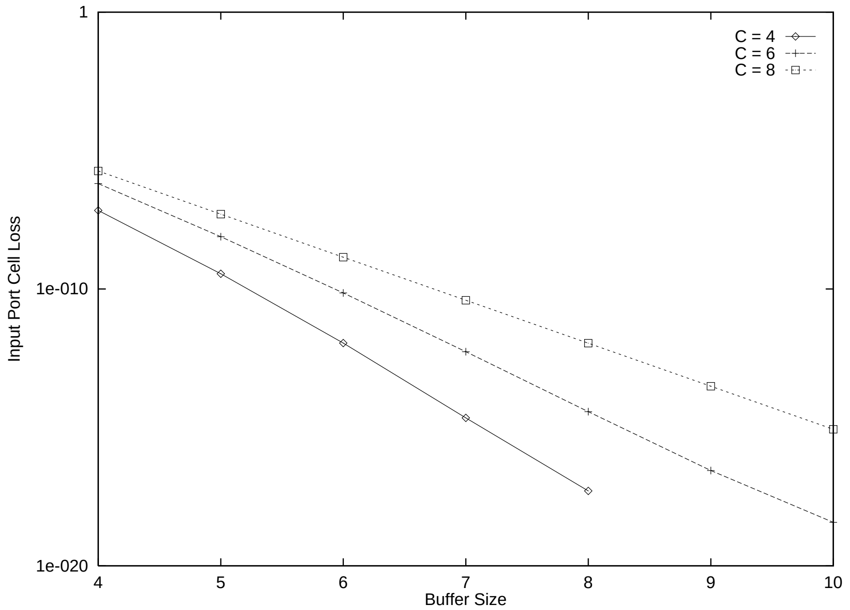

Figure 4.9: Input port cell loss probability $\Omega_{8,1}$ for $C = 4, 6, 8$ as a function of buffer size B

as that for queue 1 of input port 8 [5].

The CLP at output port 1 is similarly affected by the schedule parameters (see Figure 4.12). In the case where $C = 4$, we have $M = 8$ and $\sum_x u_1(x) = 32$, meaning that as many as 32 arrivals may be presented to this output port during a frame while only 8 may be serviced. $M = 12$ for $C = 6$ and $C = 8$ with $\sum_x u_1(x) = 32$ and $\sum_x u_1(x) = 16$, respectively. The dramatic change in $\sum_x u_1(x)$ is due to two major changes in the schedule's structure. For $C = 8$, the tuning time for a transmitter changes from 4 service slots to 2 service slots and the values of a_{ic} change from 2 service slots to 1 service slot. The effect of these parameter changes on the schedule is that a significant amount of the bandwidth in the 8 channel case is unused due to transmitters having to tune to the many frequencies. The reader is directed to [61] for a complete discussion of this phenomena. The CLP for output port 8 behaves similarly in Figure 4.13. The maximum absolute error between the simulated and analytic loss values for output ports 1 and 8 were 1.45×10^{-4} and 1.6×10^{-3}, respectively.

Finally, in order to validate the accuracy of the approximation algorithm, we ran several experiments involving different switch configurations. The switches varied from four ports and two frequencies to 16 ports with 10 frequencies. We observed that the smallest

[5]The maximum loss calculated for queue 2 of input port 8 was 7×10^{-6}.

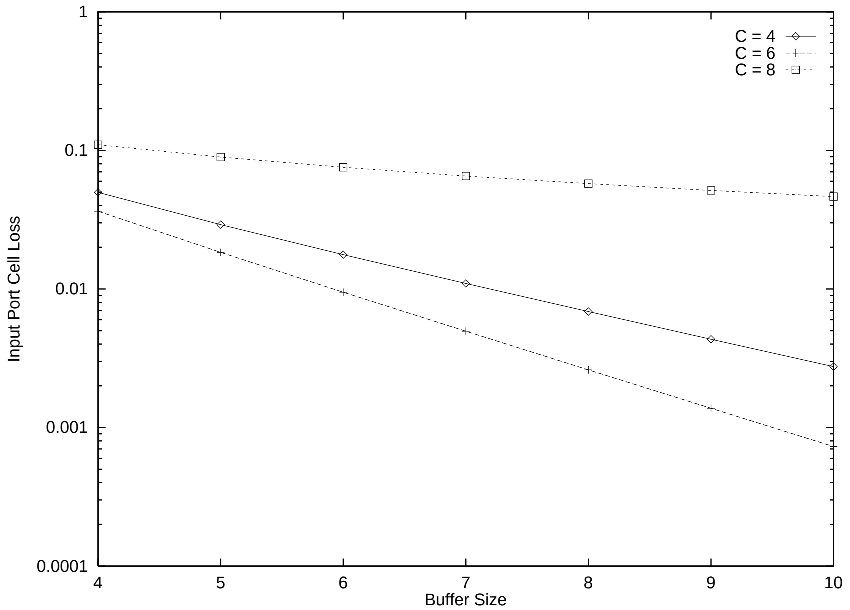

Figure 4.10: Input port cell loss probability $\Omega_{1,2}$ for $C = 4, 6, 8$ as a function of buffer size B

relative error[6] for cell loss occurred at an output port which was allocated to a dedicated frequency. The relative error observed was approximately 1×10^{-3}. The relative error increased as the number of ports sharing a single frequency increased. The worst relative error observed was 5×10^{-2}. We observed a similar behavior for the cell loss at the input ports.

[6]Relative error is taken here to mean the ratio of the absolute difference between the mean simulated value and the value calculated by our algorithm divided by the algorithm's value.

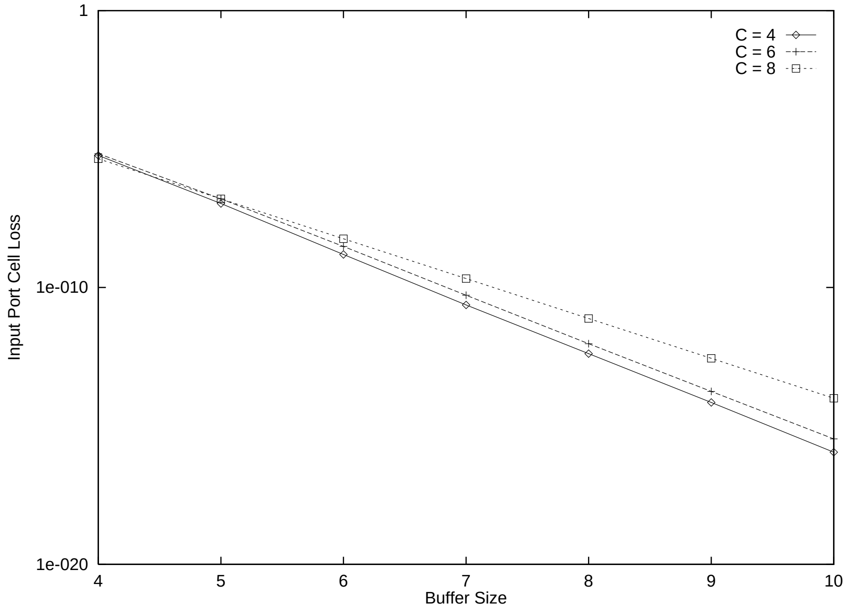

Figure 4.11: Input port cell loss probability $\Omega_{8,2}$ for $C = 4, 6, 8$ as a function of buffer size B

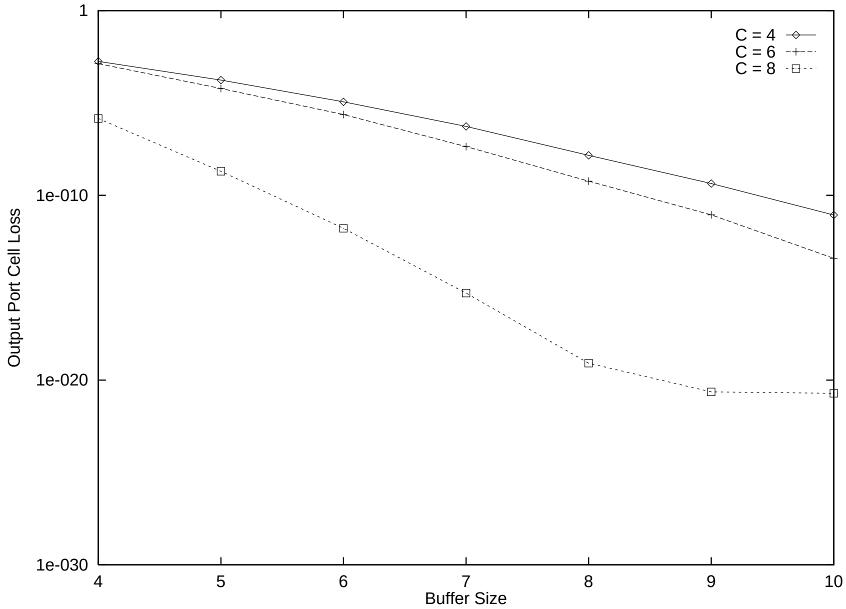

Figure 4.12: Output port cell loss probability Ω_1 for $C = 4, 6, 8$ as a function of buffer size B

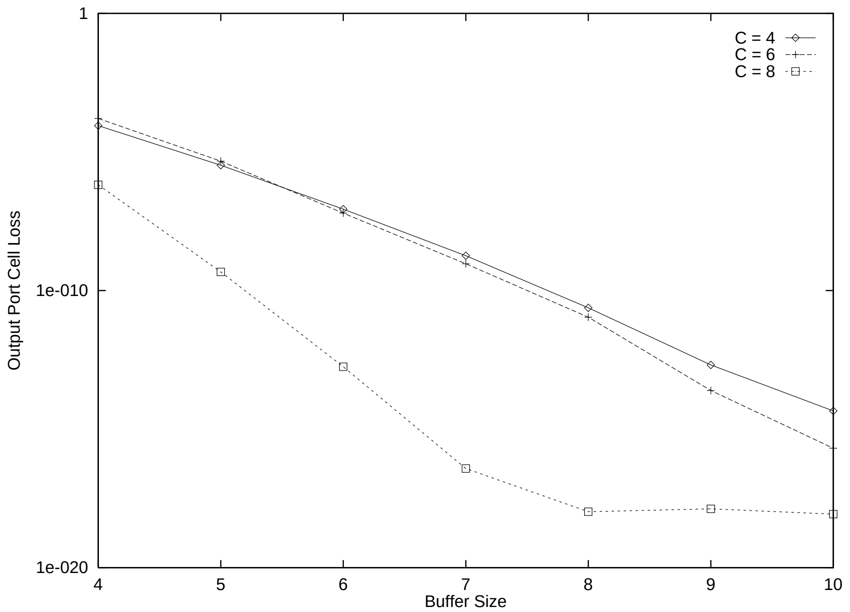

Figure 4.13: Output port cell loss probability Ω_8 for $C = 4, 6, 8$ as a function of buffer size B

Chapter 5

Analysis of a FTTR Photonic Switch Supporting Fixed Length Transmission Units

In this chapter we examine a *dual architecture* to that given in Chapter 4. This switch employs fixed frequency transmitters and tunable receivers. Sections 5.1 and 5.2 describe our system and traffic models, respectively. We then, again, develop a queueing-based decomposition algorithm to study this switch's performance in Section 5.3.

5.1 The Switch Architecture

Given the switch presented in the previous section, consider a dual architecture in which the laser at each input port is assumed to be able to transmit on only one wavelength and the receivers are tunable over all available wavelengths (i.e., a FTTR tunability configuration). This switch, shown in Figure 5.1 is the focus of the next study. It also operates in a slotted fashion and is similar to the switch of the previous section, but offers some differences due to its suitability to multicast traffic.

First, let $\lambda(i)$ denote the transmission wavelength of input port i. Further, let $\mathcal{N}_c$ denote the set of input ports which are sharing frequency c, with the cardinality of this set being denoted by N_c. Therefore, $\lambda(i)$ is the same for all $i \subseteq \mathcal{N}_c$ and $\mid \mathcal{N}_c \mid \ = N_c$.

Consider a set of output ports' receivers which are identical in terms of their

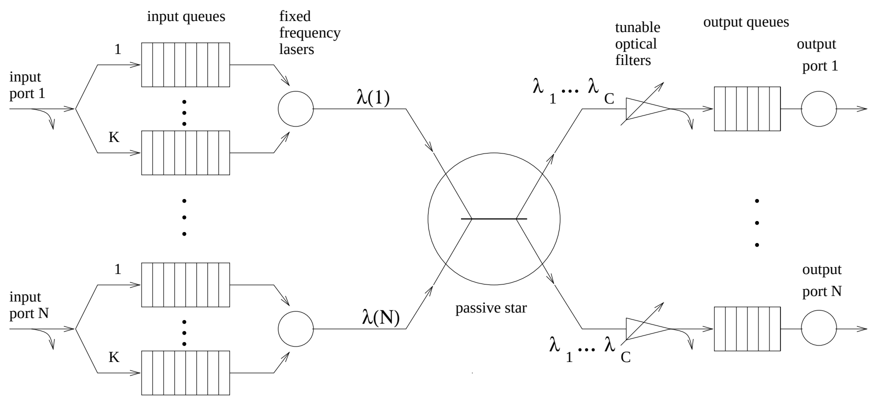

Figure 5.1: Queueing model of a switch architecture with N ports and C wavelengths

tunability characteristics (i.e., all receivers within the set are tuned to the same frequency at all times and spend identical periods tuning between frequencies). This set of receivers is defined to be a *virtual receiver*, V. Based on the notion of a virtual receiver, a *K-virtual receiver set* $V^{(K)}$, $1 \leq K \leq N$, is defined as a partitioning of the switch's N receivers into exactly K virtual receivers: $V_1^{(K)}, \cdots, V_K^{(K)}$. The work in [51] describes the process of partitioning a set of receivers into a virtual receiver set. The notation of $V^{(K)}(j)$ is used for identifying the one virtual receiver (of a K-virtual receiver set) which reflects the tuning characteristics of (physical) receiver j.

As Figure 5.1 indicates, the buffer space at each input port is assumed to be partitioned into K independent queues. Each queue k at input port i contains only cells destined for the virtual receiver $V_k^{(K)}$. $B_{ik}^{(in)}$ denotes the capacity of the queue at input port i corresponding to virtual receiver k.

As in the companion architecture, cells buffered at an input port are transmitted on a FIFO basis onto the optical medium by the port's laser. This transmission takes place on an appropriate service slot which guarantees that the cell will be correctly transmitted to all output ports serviced by the designated virtual receiver. Upon arriving at the output port, the cell is either once again placed in a finite capacity buffer or discarded (at output ports for which the cell was not originally intended). Let $B_j^{(out)}$ denote the buffer capacity of output port j. Cells arriving at an output port to find a full buffer are lost. Cells in an output buffer are also served on a FIFO basis.

5.2 Traffic Model

The arrival process to each input port of the switch is characterized by a 2-MMBP process just as in the previous chapter. The routing probabilities, however, are expressed slightly differently. Let $r_{id}^{(G)}$ denote the probability that a cell arriving to input port i will have multicast group d as its destination. We will refer to $r_{id}^{(G)}$ as the *multicast group routing probabilities*. It is assumed that the multicast group routing probabilities are known within the context of this work. The multicast group routing probabilities of successive cells are not correlated; that is, the destination group of one cell arriving to an input port does not affect the destination group of the cell behind it. This assumption is reasonable when the switch is used as part of a backbone network.

Given the definition of the virtual receiver, the probability that a cell arriving to input port i will be transmitted to virtual receiver v is:

$$r_{iv}^{(V)} = \sum_{\forall d: d \cap v \neq \emptyset} r_{id}^{(G)}, \tag{5.1}$$

Further, the probability that a cell will be buffered at output port j given that it arrived to input port i and was transmitted to virtual receiver v is:

$$r_{ivj} = \sum_{\forall d: j \subseteq d \text{ and } d \cap v \neq \emptyset} \frac{r_{id}^{(G)}}{r_{iv}^{(V)}} \tag{5.2}$$

5.3 Queueing Analysis

In this section, the queueing network shown in Figure 5.1 is analyzed. This queueing network represents the fixed-transmitter, tunable-receiver switch under study. The arrival process and the schedule for input ports accessing each of the wavelengths is assumed to be as described in the previous chapter. Also like the previous chapter, the objective of the analysis of this queueing network is to obtain the queue-length distribution in an input or output port, from which performance measures such as the cell-loss probability can be obtained.

5.3.1 Input Side Analysis

To obtain the queue length distribution of an input queue, an exact decomposition of the corresponding queueing network is shown; it is further shown, however, to not be scalable to large systems. Then an approximate method is presented and described in detail which gives accurate results.

Exact Queueing Analysis

First observe that the input side of the switch can be analyzed by decomposing it into N sub-systems corresponding to the N input ports, and analyzing each sub-system in isolation. Because of the fact that (a) the arrival processes to the various input queues are independent, (b) the way the schedule is constructed (i.e., that different inputs transmit to the same wavelength only during non-overlapping periods), and (c) the operation of the input ports is independent of the operation of output ports, this decomposition is exact. Furthermore, the sub-system corresponding to input queue i can be analyzed by defining a $(K+2)$-dimensional stochastic process $(x, y_1, \cdots, y_K, z)$, where

- x represents the arrival slot number within a frame $(x = 0, 1, \cdots, M-1)$,
- y_k indicates the number of cells in the input queue corresponding to virtual receiver k $(y_k = 0, 1, \cdots, B_{ik}^{(in)};\ k = 1, \cdots, K)$, and
- z indicates the state of the 2-MMBP describing the arrival process (studied in the previous chapter) to this queue, that is, $z = 0, 1$.

As in the dual architecture, the state space of the Markov chain grows prohibitively as $O(2M \prod_{k=1}^{K} B_{ik}^{(in)})$. In the next subsubsection, an approximate decomposition is described that can be applied to large systems.

Approximate Queueing Analysis

The main approximation is to assume that arrivals to each queue (k) of a given input port are independent and are generated by the original 2-MMBP (that which characterizes the arrival process to the input port) appropriately thinned using the routing probabilities, $r_{ik}^{(V)}$.

Given the same assumptions as in the dual architecture, the approximate queueing analysis of one of the C sub-networks (one per wavelength, as in Figure 5.2) can be found using the same techniques as in the previous chapter. The only difference in this subnetwork and the corresponding one from the previous architecture is that the division of queues within an input port, is by virtual receiver group in this case, rather than by frequency.

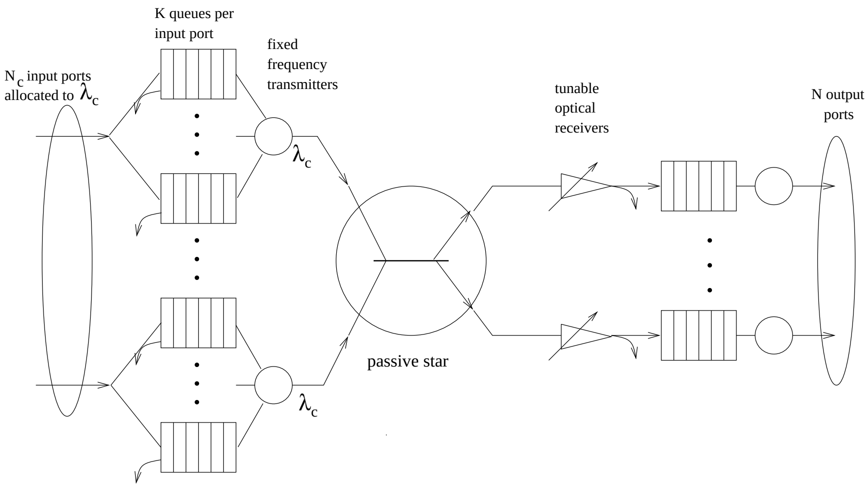

Figure 5.2: Queueing sub-network for wavelength λ_c

The Queue-Length Distribution of an Input Queue

Consider queue k of input port i from the sub-network corresponding to λ_c in isolation. This input queue receives exactly a_{ik} service slots, as shown in Figure 5.3(a). The block of a_{ik} service slots may not be aligned with the boundaries of the arrival slots. For instance, in the example shown in Figure 5.3(a), the block of a_{ik} service slots begins at the second service slot of arrival slot $x-1$, and it ends at the end of the second service slot in arrival slot $x+1$.

For each arrival slot, define $v_{ik}(x)$ as the number of service slots allocated to input queue k of port i that lie within arrival slot x. Therefore we have,

$$\sum_{x=0}^{M-1} v_{ik}(x) = a_{ik} \tag{5.3}$$

Input queue k of port i is analyzed by constructing its underlying Markov chain embedded at arrival slot boundaries. The order of events is as follows. The service (i.e., transmission) completion of a cell occurs at an instant just before the end of a service slot. An arrival may occur at an instant just before the end of an arrival slot, but after the service completion instant of a service slot whose end is aligned with the end of an arrival slot. The 2-MMBP describing the arrival process to the queue makes a state transition immediately after the arrival instant. Finally, the Markov chain is observed at the boundary of each arrival slot, *after* the state transition by the 2-MMBP. The order of these events is shown in Figure 5.3(b).

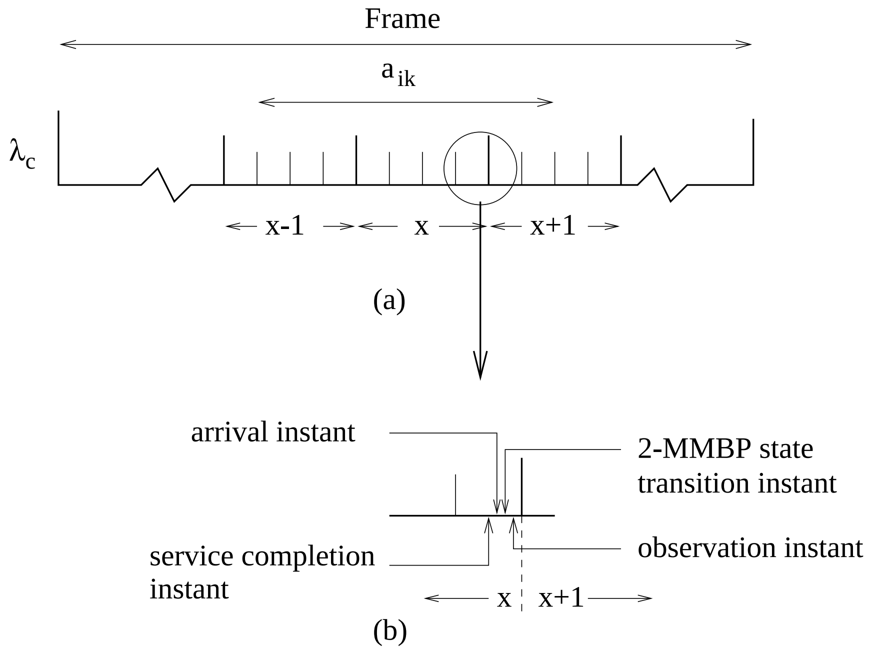

Figure 5.3: (a) Service period of input port i for virtual receiver k, and (b) detail showing the relationship among service completion, arrival, 2-MMBP state transition, and observation instants within a service and an arrival slot

The state of the input queue is described by the tuple (x, y, z), where:

- x represents the arrival slot number within a frame $(x = 0, 1, \cdots, M-1)$,
- y indicates the number of cells in the input queue $(y = 0, 1, \cdots, B_{ik}^{(in)})$, and
- z indicates the state of the 2-MMBP describing the arrival process to this queue, that is, $z = 0, 1$.

The transition probabilities out of state (x, y, z) are given in Table 5.1. As before, the next state after (x, y, z) always has an arrival slot number equal to $x \oplus 1$. In the first row of Table 5.1 we assume that the 2-MMBP makes a transition from state z to state z' (from (4.2), this event has a probability $q_i^{(zz')}$ of occurring), and that no cell arrives to this queue during the current slot (from (4.2) and (5.1), this occurs with probability $1 - \alpha_i^{(z)} r_{ik}^{(V)}$). Since at most $v_{ik}(x \oplus 1)$ cells are serviced during arrival slot $x \oplus 1$, and since no cell arrives, the queue length at the end of the slot is equal to $\max\{0, y - v_{ik}(x \oplus 1)\}$. In the second row of Table 5.1 we assume that the 2-MMBP makes a transition from state z to state z' and a cell arrives to the queue. This arriving cell cannot be serviced during this slot, and has

Table 5.1: Transition probabilities out of state (x, y, z) of the Markov chain

Current State	**Next State**	**Transition Probability**
(x, y, z)	$(x \oplus 1, \max\{0, y - v_{ik}(x \oplus 1)\}, z')$	$q_i^{(zz')}(1 - \alpha_i^{(z)} r_{ik}^{(V)})$
(x, y, z)	$(x \oplus 1, \min\{B_{ik}^{(in)}, z', \max\{0, y - v_{ik}(x \oplus 1)\} + 1\})$	$q_i^{(zz')} \alpha_i^{(z)} r_{ik}^{(V)}$

to be added to the queue. Finally, the expression for the new queue length ensures that it will not exceed the capacity $B_{ik}^{(in)}$ of the input queue.

We observe that the probability transition matrix of this Markov chain has the following block form due to the changes of the random variable x:

$$\mathbf{S}_{ik} = \begin{array}{c} \left[\begin{array}{cccccc} 0 & \mathbf{R}_{ik}(0) & 0 & 0 & \cdots & 0 \\ 0 & 0 & \mathbf{R}_{ik}(1) & 0 & \cdots & 0 \\ 0 & 0 & 0 & \mathbf{R}_{ik}(2) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \mathbf{R}_{ik}(M-2) \\ \mathbf{R}_{ik}(M-1) & 0 & 0 & 0 & \cdots & 0 \end{array} \right] \end{array} \begin{array}{c} 0 \\ 1 \\ 2 \\ \vdots \\ M-2 \\ M-1 \end{array} \tag{5.4}$$

Changes in y and z are governed by the matrices $\mathbf{R}_{ik}(x)$. There are M different $\mathbf{R}_{ik}$ matrices, one for each arrival slot x in the frame. Let us define matrices $\mathbf{X}_{ik}$ and $\mathbf{Y}_{ik}$ as follows:

$$\mathbf{X}_{ik} = r_{ik}^{(V)}\, \mathbf{A}_i\, \mathbf{Q}_i \quad \text{and} \quad \mathbf{Y}_{ik} = (\mathbf{I} - r_{ik}^{(V)}\, \mathbf{A}_i)\, \mathbf{Q}_i, \tag{5.5}$$

where $\mathbf{I}$ is the identity matrix. Then, the transition matrix $\mathbf{R}_{ik}(x)$ associated with arrival slot x can be written as:

$$
\mathbf{R}_{ik}(x) \quad = \quad \left[\begin{array}{cccccccc}
\mathbf{Y}_{ik} & \mathbf{X}_{ik} & 0 & 0 & 0 & 0 & 0 \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \;\vdots & \vdots \\
\mathbf{Y}_{ik} & \mathbf{X}_{ik} & 0 & 0 & 0 & 0 & 0 \cdots & 0 \\
0 & \mathbf{Y}_{ik} & \mathbf{X}_{ik} & 0 & 0 & 0 & 0 \cdots & 0 \\
0 & 0 & \mathbf{Y}_{ik} & \mathbf{X}_{ik} & 0 & 0 & 0 \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \;\vdots & \vdots \\
0 & 0 & \cdots & 0 & \mathbf{Y}_{ik} & \mathbf{X}_{ik} & 0 \cdots & 0
\end{array}\right]
\begin{array}{c}
0 \\ \vdots \\ v_{ik}(x \oplus 1) \\ v_{ik}(x \oplus 1) + 1 \\ v_{ik}(x \oplus 1) + 2 \\ \vdots \\ B_{ik}^{(in)}
\end{array} \tag{5.6}
$$

The structure of matrix $\mathbf{R}_{ik}(x)$ given in (5.6) can be explained as follows. Suppose that the number of cells y in the queue at the end of slot x is at most $v_{ik}(x \oplus 1)$. Since up to $v_{ik}(x \oplus 1)$ cells can be served within slot $x \oplus 1$, the number in the queue at the end of that slot will be 1 or 0, depending on whether an arrival occurred or not. This is indicated by the transitions in rows 0 through $v_{ik}(x \oplus 1)$ of matrix $\mathbf{R}_{ik}(x)$. However, if at the end of slot x we have $y > v_{ik}(x \oplus 1)$, then the number in the queue at the next transition will be $y - v_{ik}(x \oplus 1)$ (plus one if an arrival occurred). This is indicated by the transitions in rows $v_{ik}(x \oplus 1) + 1$ through B_{ik} of $\mathbf{R}_{ik}(x)$. Of course, y cannot exceed the queue capacity $B_{ik}^{(in)}$. Since the number of service slots $v_{ik}(x \oplus 1)$ depends on the particular slot $x \oplus 1$ within the frame, $\mathbf{R}_{ik}(x)$ is a function of x.

Matrix $\mathbf{R}_{ik}(x)$ is slightly different when $v_{ik}(x \oplus 1) = 0$. This is because, in this case, if the state of the input queue is such that $y = B_{ik}^{(in)}$, a new arrival will be discarded. So when $y = B_{ik}^{(in)}$, the 2-MMBP is allowed to make a transition, but regardless of whether or not an arrival is generated, the number of cells in the queue will remain equal to $B_{ik}^{(in)}$. Thus, the the last row of $\mathbf{R}_{ik}(x)$ will be: $[0\ 0\ \cdots\ 0\ \mathbf{Q}_i]$.

Transition matrix $\mathbf{S}_{ik}$ can again be solved using any of the techniques for p-cyclic Markov chains in [63, ch. 7]. The steady-state probability that the queue has y cells at the end of slot x, independent of the state of the 2-MMBP is:

$$
\pi_{ik}(x, y) \quad = \quad \sum_{z=0,1} \pi_{ik}(x, y, z) \tag{5.7}
$$

5.3.2 Output Side Analysis

We now obtain the queue length distribution of an output queue. Our analysis follows steps similar to the input side case.

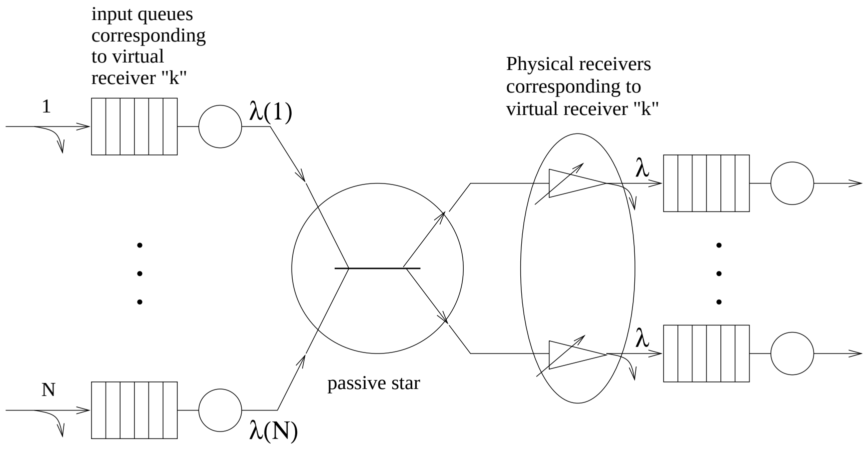

Figure 5.4: Queueing sub-network for virtual receiver $V_k^{(K)}$

Exact Queueing Analysis

Let us suppose that the (exact or approximate) queue length distribution of the input queues is known. We can then define a $(N+1)$-dimensional Markov chain, where

- x indicates the arrival slot number within the frame ($x = 0, 1, \cdots, M-1$), and
- $w_n \ \forall \ n = 1, \cdots, N$ indicates the number of cells at the n^{th} output queue ($w_n = 0, 1, \cdots, B_j^{(out)}$).

The transitions out of state $(x, w_1, \cdots, w_N)$ can be computed given the schedule and the queue length distribution of the input ports. However, for realistic switch dimensions, this method will lead to a state space explosion since the total number of states is of the order of $M \times \prod_{j=1}^{N} B_j^{(out)}$.

Approximate Queueing Analysis

Consider a virtual receiver $V_k^{(K)}$. Since this virtual receiver receives traffic exclusively from the N input queues (exactly 1 queue at each input port) corresponding to this virtual receiver, the switch may be decomposed by virtual receiver in order to analyze the output side. This decomposition is shown in Figure 5.4. In considering virtual receiver k, it is implicit in discussing input port i that only the queue k of the input port is relevent; therefore this specification is not explicitly denoted in the following discussion.

An interesting aspect of the departure process from the input queues is that, for each frame, during the sub-period a_{ik}, we only have departures from ith input queue of the decomposed system. This period is then followed by a gap g_{ik} during which no departure occurs. This cycle repeats for the next input queue. Thus, in order to characterize the overall departure process offered as the arrival process to these output queues, it suffices to characterize the departure process from each input queue, and then combine them. (We note that this overall departure process is quite different from the typical process of constructing the superposition of a number of departures into a single stream, where, at each slot, more than one cell may be departing.)

However, the arrival processes to the various receivers governed by the same virtual receiver are not independent. Specifically, if j and j' are two receivers such that $V^{(K)}(j) = V^{(K)}(j') = V$, and there is a transmission from input port i to multicast group V, in a given service slot, the cell will necessarily be intended for exactly one multicast group. Therefore, depending on the participation of j and j' in the multicast groups which span the virtual receiver V, the probabilities of receiving the cell at output ports j and j' cannot be considered to be completely independent. As in the input side case, we will nevertheless make the assumption that these arrival processes are indeed independent, and that each is an appropriately thinned (based on the routing probabilities) version of the departure process from the input queues.

The Queue-Length Distribution of an Output Queue

As in the previous section, we obtain the queue-length distribution of output port j at arrival slot boundaries. Recall that an arrival slot to an input queue is equal to a departure slot from an output queue. Also, arrival and departure slots are synchronized. Therefore, during an arrival slot x, a cell may be transmitted to the outgoing link from the output queue. However, during slot x, there may be several arrivals to the output queue from the input queues.

Let (x, w) be the state associated with output port j, where

- x indicates the arrival slot number within the frame ($x = 0, 1, \cdots, M-1$), and
- w indicates the number of cells at the output queue ($w = 0, 1, \cdots, B_j^{(out)}$).

We assume the following order of events. A cell will begin to depart from the output queue at an instant immediately after the beginning of an arrival slot and the departure

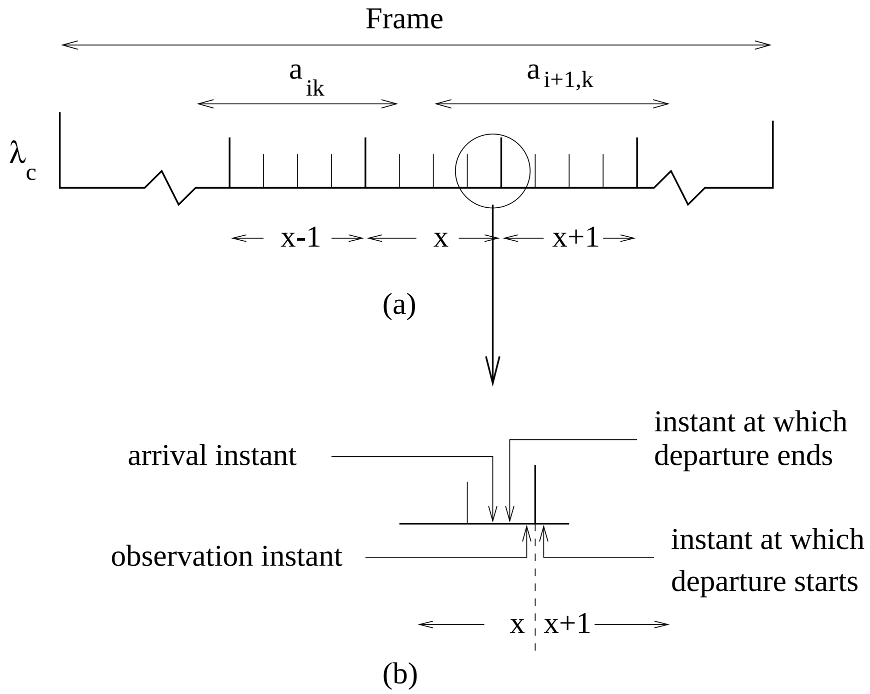

Figure 5.5: (a) Arrivals to output port j from input ports i and $i+1$, and (b) detail showing the relationship of departure, arrival, and observation instants

will be completed just before the end of the slot. A cell from an input port arrives at an instant just before the end of a service slot, but before the end-of-departure instant of an arrival slot whose end is aligned with the end of the service slot. Finally, the state of the queue is observed just before the end of an arrival slot and after the arrival associated with the last service slot has occurred (see Figure 5.5(b)).

Let $u_j(x)$ be the number of serviced service slots of any input queue on corresponding to virtual receiver $V^{(K)}(j)$ within arrival slot x. We have that

$$u_j(x) = \sum_{i=1}^{N} v_{ik}(x) \tag{5.8}$$

where $v_{ik}(x)$ is as defined in (5.3). Quantity $u_j(x)$ represents the maximum number of cells that may arrive to output port j within slot x. In the example of Figure 5.5(a) where we show the arrival slots during which cells from input ports i and $i+1$ may arrive to output port j, we have: $u_j(x-1) = v_{ik}(x-1) = 4$, $u_j(x) = v_{ik}(x) + v_{i+1,c}(x) = 1 + 2 = 3$, and $u_j(x+1) = v_{i+1,c}(x+1) = 4$.

Observe now that (a) at each state transition x advances by one (modulo-M), (b)

Table 5.2: Transition probabilities out of state (x, w) of the Markov chain

Current State	Next State	Transition Probability
$(x, 0)$	$(x \oplus 1, \min\{B_j^{(out)}, s\})$, $0 \leq s \leq u_j(x \oplus 1)$	$\sum_{s_1+\cdots+s_N=s} \prod_{i=1}^{N} L_i(s_i \mid x)$
$(x, w), w > 0$	$(x \oplus 1, \min\{B_j^{(out)}, w+s\} - 1)$, $0 \leq s \leq u_j(x \oplus 1)$	$\sum_{s_1+\cdots+s_N=s} \prod_{i=1}^{N} L_i(s_i \mid x)$

exactly one cell departs from the queue as long as the queue j not empty, (c) a number $s \leq u_j(x \oplus 1)$ of cells may be transmitted from the input ports to output port j within arrival slot $x \oplus 1$, and that (d) the queue capacity is $B_j^{(out)}$. Then, the transition probabilities out of state (x, w) for this Markov chain are given in Table 5.2.

In Table 5.2, $L_i(s_i \mid x)$ is the probability that input port i transmits s_i cells to output port j given that the system is at the end of arrival slot x (in other words, it is the probability that s_i cells are transmitted within slot $x \oplus 1$[1]. The "thinning" of the arrival processes using the r_{ikj} routing probabilities discounts the correlation between arrival streams and is the crux of the approximation for the output side of the switch. The error introduced by this approximation will be discussed later in this work.

Define $\pi_{ik}(y \mid x)$ as the conditional probability of having y cells at input queue k of port i given that the system is observed at the end of slot x:

$$\pi_{ik}(y \mid x) \quad = \quad \frac{\pi_{ik}(x, y)}{\pi_{ik}(x)} \quad = \quad M \ \pi_{ik}(x, y) \tag{5.9}$$

Then, for $r_{ikj} < 1$, the probability $L_i(s_i \mid x)$ is given by

$$L_i(s_i \mid x) = \begin{cases} 0, & s_i > v_{ik}(x \oplus 1) \\ \sum_{y=s_i}^{B_{ik}^{(in)}} \pi_{ik}(y \mid x) \begin{pmatrix} \min\{y, v_{ik}(x \oplus 1)\} \\ s_i \end{pmatrix} \times & s_i \leq v_{ik}(x \oplus 1) \\ \qquad (r_{ikj})^{s_i} (1 - r_{ikj})^{\min\{y, v_{ik}(x \oplus 1)\} - s_i}, & \end{cases} \tag{5.10}$$

Expression (5.10) can be explained by noting that input port i will transmit s_i cells to output port j during arrival slot $x \oplus 1$ if (a) $v_{ik}(x \oplus 1) \geq s_i$, (b) input port i has $y \geq s_i$ cells

[1]Since in most cases only one or two input ports will transmit to the same channel within an arrival slot (refer again to Figure 2.2), the summation and product in the expression in the last column of Table 5.2 does not necessarily run over all N values of i, only over one or two values of i. Thus, this expression can be computed very fast, not in exponential time as implied by the general form presented in the table.

in its queue for virtual receiver k at the beginning of the slot (equivalently, at the end of slot x), and (c) exactly s_i of $\min\{y, v_{ik}(x \oplus 1)\}$ cells that will be transmitted by this queue in this arrival slot are actually intended for output j.

If $r_{ikj} = 1$, in which case j is the only output port for which $V^{(K)}(j) = V_k^{(K)}$, the expression for $L_i(s_i \mid x)$ must be modified as follows:

$$L_i(s_i \mid x) = \begin{cases} 0, & s_i > v_{ik}(x \oplus 1) \\ \pi_{ik}(s_i \mid x), & s_i < v_{ik}(x \oplus 1) \\ \sum_{y=s_i}^{B_{ik}^{(in)}} \pi_{ik}(y \mid x), & s_i = v_{ik}(x \oplus 1) \end{cases} \tag{5.11}$$

Expressions (5.10) and (5.11) are based on the assumption that $v_{ik}(x \oplus 1) < B_{ik}^{(in)}$ which we believe is a reasonable one. In the general case, quantity $v_{ik}(x \oplus 1)$ in both expressions must be replaced by $\min\{v_{ik}(x \oplus 1), B_{ik}^{(in)}\}$.

The transition matrix $\mathbf{T}_j$ of the Markov chain defined by the evolution of the state (x, w) of output queue j has the following form, which is similar to that of matrix $\mathbf{S}_{ik}$ given by (5.4):

$$\mathbf{T}_j = \begin{array}{c} \left[\begin{array}{cccccc} 0 & \mathbf{U}_j(0) & 0 & 0 & \cdots & 0 \\ 0 & 0 & \mathbf{U}_j(1) & 0 & \cdots & 0 \\ 0 & 0 & 0 & \mathbf{U}_j(2) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \mathbf{U}_j(M-2) \\ \mathbf{U}_j(M-1) & 0 & 0 & 0 & \cdots & 0 \end{array}\right] \end{array} \begin{array}{c} 0 \\ 1 \\ 2 \\ \vdots \\ M-2 \\ M-1 \end{array} \tag{5.12}$$

$\mathbf{U}_j(x)$ is a $(B_j^{(out)}+1) \times (B_j^{(out)}+1)$ matrix that governs changes in random variable w of the state of the output queue The elements of this matrix can be determined using Table 5.2 and expression (5.10) or (5.11). Since $L_i(s_i \mid x)$ depends on $v_{ik}(x)$ and $u_j(x)$, $\mathbf{U}_j(x)$ also depends on x, the slot number within the frame.

We observe that $\mathbf{T}_j$ also defines a *p-cyclic* Markov chain. We have used the LU decomposition method as prescribed in [63] to obtain $\pi_j(x, w)$, the steady-state probability that output queue j has w cells at the end of slot x.

5.3.3 Summary of the Decomposition Algorithm

Below we summarize our approach to analyzing the switch architecture of Figure 5.1. We assume that quantities $\{a_{ik}\}$, the corresponding schedule (see [60]), and the multicast group routing probabilities $(r_i^M m)$ are given.

1. For each arrival slot x, use the schedule and expressions (5.3) and (5.8) to compute the quantities $v_{ik}(x)$ and $u_k(x)$, $\forall i \in \{1, \cdots, N\}$; $\forall k \in \{1, \cdots, K\}$.

2. For each queue of input port i, construct the transition probability matrix $\mathbf{S}_{ik}$ from (4.2), (5.1), (5.4), (5.5), and (5.6). Solve this matrix and use (5.7) to obtain the steady-state probability $\pi_{ik}(x, y)$ that input queue k has y cells at the end of the x-th slot of the frame.

3. For each output port j, use $\pi_{ik}(x, y)$ $(\forall i \in \{1, \cdots, N\}, k = V^{(K)}(j))$ derived in Step 2, and (5.10)-(5.11) to construct the transition matrix $\mathbf{T}_j$ given by (5.12). Solve the matrix as in Step 2 to obtain $\pi_j(x, w)$, the steady-state probability that port j has w cells in its queue at the end of slot x.

Note that the complexity of this approach is dominated by Step 2. For each of the N input queues we have to solve a matrix of dimensions $\left[2M(B_{ik}^{(in)} + 1)\right] \times \left[2M(B_{ik}^{(in)} + 1)\right]$, where M is the length of the schedule (in arrival slots) and $B_{ik}^{(in)}$ is the capacity of the respective queue. (Inverting a $K \times K$ matrix takes time $O(K^3)$, although we can take advantage of the fact that the matrix is sparse to solve for the queue-length distributions at a significantly faster rate.) Thus, in the worst case, the overall complexity of our algorithm is $O(NM^3B^3)$, where $B = \max_i\{B_{ik}^{(in)}\}$.

Since it is now seen that the algorithm for deriving occupancy probabilities is extremely similar to that for the TTFR case, the cell loss probability calculations are left unstated as a trivial extension to those shown previously.

Chapter 6

Analysis of a TTFR Photonic Switch Supporting Variable Length Transmission Units

In the previous chapters, two photonic switches for transporting fixed length transport units (e.g., ATM cells) were presented and analyzed for cell loss. It is reasonable, however, to assume that some network architectures will require the additional switching capacity which photonic technology affords but will not employ fixed length transport units (packets). In this chapter, we analyze the performance of a photonic switch, similar to that presented in Chapter 3, which supports varying packet lengths.

We begin by describing the switch's architecture and the traffic model used in this work. The buffer occupancy analysis for both the input and output sides is presented in Section 6.3; Section 6.4 presents loss calculations for both the input and output sides of the switch. We then conclude this chapter with numerical results and interpretations.

6.1 The Switch Architecture

Consider a switch with N input ports and N output ports interconnected through a broadcast passive star that can support $C \leq N$ wavelengths $\lambda_1, \cdots, \lambda_C$ (see Figure 6.1). The switch is similar to the one studied in Chapter 3 with certain specific modifications made to accommodate variable length packets.

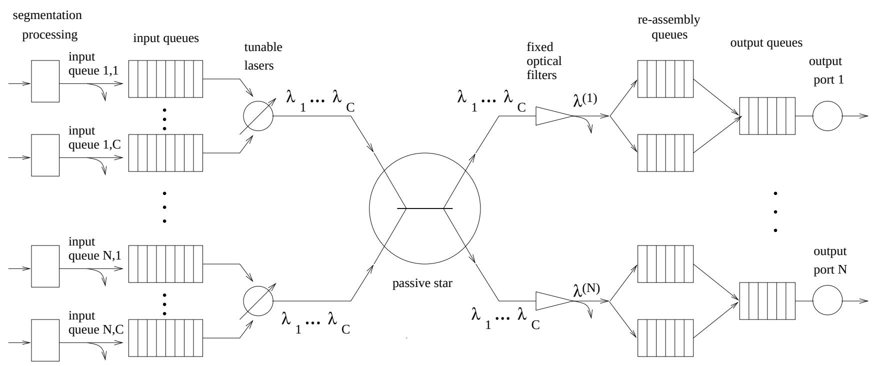

Figure 6.1: Queueing model of the switch architecture for variable packet lengths

The switch is composed of four stages:

1. the segmentation stage;
2. the transmission (or input buffering) stage;
3. the re-assembly stage; and,
4. the output buffering stage.

The switch as a whole may accommodate packets of varying length up to an arbitrary, but prespecified, maximum length. A packet which is received by the switch is decomposed upon arrival to the "segmentation stage" into *segments* of fixed length. The last segment can be padded, if necessary; we assume for this analysis that it is. These segments are enqueued at the "input buffering stage" before being transmitted through the switch over one of its C channels. As a packet may require several arrival slots to be completely received, segments are generated and transmitted from the "segmentation stage" to the "input buffering" stage as they are accumulated from the input link. Therefore, it is possible for part of a packet to exist in the form of segments in the "input buffering" stage while the remainder of the packet has not yet been received from the input link. Segments are buffered in a finite capacity queue in the "input buffering" stage, if the queue is not full. If the queue is full, the segment is dropped along with all other segments of the packet already in the queue. Those segments of the packet that have not arrived yet are also dropped upon arrival[1]. Let $B_{ic}^{(in)}$ denote the buffer capacity of the queue corresponding to channel c at input port i.

[1]This phenomena is reminiscent of the "Partial Packet Discard" feature of some ATM switches.

Upon arriving at the output port, the segment is buffered in an appropriate finite capacity queue in the "re-assembly" stage. Queues in an output port's "re-assembly" stage are dedicated per input port[2] and are provisioned to be long enough to store a packet of the maximum length. These queues accumulate segments of packets until an entire packet has been completely received by the intended output port. It should be noted that, by the nature of the system described so far, the segments will necessarily be received in order and a new packet will not begin to be buffered in the "re-assembly" stage until all segments of the previous packet have been received. Furthermore, since this buffer can accommodate a packet of the maximum specified size, no losses will occur due to a lack of space in this queue. Once all segments of a packet have been received at the "re-assembly" queue, the packet is instantaneously transferred *en masse* to the "output buffering" stage. If adequate space is not available in the "output buffering" stage for the arriving packet, the entire packet is lost. Let $B_j^{(out)}$ denote the buffer capacity of output port j in terms of packet segments[3].

6.2 Traffic Model

The arrival process to each input queue of the router is comprised of a stream of packets which conform to an arbitrary (but non-varying) packet length distribution bounded by a minimum and a maximum size. Since packets must be broken into fixed length segments when transferred from the "segmentation stage" to the "input buffering" stage (with the final segment being padded if necessary), the probability distribution of a packet arriving to input queue c of port i having a size of s segments is denoted by $f_{ic}(s)$. In the previous chapters we assumed that traffic would be generated by a (two state) MMBP. However, a MMBP cannot accurately model the correlated arrivals of a variable length packet previously described as having a fixed distribution. We therefore assume a *state machine* for describing the arrivals to an input queue. The means by which this state machine accommodates the correlated arrivals of segments where the MMBP would not is discussed below. By using a separate state machine to generate packet arrivals for a single input queue, we assume that

[2]In fact, the number of queues a given output port's "re-assembly" stage only need be equal to the number of input ports which may transmit to the given output port.

[3]Obviously, the discussion of any internal overhead required to achieve the functionality specified by this architecture is not specifically described here. Suffice it to say that this overhead can easily be specified and would require a minor modification to the $\frac{N}{C}$ factor between the internal and external channel speeds.

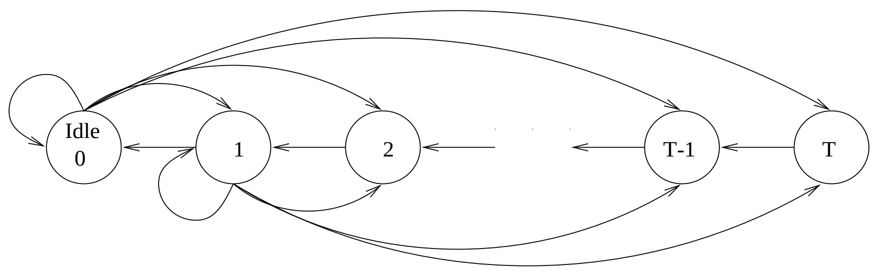

Figure 6.2: State machine for packet segment arrival process

any significant correlation between arrivals to different queues of the same port (which are related by virtue of the fact that the arrivals are initially multiplexed on the same input channel) can be captured by the sufficient description of the flexible model presented in this section.

Consider the state machine shown in Figure 6.2. First, assume that the governing packet length distribution calls for a minimum packet size to be m segments per packet (m is the ratio of switch ports, N, to the number of available channels, C) and the maximum is assumed to be T segments per packet. The minimum packet size assumption is made only to simplify the following analysis and may be relaxed if necessary; the maximum packet size can be specified as an arbitrary value (greater than the minimum). The arrival slots during which no segment arrives to the input queue are represented by the "0" ("Idle") state at the far left. From that point, the arrival process may transition at discrete intervals (corresponding to the arrival slots) to any of the states based upon the arrival of a packet and the packet's length in terms of packet segments. The state identifiers indicate the number of segments which have yet to arrive to the "input buffering" stage of the switch. Therefore, after every arrival slot, the state decreases by one. After the last segment has arrived (i.e., at the instant at which the state machine is leaving the state denoted "1"), the system will either become idle (indicating that no packet is being transmitted on the input link or that a packet not intended for the particular input queue is being transmitted) or another packet of arbitrary length may be immediately received[4].

Based on this description, the packet arrival process may be characterized as fol-

[4]Figure 6.2 is not to be construed as depicting all of the flows which must have probabilities greater than 0.0 of occurring; instead, these flows are the only ones which *may* be greater than 0.0. It is still possible for some of the shown flows to be equal to 0.0. In fact, given the restriction that a packet is at least m segments in this work, the flows from states "0" and "1" to all of the states numbered less than m will necessarily have probabilities equal to 0.0. Figure 6.2 is meant to depict an arbitrary packet length distribution.

lows. There is a geometrically distributed idle period in which no arrivals occur followed by the arrival of a packet. The length of the packet, expressed in segments, is arbitrarily distributed between 1 and T. After the last segment has arrived, another packet arrives immediately or the process becomes idle and repeats. The time for a segment to arrive is equal to an arrival slot. Specifically, the arrival process to queue $(i, c),\ i = 1, \cdots, N,\ c = 1, \cdots, C$, is characterized by the transition probability matrix $\mathbf{Q}_{ic}$

$$\mathbf{Q}_{ic} = \begin{bmatrix} q_{ic}^{(0,0)} & q_{ic}^{(0,1)} & q_{ic}^{(0,2)} & \cdots & q_{ic}^{(0,T-1)} & q_{ic}^{(0,T)} \\ q_{ic}^{(1,0)} & q_{ic}^{(1,1)} & q_{ic}^{(1,2)} & \cdots & q_{ic}^{(1,T-1)} & q_{ic}^{(1,T)} \\ 0 & q_{ic}^{(2,1)} = 1.0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & q_{ic}^{(3,2)} = 1.0 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & q_{ic}^{(T,T-1)} = 1.0 & 0 \end{bmatrix} \tag{6.1}$$

and the arrival probability matrix $\mathbf{A}_{ic}$

$$\mathbf{A}_{ic} = \begin{bmatrix} 0 & 0 & 0 & \cdots \\ 0 & 1 & 0 & \cdots \\ \vdots & \vdots & \ddots & \ddots \\ 0 & \cdots & 0 & 1 \end{bmatrix} \tag{6.2}$$

In (6.1), $q_{ic}^{(k,l)}, k, l = 0, \cdots, T$ is the probability that the state machine will make a transition to state l, given that it is currently at state k. Obviously, $\sum_l q_{ic}^{(k,l)} = 1,\ \forall k$. Transitions between states of Q_{ic} occur only at the boundaries of *external* slots.

There exist a number of interesting relationships between $\mathbf{Q}_{ic}$ and the packet size distribution, f_{ic}. First, it can be easily shown that the following expression is true:

$$f_{ic}(s) = \sum_{k=0,1} \frac{q_{ic}^{(k,s)}}{1 - q_{ic}^{(k,0)}} \times \frac{Pr(\text{state } k)}{Pr(\text{state } 0) + Pr(\text{state } 1)} \tag{6.3}$$

The probabilities $Pr(\text{state } 0)$ and $Pr(\text{state } 1)$ may be found by solving for the steady state occupancy probability of the arrival process' state transition matrix, $\mathbf{Q}_{ic}$.

More importantly, several expressions based on the arrival processes are necessary for the analysis presented below. First, the probability of the arrival process occupying a particular state given that the system is in (external) slot x is of significance. This value may be found by first formulating the transition probability matrix at the boundaries of the repeating schedule's frame (i.e., on either side of M arrival slots), denoted $\mathbf{K}_{ic}$.

$$\mathbf{K}_{ic} = \mathbf{Q}_{ic}^{M} \tag{6.4}$$

It can be shown that this process is, in fact, a Markov chain and a steady state occupancy solution can be determined. This solution is denoted $\psi_{ic}(s \mid 0)$ and is the probability of the arrival process occupying state s at the boundaries of the frame (i.e., before arrival slot 0 of the frame). The same solution may be found for other arrival slot boundaries by using Equation (6.5).

$$\psi_{ic}(s \mid x) = \psi_{ic}(s \mid x \ominus 1)\mathbf{Q}_{ic} \tag{6.5}$$

6.3 Queueing Analysis

In this section the queueing network shown in Figure 6.1, which represents the tunable-transmitter, fixed-receiver switch under study, is analyzed. The access of the input ports to the wavelengths and the arrival process to each queue is described in Chapter 2. This queueing network is analyzed in order to obtain the queue-length distribution in an input or output port, from which performance measures such as the probability of segment and packet loss can be obtained.

6.3.1 Exact Input Queue Analysis

We can analyze the sub-system corresponding to input queue i by defining a multi-dimensional stochastic process containing the following information:

- the arrival slot number within a frame
- the number of packets and the packet sizes (in terms of segments);
- the state of the arrival process for the given queue (i.e., which segment of the most recently arriving packet is being buffered); and,
- the state of the server for the given queue (i.e., which segment of the oldest packet enqueued is being served).

It quickly becomes obvious that such a Markov chain cannot be practically analyzed.

6.3.2 Approximate Input Queue Analysis

In order to make the input queue's analysis more manageable, we reduce the most significant source of complexity (the number of packets and their lengths) and only keep track of the number of segments enqueued on the input side of the switch. We also simplify the analysis by not tracking the state of the server described previously. We analyze input queue i by constructing its underlying Markov chain embedded at arrival slot boundaries; our Markov chain consists of the tuple (x, y, z) such that

- x represents the arrival slot number within a frame ($x = 0, 1, \cdots, M-1$),
- y indicates the number of segments in the input queue ($y = 0, 1, \cdots, B_{ic}^{(in)}$), and
- z indicates the state of the state machine describing the arrival process to this queue ($z = -(T-1), \cdots, -1, 0, 1, \cdots, T$).

For each arrival slot, define $v_{ic}(x)$ as the number of service slots allocated to input queue i during arrival slot x. Obviously we have,

$$\sum_{x=0}^{M-1} v_{ic}(x) = a_{ic} \tag{6.6}$$

The order of events is as follows. The service (i.e., transmission) completion of a segment occurs at an instant just before the end of an service slot. An arrival may occur at an instant just before the end of an arrival slot, but after the service completion instant of a service slot whose end is aligned with the end of an arrival slot. The arrival process to the queue makes a state transition immediately after the arrival instant. Finally, the Markov chain is observed at the boundary of each arrival slot, *after* the state transition by the state machine. The order of these events is shown in Figure 6.3(b).

Modulo-M addition is still denoted by $\oplus$, where M is the number of arrival slots per frame; $\mathbf{I}_{f(x)}$ is an indicator function which evaluates to "1" if the boolean condition "$f(x)$" is true and "0" otherwise. The transition probabilities out of state (x, y, z) are given in Table 6.1.

It should be noted that the definition of z does not follow exactly the state machine shown in Figure 6.2. The state machine is modified in order to account for a discarded packet which is partially enqueued. Specifically, consider the case in which the buffer receives a segment while full. In this case, not only are the segments of the packet which are already

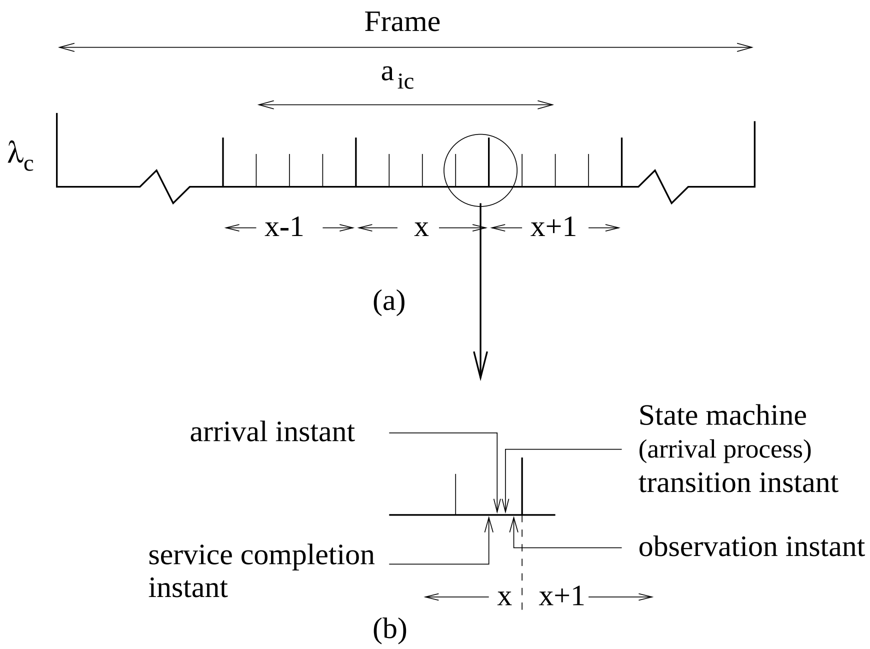

Figure 6.3: (a) Service period of input port i on channel λ_c, and (b) detail showing the relationship among service completion, arrival, arrival process state transition, and observation instants within an service slot and an arrival slot

enqueued discarded, but the remaining segments of the packet which have not yet been received must also be discarded. In order to preserve the inter-packet arrival distribution, an additional set of states must be added to the arrival process' state description to indicate when a segment from a discarded packet is being received (and, hence, itself discarded). These states are identified by the negative of the states shown in Figure 6.2. Therefore, if the arrival process is in state $t, 2 \leq t \leq T$, and the queue discards the segment as described, the process will transition to state $-(t-1)$, not state $t-1$. Figure 6.4 illustrates the progression of this variable.

Note that, the next state after (x, y, z) always has an arrival slot number equal to $x \oplus 1$. In the first row of Table 6.1, we assume that the state machine makes a transition from state z to state z' (from (6.1), this event has a probability $q_i^{(zz')}$ of occurring), and a segment arrives and is buffered by the queue. This event can only occur if z' is positive (see Figure 6.4) *and* either $v_{ic}(x) > 0$ or $y < B_{ic}$. The latter conditions are imposed to ensure that the new queue length will not exceed the capacity $B_{ic}^{(in)}$ of the input queue[5]. This

[5]Due to the nature of the system, segment loss can only occur if both of these conditions are not true *and* an arrival occurs.

Table 6.1: Transition probabilities out of state (x, y, z) of the Markov chain

Current State	**Next State**	**Transition Probability**
(x, y, z)	$(x \oplus 1,$ $\max\{0, y - v_{ic}(x \oplus 1)\} + 1, z')$	$q_{ic}^{(zz')} \mathbf{I}_{v_{ic}(x)>0 \text{ or } y<B_{ic}} \times$ $\mathbf{I}_{(z>1 \ \& \ z'=(z-1)) \text{ or } (z=1)}$
(x, y, z)	$(x \oplus 1,$ $\max\{0, y - v_{ic}(x \oplus 1)\}, z')$	$q_{ic}^{(\lvert z \rvert \lvert z' \rvert)} \times$ $\mathbf{I}_{(z=0,-1) \text{ or } ((z<-1) \ \& \ (z'=z+1))}$
(x, y, z)	$(x \oplus 1, B_{ic} - s, z')$	$f_{ic}(s+z) q_{ic}^{(z \lvert z' \rvert)} \mathbf{I}_{v_{ic}(x)=0 \ \& \ y=B_{ic}} \times$ $\mathbf{I}_{(z>1 \ \& \ z'=-(z-1)) \text{ or } (z=1 \ \& \ z' \geq 0)}$

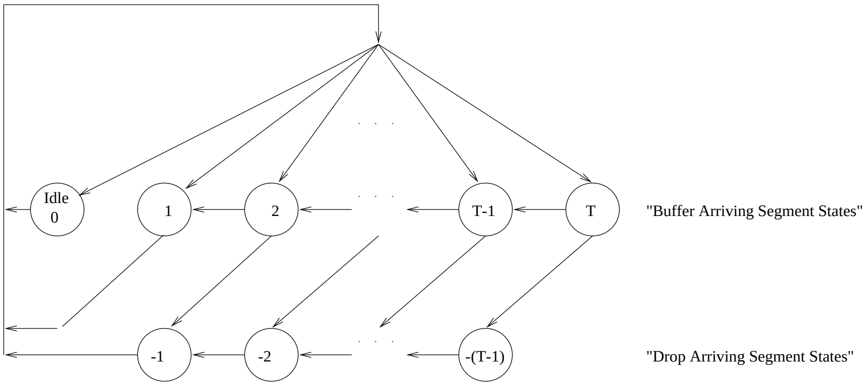

Figure 6.4: State machine for arrival state accounting for buffering and dropping of packet segments

arriving segment cannot be serviced during this slot, and has to be added to the queue. Since at most $v_{ic}(x \oplus 1)$ segments are serviced during arrival slot $x \oplus 1$, and since exactly one segment arrives, the queue length at the end of the slot is equal to $\max\{0, y - v_{ic}(x \oplus 1) + 1\}$.

In the second row of Table 6.1, we assume that the arrival process makes a transition from state z to state z' and that no segment is buffered at this queue during the current slot, but that the queue does not overflow at this time. This event will occur unconditionally only if the buffer has already overflowed or the source is idle (i.e., $z \leq 0$, see (6.2)). Again, at most $v_{ic}(x \oplus 1)$ segments are serviced during arrival slot $x \oplus 1$, resulting in the queue length at the end of the slot being $\max\{0, y - v_{ic}(x \oplus 1)\}$.

Finally, we assume that a segment arrives to the input buffering queue causing it to overflow. This event occurs if and only if the queue has not yet overflowed, the buffer is full, and the buffer receives no service during the arrival slot (i.e., $y = B_{ic}$, $z > 0$, and $v_{ic}(x) = 0$). In this case, the arrival process transitions to the appropriate state reflecting that future segments of this packet are to be dropped. Also, the buffer will lose s segments of the arriving packet which have already been buffered, provided that the packet size was $s + z$ segments.

We observe that the probability transition matrix of this Markov chain has the following block form:

$$\mathbf{S}_{ic} = \begin{array}{c} \left[\begin{array}{cccccc} 0 & \mathbf{R}_{ic}(0) & 0 & 0 & \cdots & 0 \\ 0 & 0 & \mathbf{R}_{ic}(1) & 0 & \cdots & 0 \\ 0 & 0 & 0 & \mathbf{R}_{ic}(2) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \mathbf{R}_{ic}(M-2) \\ \mathbf{R}_{ic}(M-1) & 0 & 0 & 0 & \cdots & 0 \end{array} \right] \end{array} \begin{array}{c} 0 \\ 1 \\ 2 \\ \vdots \\ M-2 \\ M-1 \end{array} \qquad (6.7)$$

This block form is due to the fact that at each transition instant (i.e., at each arrival slot boundary), the random variable x changes to $x \oplus 1$. Changes in the other two random variables, y and z, are governed by the matrices $\mathbf{R}_{ic}(x)$. There are M different $\mathbf{R}_{ic}$ matrices, one for each arrival slot x in the frame.

The $\mathbf{R}_{ic}$ matrices may be formulated as follows. Let us define matrices $\mathbf{X}_{ic}(\cdot \mid x, y)$ and $\mathbf{Y}_{ic}$ as follows:

$$\mathbf{X}_{ic}(\cdot \mid x, y) = \tilde{\mathbf{A}}_{ic}\, \tilde{\mathbf{Q}}_{ic}(\mathbf{I}_{v_{ic}(x)=0 \text{ and } y=B_{ic}}) \quad \text{and} \quad \mathbf{Y}_{ic} = (\mathbf{I} - \tilde{\mathbf{A}}_{ic})\, \tilde{\mathbf{Q}}_{ic}(0), \qquad (6.8)$$

where $\mathbf{I}$ is the identity matrix and $\tilde{\mathbf{A}}_{ic}$ and $\tilde{\mathbf{Q}}_{ic}(\cdot)$ are given in (6.9), (6.10), and (6.11)[6].

[6]The matrices $\mathbf{X}_{ic}(\cdot \mid x, y)$ and $\mathbf{Y}_{ic}$ are the arrival process' state transition probability matrices given that an arrival is (is not) accepted by the input buffering queue, respectively. Given that definition, it becomes obvious why only $\mathbf{X}_{ic}$ (and not $\mathbf{Y}_{ic}$) is dependent on the evaluation of the given indicator function. $\mathbf{X}_{ic}(\cdot \mid x, y)$ will be denoted as simply $\mathbf{X}_{ic}(\cdot)$ for the sake of clarity for the remainder of the paper.

$$\tilde{\mathbf{A}}_{ic} = \begin{array}{cc} & \begin{array}{ccc|ccc} -(T-1) & \cdots & -1 & 0 & \cdots & T \end{array} \\ \begin{array}{l} -(T-1) \\ \vdots \\ -1 \\ \hline 0 \\ \vdots \\ T \end{array} & \left(\begin{array}{c|c} 0 & 0 \\ \hline 0 & \mathbf{A}_{ic} \end{array} \right) \end{array} \tag{6.9}$$

$$\tilde{\mathbf{Q}}_{ic}(0) = \begin{array}{cc} \downarrow_k \quad {}^{l} \rightarrow & \begin{array}{ccc|cccc} -(T-1) & \cdots & -1 & 0 & 1 & \cdots & T \end{array} \\ \begin{array}{l} -(T-1) \\ \vdots \\ -2 \\ \hline -1 \\ \hline 0 \\ \vdots \\ T \end{array} & \left(\begin{array}{c|c} q_{ic}^{|k||l|} & 0 \\ \hline 0 & q_{ic}^{|k||l|} \\ \hline 0 & \mathbf{Q}_{ic} \end{array} \right) \end{array} \tag{6.10}$$

and

$$\tilde{\mathbf{Q}}_{ic}(1) = \begin{array}{cc} \downarrow_k \quad {}^{l} \rightarrow & \begin{array}{ccc|cccc} -(T-1) & \cdots & -1 & 0 & 1 & \cdots & T \end{array} \\ \begin{array}{l} -(T-1) \\ \vdots \\ 0 \\ \hline 1 \\ \hline 2 \\ \vdots \\ T \end{array} & \left(\begin{array}{c|c} 0 & 0 \\ \hline 0 & q_{ic}^{|k||l|} \\ \hline q_{ic}^{|k||l|} & 0 \end{array} \right) \end{array} \tag{6.11}$$

Then, the transition matrix $\mathbf{R}_{ic}(x)$ associated with arrival slot x can be written as:

$$\mathbf{R}_{ic}(x|v_{ic}(x\oplus 1)>0) = \begin{array}{c} \left[\begin{array}{ccccccc} \mathbf{Y}_{ic} & \mathbf{X}_{ic}(\cdot) & 0 & 0 & 0 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{Y}_{ic} & \mathbf{X}_{ic}(\cdot) & 0 & 0 & 0 & 0 & \cdots \\ 0 & \mathbf{Y}_{ic} & \mathbf{X}_{ic}(\cdot) & 0 & 0 & 0 & \cdots \\ 0 & 0 & \ddots & \ddots & \ddots & 0 & \cdots \\ 0 & \cdots & 0 & \mathbf{Y}_{ic} & \mathbf{X}_{ic}(\cdot) & 0 & \cdots \end{array} \right] \end{array} \begin{array}{c} 0 \\ \vdots \\ v_{ic}(x\oplus 1) \\ v_{ic}(x\oplus 1)+1 \\ \vdots \\ B_{ic}^{(in)} \end{array} \tag{6.12}$$

The structure of matrix $\mathbf{R}_{ic}(x)$ given in (6.12) can be explained as follows. Suppose that the number of segments y in the queue at the end of slot x is at most $v_{ic}(x \oplus 1)$. Since up to $v_{ic}(x \oplus 1)$ segments can be served within slot $x \oplus 1$, the number in the queue at the end of that slot will be 1 or 0, depending on whether an arrival occurred or not. This point is indicated by the transitions in rows 0 through $v_{ic}(x \oplus 1)$ of matrix $\mathbf{R}_{ic}(x)$. However, if at the end of slot x we have $y > v_{ic}(x \oplus 1)$, then the number in the queue at the next transition will be $y - v_{ic}(x \oplus 1)$ (plus one if an arrival occurred), as indicated by the transitions in rows $v_{ic}(x \oplus 1) + 1$ through B_{ic} of $\mathbf{R}_{ic}(x)$. Of course, y cannot exceed the queue capacity $B_{ic}^{(in)}$. Since the number of service slots $v_{ic}(x \oplus 1)$ depends on the particular slot $x \oplus 1$ within the frame, $\mathbf{R}_{ic}(x)$ is a function of x.

Matrix $\mathbf{R}_{ic}(x)$ is slightly different when $v_{ic}(x \oplus 1) = 0$; its structure is shown in 6.13. In this case, if the state of the input queue is $y = B_{ic}^{(in)}$, not only will a new arrival be discarded, but some number of currently enqueued segments may also be discarded. The matrix $\mathbf{P}_n(\cdot \mid x)$ represents the probability of segment n of an n' segment packet causing the overflow of the buffer with the arrival process in a given state z, $z > 0$, during arrival slot x. The structure of $\mathbf{P}_n(\cdot \mid x)$ may be described simply: $\mathbf{P}_n$ is a square matrix with indices on both dimensions running from $-(T-1)$ to T. The matrix may have non-zero values only for rows z, $0 < z \leq (T - n + 1)$. The reason for this boundary is that, first, for there to be an arrival, z must be greater than 0, and, second, if exactly n segments (including the currently arriving segment) are to be lost, and the total packet size is bounded by T, z (before the arrival occured) must be bounded by $T - (n-1)$. Given the arriving traffic description shown in Figure 6.4, the complete packet size (in terms of segments) may be inferred exactly as $z + n - 1$, occuring with the probability shown in (6.14).

$$\mathbf{R}_{ic}(x|v_{ic}(x\oplus 1)=0) = \begin{bmatrix} \mathbf{Y}_{ic} & \mathbf{X}_{ic}(\cdot) & 0 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{Y}_{ic} & \mathbf{X}_{ic}(\cdot) & 0 & 0 & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 0 & 0 & \mathbf{Y}_{ic} & \mathbf{X}_{ic}(\cdot) \\ 0 & \cdots & 0 & \mathbf{P}_T(\cdot)\mathbf{X}_{ic}(\cdot) & \cdots & \mathbf{P}_2(\cdot)\mathbf{X}_{ic}(\cdot) & \mathbf{Y}_{ic}+\mathbf{P}_1(\cdot)\mathbf{X}_{ic}(\cdot) \end{bmatrix} \tag{6.13}$$

$$\mathbf{P}_n(z,z) = \begin{cases} \sum_{k=0,1} \psi_{ic}(k \mid x \ominus n) q_{ic}^{(k,z+n-1)} & \forall\ z\ s.t.\ (T-n+1) \geq z > 0 \\ 0.0, & \text{otherwise} \end{cases} \tag{6.14}$$

Again, it can be easily verified that the Markov chain is irreducible, and therefore a steady-state distribution exists. Transition matrix $\mathbf{S}_{ic}$ defines a *p-cyclic* Markov chain [63], and therefore it can be solved using any of the techniques for p-cyclic Markov chains in [63, ch. 7]. We have again used the LU decomposition method in [63] to obtain the steady state probability $\pi_{ic}(x,y,z)$.

We present in Appendix B an alternative analysis which only computes occupancy probabilities for relevent slots during a frame (i.e., those arrival slots during which an input queue receives service). While the alternative analysis is less computationally intensive, it is less accurate as it assumes a fixed length packet distribution. This analysis is viewed as having merit as it should provide an upper bound for loss in subsequent calculations.

6.3.3 Output Side Analysis

Re-assembly Queue Analysis

Given the independent operation of each frequency which was described earlier, and since each output port operates on a single, fixed frequency with non-overlapping buffers, each output port may be considered in isolation. Each output port is composed of a single, fixed-frequency optical receiver, N "re-assembly queues", and one "output queue", as shown in Figure 6.1. The receiver filters out segments from the passive star coupler, allowing only those segments transmitted on a given frequency to pass to the remainder of the output port. For a given output port, j, the re-assembly queues are allocated per input port (i.e., exactly one queue per input port). As segments are received from a particular input port over a given frequency, the segments are buffered in the appropriate re-assembly

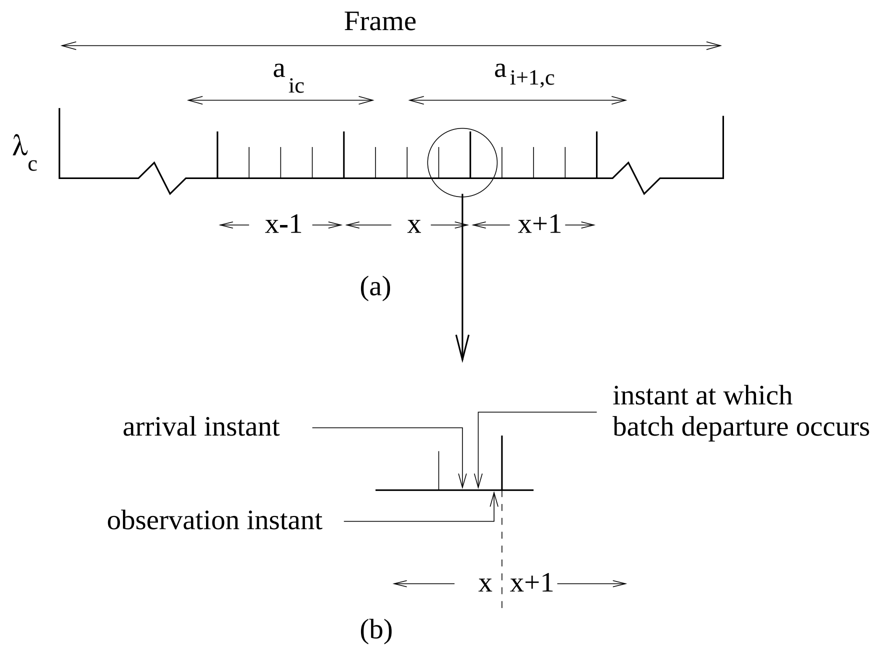

Figure 6.5: (a) Arrivals to reassembly queue i of output port j, and (b) detail showing the relationship of departure, arrival, and observation instants

Table 6.2: Transition probabilities out of state (x, ζ) of the Markov chain for re-assembly queue i of output port j

Current State	**Next State**	**Transition Probability**
(x, ζ)	$(x \oplus 1, \zeta')$	$\sum_{\zeta_{add}=\max(\zeta'-\zeta,0)}^{v_{ic}(x\oplus 1)} [\ \mathcal{P}(\zeta_{add} \mid x) \ \times\ \mathcal{F}(\zeta' \mid \zeta, \zeta_{add}, x)\ ]$

queue until a complete packet is enqueued. The re-assembly queue is large enough to accommodate at least T segments.

Let us consider the N re-assembly queues of output port j. We may define a set of N markov chains, (x, ζ), where

- x indicates the arrival slot number within the frame $(x = 0, 1, \cdots, M-1)$, and
- ζ indicates the occupancy of reassembly queue i in segments $(\zeta = 0, \cdots, T)$.

Based on the architecture description and the ordering of events shown in Figure 6.5, the transition probabilities shown in Table 6.2 follow.

The term $\mathcal{P}(\cdot)$ reflects the probability of receiving ζ_{add} segments from the appropriate input queue during slot x. We have

$$\mathcal{P}(\zeta_{add} \mid x) = \begin{cases} 1.0, & v_{ic}(x \oplus 1) = 0 \text{ and } \zeta_{add} = 0 \\ 0.0, & v_{ic}(x \oplus 1) = 0 \text{ and } \zeta_{add} \neq 0 \\ M \sum_z \pi_{ic}(x, \zeta_{add}, z), & \zeta_{add} < v_{ic}(x \oplus 1) \text{ and } v_{ic}(x \oplus 1) > 0 \\ M \sum_{y'=v_{ic}(x\oplus 1)}^{B_{ic}} \sum_z \pi_{ic}(x, y', z), & \zeta_{add} \geq v_{ic}(x \oplus 1) \text{ and } v_{ic}(x \oplus 1) > 0 \end{cases} \tag{6.15}$$

The term $\mathcal{F}(\cdot)$ is the probability that the system will be in state ζ' given that it had ζ segments in it and it received ζ_{add} segments arrive from an input queue (i.e., it discharges $\max(0, \zeta - \zeta' + \zeta_{add})$ segments), where

$$\mathcal{F}(\zeta' \mid \zeta, \zeta_{add}, x) = \begin{cases} f_{ic}(\zeta - \zeta' + \zeta_{add} \mid \zeta), & \zeta_{add} > \zeta' - \zeta \text{ and } \zeta_{add} = v_{ic}(x \oplus 1) \\ F_{ic}(\zeta' \mid \zeta), & \zeta_{add} \leq \zeta' - \zeta \text{ and } \zeta_{add} = v_{ic}(x \oplus 1) \end{cases} \tag{6.16}$$

Notice that $f_{ic}(s' \mid s)$ is defined to be the probability that the packet size is equal to s' segments given that it is greater than s segments.

$$f_{ic}(s' \mid s) = \begin{cases} \frac{f_{ic}(s')}{F_{ic}(s)}, & s' > s \\ 0.0, & \text{otherwise} \end{cases} \tag{6.17}$$

Also $F_{ic}(s \mid s')$ is the conditional cumulative probability distribution for packet size in terms of segments given that the packet size is greater than s', i.e.,

$$F_{ic}(s) = \sum_{\dot{s}=s+1}^{T} f_{ic}(\dot{s}) \tag{6.18}$$

$$F_{ic}(s \mid s') = \begin{cases} \frac{F_{ic}(s)}{F_{ic}(s')}, & s > s' \\ 0.0, & otherwise \end{cases} \tag{6.19}$$

In addition, the following was taken into account when constructing Table 6.2.

1. The conditional probability of having y customers in an input queue during only slot x (i.e., given slot x) and without regard for the state of the input process is $M \sum_z \pi_{ic}(x, y, z)$.
2. The number of departures, if any, must exceed the occupancy of the system before the arrivals, ζ.

3. If the arriving state ζ is equal to the departing state ζ', there were either (a) no arrivals *and* no departures from the system or (b) the exact same number of arrivals and departures from the system.

4. As this analysis assumes that only one packet may be completed during a slot, we impose the restriction that the minimum packet size must be greater than or equal to m segments where m is the ratio of switch ports to channels. This restriction is imposed only to simplify the analysis and its revocation would only add a simple, but tedious, element of combinatorics to the anlaysis.

The transition probability matrix for each reassembly queue is then constructed and the occupancy probability for re-assembly queue i of output port j, $\Phi_{ij}(x, \zeta)$, may be determined. We have used the LU decomposition method as prescribed in [63] to obtain $\Phi_{i,j}(x, \zeta)$, the steady-state probability that reassembly queue (i, j) has ζ segments at the end of slot x. However, since the actual number of segments in each packet is lost from our model once the segments have been buffered at the input queue, the re-assembly of the segments at this point will most likely not result in the exact, originally presented packets and will impact the accuracy of $\Phi_{ij}(x, \zeta)$. The following sequence of events exemplifies this point.

1. Assume a packet length distribution which allows for packets between four and seven segments long.

2. Assume that a packet which was originally six segments long is forwarded, one segment at a time, to a reassembly queue.

3. After the fifth segment, based on re-partitioning according to the original packet distribution, a new "packet" is assumed to be generated and is forwarded to the output queue.

4. One segment is forwarded to the re-assembly queue after the other five have departed for the output queue and remains in the re-assembly queue until it is "absorbed" as part of another packet.

In a lightly loaded system, this type of situation could significantly skew the buffer occupancy distribution, $\Phi_{ij}(x, \zeta)$.

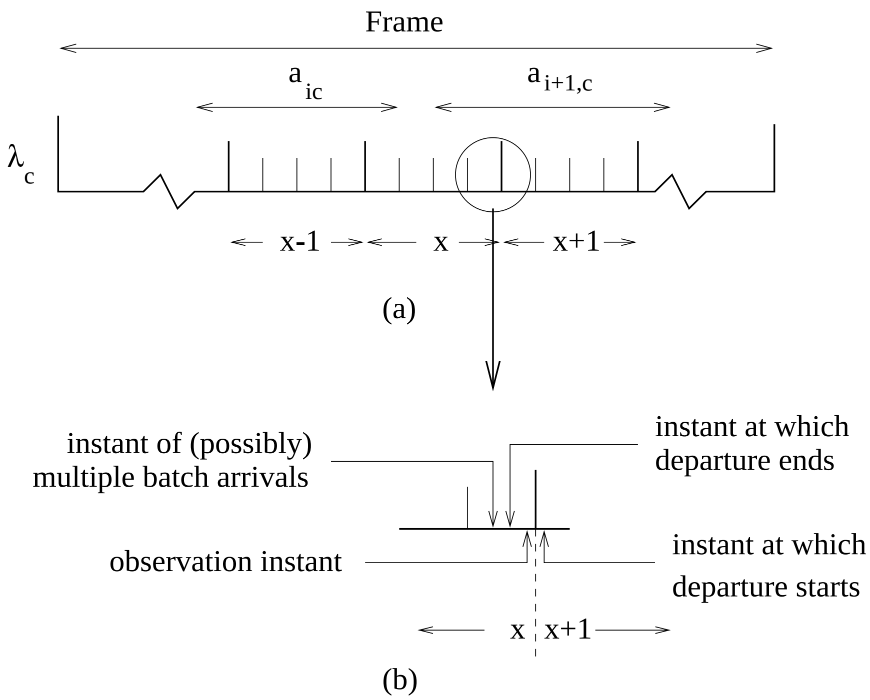

Figure 6.6: (a) Arrivals to output port j from re-assembly queues i and $i+1$, and (b) detail showing the relationship of departure, arrival, and observation instants

Output Queue Analysis

As in the previous section, we obtain the queue-length distribution of output port j at arrival slot boundaries.

Consider the point that several output ports may share a single channel. Define r_{cj} as probability that a particular packet transmitted on channel c (and subsequently reassembled) is intended for output port j.

Let (x, w) be the state associated with output port j, where

- x indicates the arrival slot number within the frame ($x = 0, 1, \cdots, M-1$), and
- w indicates the number of segments at the output queue ($w = 0, 1, \cdots, B_j^{(out)}$).

Observe now that (a) at each state transition x advances by one (modulo-M), (b) exactly one segment departs from the queue as long as the queue is not empty, (c) a number $0 \le s \le T$ of segments may be transmitted from each of the relevent reassembly queues to output port j within external slot $x \oplus 1$, and that (d) the queue capacity is $B_j^{(out)}$. Based on the first item above, it can be easily seen that, the transition matrix $\mathbf{T}_j$ of the Markov chain defined by the evolution of the state (x, w) of output queue j has the following form, which is similar to that of matrix $\mathbf{S}_{ic}$ given by (6.7):

$$\mathbf{T}_j = \begin{bmatrix} 0 & \mathbf{U}_j(0) & 0 & 0 & \cdots & 0 \\ 0 & 0 & \mathbf{U}_j(1) & 0 & \cdots & 0 \\ 0 & 0 & 0 & \mathbf{U}_j(2) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \mathbf{U}_j(M-2) \\ \mathbf{U}_j(M-1) & 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \begin{matrix} 0 \\ 1 \\ 2 \\ \vdots \\ M-2 \\ M-1 \end{matrix} \tag{6.20}$$

Formulating the matrices $\mathbf{U}_j(x)$, however, is somewhat more complicated. The main difficulty is encountered in cases in which multiple input ports are serviced by the same channel during a given arrival slot, x. In this situation, determining which of a number of packets may be accepted by an output port's queue which may be close to completely occupied may be easily done using a heuristic. Expressing the transition probabilities between two states in a closed-form expression, however, is significantly more difficult. Therefore, we present the algorithmic form for building the matrices $\mathbf{U}_j(x)$. Notice three points:

1. The ordering of events relevent to the output queues are given in Figure 6.6[7].
2. For notational clarity, the elements of matrix $\mathbf{U}_j(x)$ are denoted as $\mathbf{U}_j(x, k, l)$ where k is the row index and l is the column index.
3. The channel c referenced in line 3 of the following algorithm is the (fixed) channel which output port j observes.

[7]While, in reality, the packet would be transferred from the re-assembly queue to the output queue immediately upon its consolidation, the representation that the packet is transferred at the end of the arrival slot does not sacrifice any accuracy.

$\mathbf{U}_j(x) \leftarrow [\mathbf{0}]$

loop: $\forall\ w \in \{0, 1, \cdots, B_j\}$

  $\mathcal{I}_j(x) \leftarrow$ an ordered set of the input ports serviced by channel c during slot x with arity $\mid \mathcal{I}_j(x) \mid$

  loop: $\forall\ s \in \{\{\overbrace{0, 0, \cdots, 0}^{|\mathcal{I}_j(x)|}\}, \{0, 0, \cdots, 1\}, \cdots, \{T, T, \cdots, T\}\}$

    $w' \leftarrow w$

    loop: $\forall\ i \in \{1, \cdots, \mid \mathcal{I}_j(x) \mid\}$

      if $\{w' + s_i \leq B_j\} \Rightarrow \{w' \leftarrow w' + s_i\}$

    if $\{w > 0\} \Rightarrow \{w' \leftarrow w' - 1\}$

    $\mathbf{U}_j(x, w, w') \leftarrow \mathbf{U}_j(x, w, w') + \prod_{i=1}^{|\mathcal{I}_j(x)|} L_i(s_i \mid x)$

In the preceding algorithm, $L_i(s_i \mid x)$ is the probability that a packet consisting of s_i segments is transmitted during slot x from reassembly queue i to output port j[8]. Recall that the actual packets are not propagated from the reassembly queue to the output queue; instead, the probability of a packet being completed during a slot is used with the packet length distribution to generate $L_i(s_i \mid x)$. $L_i(s_i \mid x)$ is defined in (6.21).

$$L_i(s_i \mid x) = Mr_{cj} \sum_{\zeta=0}^{T-1} \sum_{\zeta_{add}=\max(1, s_i-\zeta)}^{v_{ic}(x \oplus 1)} \mathcal{F}(\zeta + \zeta_{add} - s_i \mid \zeta, \zeta_{add}, x) \mathcal{P}(\zeta_{add} \mid x) \Phi_{ij}(x \ominus 1, \zeta) \tag{6.21}$$

Based on the presented algorithm, we can solve for the steady state occupancy probability of output port j during a slot x, $\pi_j(x, w)$; we have again used an LU decomposition to obtain these values.

[8]Since in most cases only one or two input ports will transmit to the same channel within an arrival slot (refer also to Figure 6.3), and since a packet can only be completed in a reassembly queue while the input port is transmitting over the given channel, the dimension of the vector s will generally be only one or two. Thus, this loop can be executed very fast, in spite of the exponential time implied by the general form presented.

6.3.4 Summary of the Decomposition Algorithm

Below we summarize our approach . We assume that quantities $\{a_{ic}\}$ and the corresponding schedule (see [60]) are given.

1. Given each arrival process, defined by $\mathbf{A}_{ic}$ and $\mathbf{Q}_{ic}$, formulate $\tilde{\mathbf{A}}_{ic}$, $\tilde{\mathbf{Q}}_{ic}(0)$, and $\tilde{\mathbf{Q}}_{ic}(1)$ per equations (6.9), (6.10), and (6.11), respectively. Additionally, compose the matrices $\mathbf{P}_n$ corresponding to each arrival process per equation (6.14).

2. For each arrival slot x, use the schedule and expressions (6.6) to compute the quantities $v_{ic}(x),\ i = 1, \cdots, N,\ j : \lambda(j) = \lambda_c$.

3. For each input queue i, construct the transition probability matrix $\mathbf{S}_{ic}$ from (6.1), (6.2), (6.7), (6.8), and (6.12). Solve this matrix for $\pi_{ic}(x, y, z)$.

4. For each reassembly queue (i, j), use $\pi_{ic}(x, y, z)$, (6.15), and (6.16) to to build a transition probability matrix for that queue. Solve the matrix to obtain $\Phi_{ij}(x, \zeta)$, the steady-state probability that reassembly queue (i, j) has ζ cells in its queue at the end of slot x.

5. For each output port $j \in \mathcal{R}_c$, use $\pi_{ic}(x, y, z)$, $\Phi_{ij}(x, \zeta)$, and (6.21) to construct the transition matrix $\mathbf{T}_j$ given by (6.20). Solve the matrix to obtain $\pi_j(x, w)$, the steady-state probability that port j has w cells in its queue at the end of slot x.

6.4 Loss Probabilities

We now use the queue-length distributions derived in the previous section ($\pi_{ic}(x, y, z)$, $\Phi_{ij}(x, \zeta)$, and $\pi_j(x, w)$) to obtain the packet loss probability at the input and output ports.

6.4.1 Segment and Packet Loss Probability at an Input Port

Let Ω_{ic} be the mean packet loss probability at the c-th queue of input port i, i.e., the probability that a packet arriving to that queue will be lost. Ω_{ic} can be expressed as:

$$\Omega_{ic} = \frac{E[\text{number of packets lost per frame at queue } c \text{ of port } i]}{E[\text{number of arrivals per frame at queue } c \text{ of port } i]} \tag{6.22}$$

Notice that, for the system to complete a packet, the source must go through either state 1 (if the arriving packet is buffered) or state -1 (if the arriving packet is discarded). Therefore, the denominator of this fraction may be easily reduced.

$$E[\text{number of arrivals per frame at queue } c \text{ of port } i] = \sum_{x=0}^{M-1}\left(\sum_y \pi_{ic}(x,y,z=1) + \sum_y \pi_{ic}(x,y,z=-1)\right) \tag{6.23}$$

To calculate the numerator we observe that all packets begin their transmission periods in one of the states numbered above zero. In the case where the packet length follows an arbitrary distribution, therefore, it is possible for a one segment packet to traverse state 1 and still be dropped. In view of this point, we have

$$E[\text{number of packets lost per frame at queue } c \text{ of port } i] = \sum_{x=0}^{M-1}\sum_y \pi_{ic}(x,y,z=-1) + \sum_{x:v_{ic}(x)=0} \pi_{ic}(x,y=B_{ic},z=1)f_{ic}(1) \tag{6.24}$$

Using this same strategy, the mean segment loss probability, ω_{ic} may also be easily found.

$$\begin{aligned} \omega_{ic} &= \frac{E[\text{number of segments lost per frame at queue } c \text{ of port } i]}{E[\text{number of arriving segments per frame at queue } c \text{ of port } i]} \\ &= \frac{\sum_{\forall x,y;\forall z<0}\pi_{ic}(x,y,z)+\sum_{\forall x;\forall z>0}\pi_{ic}(x,y=B_{ic},z)}{\sum_{\forall x,y;\forall z\neq 0}\pi_{ic}(x,y,z)} \end{aligned} \tag{6.25}$$

6.4.2 Segment and Packet Loss Probability at an Output Port

The packet and segment loss probabilities at an output port is more complicated to calculate, since we may have multiple packet arrivals to the given output port within a single arrival slot (refer to Figure 6.6(a)). Additionally, the order of the arrivals must be accounted for in determining which packets are potentially lost. Therefore, the packet and segment loss probabilities, $\Omega_j(x)$ and $\omega_j(x)$, are not easily expressed in a closed form expressions but they can be easily calculated using a slightly modified version of the algorithm presented in Section 6.3.4.

6.5 Numerical Results

We now demonstrate the accuracy of our analysis by applying the decomposition algorithm to a 8×8 switch and comparing it to simulation results. We have selected the following set of parameters for our study case. Four different packet length distributions (and, hence, arrival processes) are used in these experiments. In varying these four distributions,

Table 6.3: Packet length distributions for considered arrival processes

Percentage of all packets	Process 1	Process 2	Process 3	Process 4
5 segments	100%	25%	16.7%	0%
6 segments	0%	25%	16.7%	0%
7 segments	0%	25%	16.7%	0%
8 segments	0%	25%	50%	100%

Table 6.4: Channel sharing for $C = 2, 3$

	$C = 2$	$C = 3$
$\mathcal{R}_1$	$\{1, 3, 5, 7\}$	$\{1, 4, 7\}$
$\mathcal{R}_2$	$\{2, 4, 6, 8\}$	$\{2, 5, 8\}$
$\mathcal{R}_3$		$\{3, 6\}$

the mass of the packet length distribution was shifted from favoring shorter packet lengths to favoring longer packet lengths. The mean utilization of the input channel to each input queue remained fixed at 10% for all experiments. The four packet length distributions are shown in Table 6.3.

We examine cases in which the number of channels in the switch is either two or three. The eight output ports are assigned to the channels using a round-robin assignment algorithm. Based on these assignments, the traffic dispersed over a channel is equally divided between the assigned output ports, thereby generating values for r_{cj} in our analysis. The output port assignments for each channel is shown in Table 6.4; the notation for sets of output ports sharing channel c, $\mathcal{R}_c$, is as was introduced in Chapter 4.

For all the results we present in this section, we have let all input and output queues have the same buffer capacity B (i.e., $B_{ic}^{(in)} = B_j^{(out)} = B$) so that we can vary a single parameter; the buffer length was varied from 10 to 20 segments for all experiments. The algorithm for generating a_{ic} and, as a result, the schedules, and the simulation parameters are as reported in Section 4.6.

Figures 6.7, 6.8, 6.9, and 6.10 show the segment and packet loss probabilities for the input queues as a function of buffer size B in the two and three wavelength switches. The simulation results are presented as confidence intervals, though they appear to be only

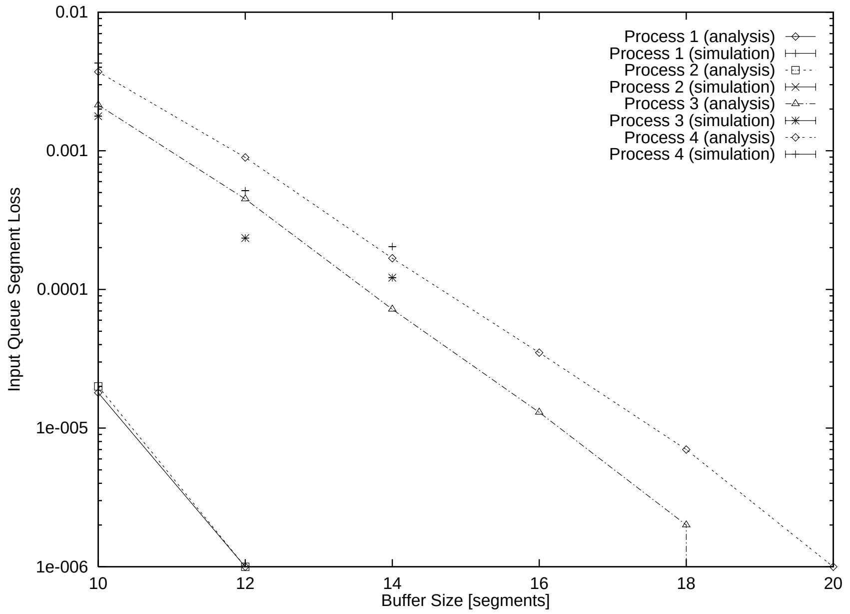

Figure 6.7: Input Queue Segment Loss Probability $\omega_{1,1}$ for $C = 2$

single points. We present only the input queue in input port 1 which corresponds to the the first wavelength, λ_1; given our system description, this queue is representative of the other input queues.

Considering each of the graphs independently, while noting that the mean arrival rate (i.e., the mean number of arriving segments per arrival slot) of the input queue remains constant for each of the arrival processes, we observe that the loss probabilities all increase as the average packet size increases. (Notice that the mean service rate of each of the input queues changes from 2 segments per 4 arrival slots when $C = 2$ to 2 segments per 6 arrival slots when $C = 3$.) This point may be intuitively justified since, knowing that the longer packets means longer streams of back-to-back segment arrivals, the probability of loss may be inferred from the probability of lower (packet) interarrival times.

Consider the following example. First, observe that once an input queue begins to receive a packet, the queue's occupancy will necessarily grow (since its arrival rate will instantaneously exceed its service rates while receiving the packet) until the entire packet has been enqueued. For example, given an 8 segment packet and $C = 2$, the packet will be received over exactly 8 arrival slots. During those 8 arrival slots, only 4 segments will be serviced, meaning that the arrival of this packet may increase the queue's length

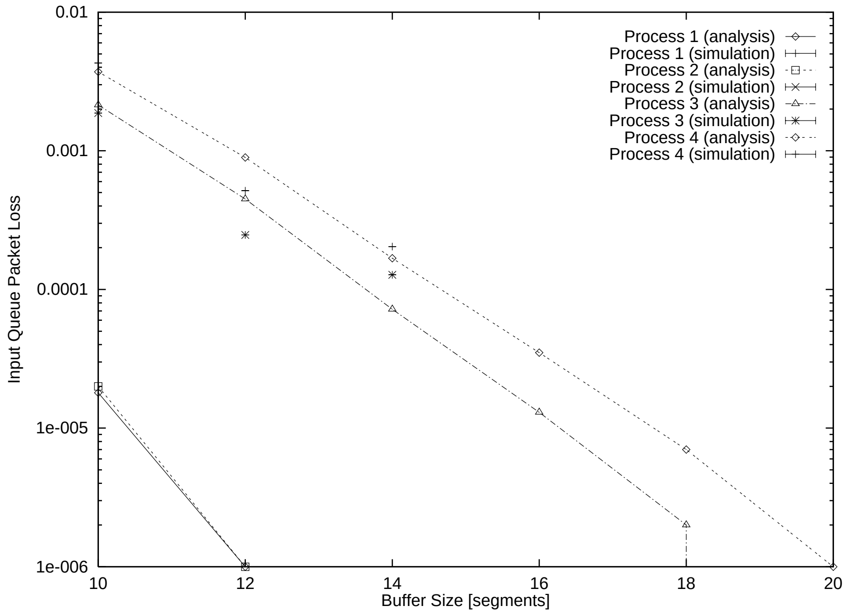

Figure 6.8: Input Queue Packet Loss Probability $\Omega_{1,1}$ for $C = 2$

instantaneously by as much as 4 segments (if another packet were immediately received). Further, the packet will not complete its transmission through the switch for a total of *16* arrival slots due to the structure of the schedule involved. Therefore, if the probability of another packet's arrival within, in this example, the next 8 arrival slots were high, the probability of loss of subsequent packets would be significant. Since the mean service rate of the input queue is lower in the case of $C = 3$, the loss rate would be even higher (than when $C = 2$); this point is shown by comparing Figures 6.7 and 6.8 to Figures 6.9 and 6.10, respectively.

It may have been noticed in the previous paragraphs that, at several points, observations were made regarding "loss probability", not "segment loss probability" or "packet loss probability". In fact, it is shown in the corresponding figures that the values of $\omega_{1,1}$ and $\Omega_{1,1}$ are virtually identical. Since packet arrivals (and their lengths) are independent of the occupancy level of the input queue, and since the packet length distribution is fixed for the entirety of each of the experiments, it is expected that the percentage of packets and segments which are lost would be comparable. This explains the similarities between the pairs of corresponding figures depicting $\omega_{1,1}$ and $\Omega_{1,1}$.

Our output port loss calculations seem to be most accurate as packet sizes are

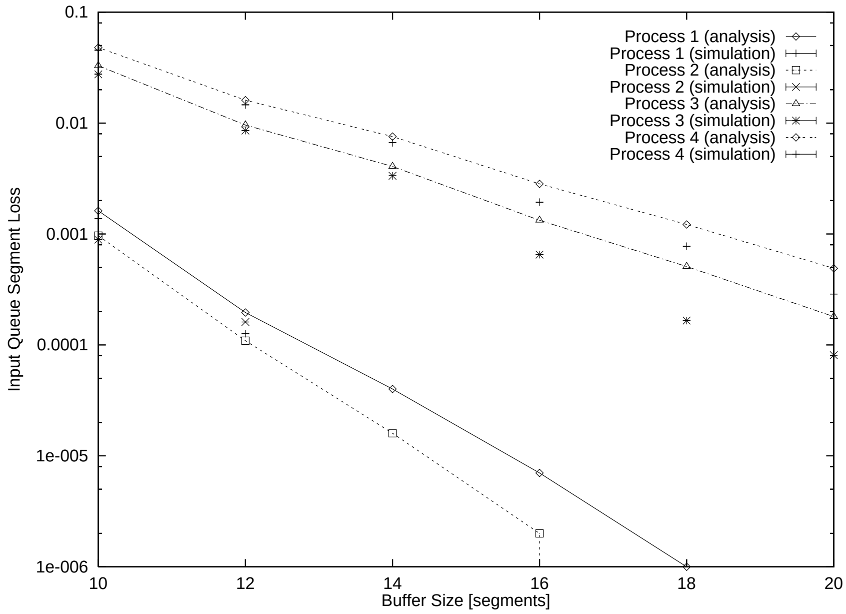

Figure 6.9: Input Queue Segment Loss Probability $\omega_{1,1}$ for $C = 3$

shorter and buffer lengths are shorter. Most notably, the analysis indicates a significant drop in loss when the buffer size is increased from 14 to 16 segments. Consider, however, the formulation of the expression for $L_i(\cdot)$ in (6.21). It should be noticed that this equation relies on steady-state calculations for occupancy levels of the input, re-assembly, and output queues, and does not account for the correlated arrivals. Therefore, the calculation for $L_i(s_i \mid x)$ does not account for the probability of there having been a packet delivered, for instance, during arrival slot $x \ominus 1$. In fact, it would be difficult to determine over what duration of arrival slots a significant correlation between successively arriving packets would exist. Appendix C does present an alternative expression for $L_i(s_i \mid x)$ which more accurately estimates the output loss.

Given this point, the reasoning for the sharp decrease in loss as B increases to 16 is that the calculations now conclude that two 8-segment packets can be stored in the buffer, whereas, when B was less than 16, obviously, they could not. Similarly, we provide more accurate estimates for larger, less varying packet sizes and smaller buffers (when there exists fewer combinations of packet sizes which may comprise a full buffer).

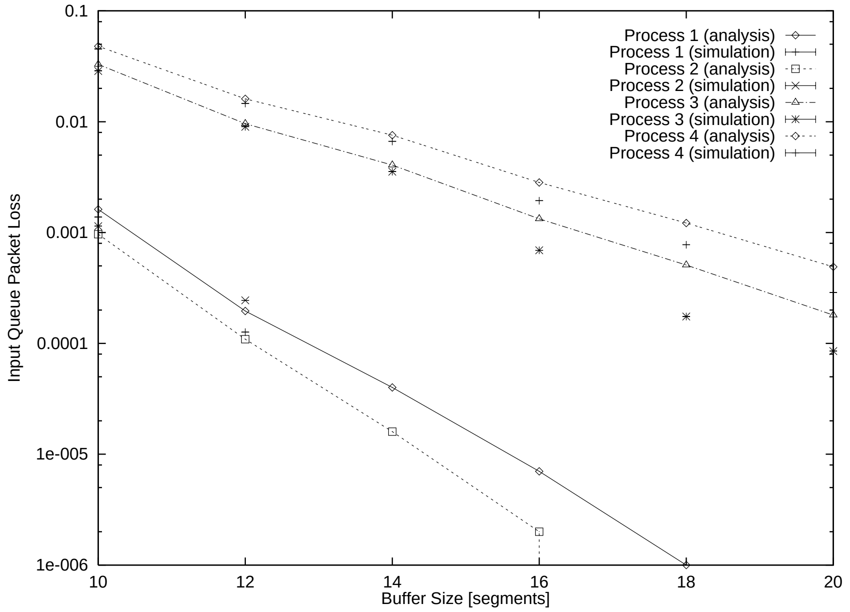

Figure 6.10: Input Queue Packet Loss Probability $\Omega_{1,1}$ for $C = 3$

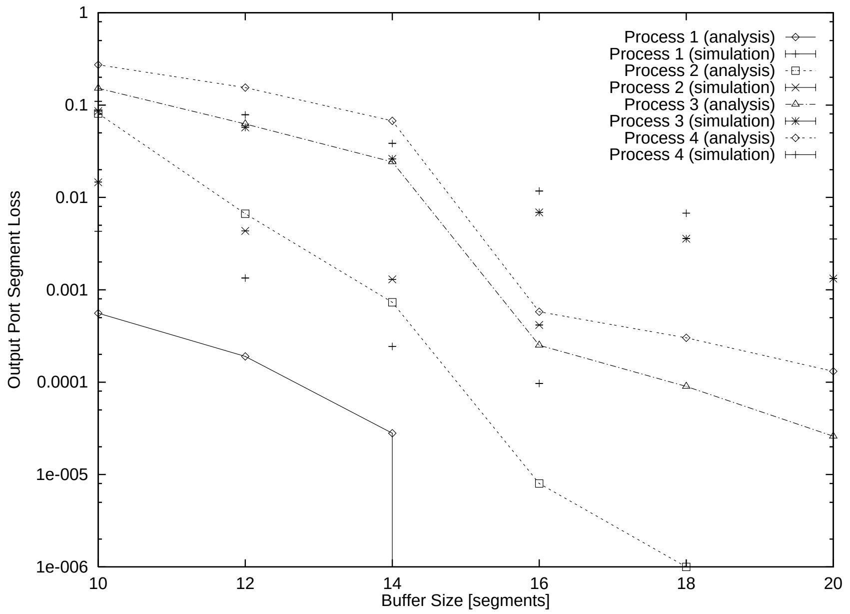

Figure 6.11: Output Port Segment Loss Probability ω_1 for $C = 2$

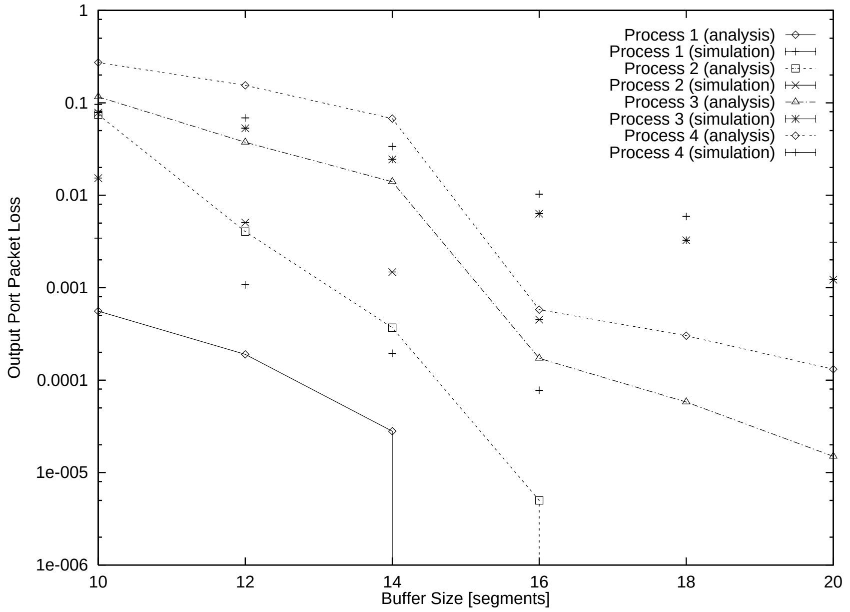

Figure 6.12: Output Port Packet Loss Probability Ω_1 for $C = 2$

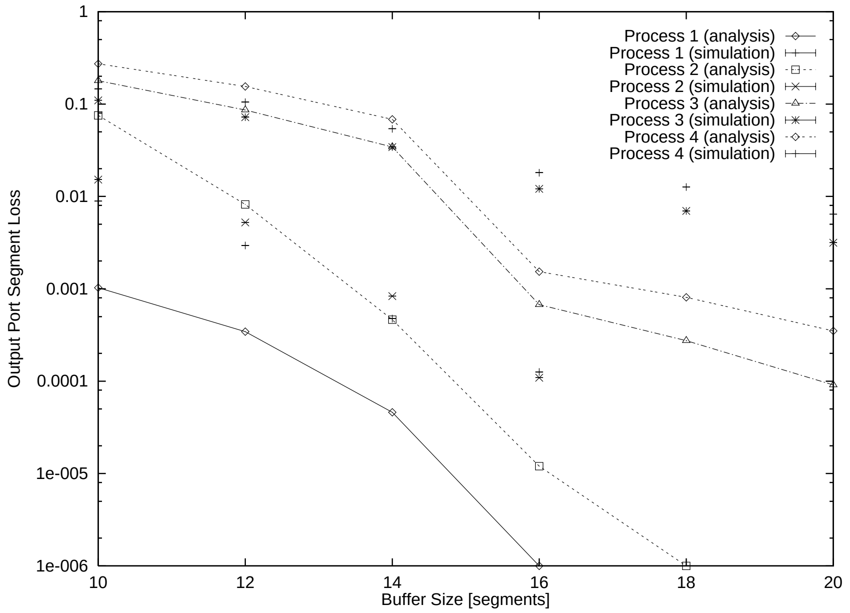

Figure 6.13: Output Port Segment Loss Probability ω_1 for $C = 3$

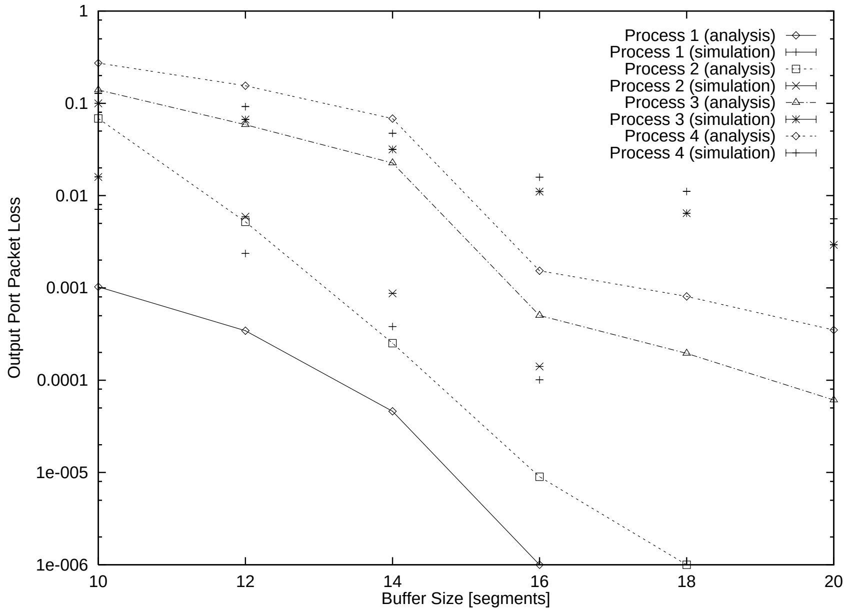

Figure 6.14: Output Port Packet Loss Probability Ω_1 for $C = 3$

Chapter 7

Conclusions and Future Research

In this work, we have developed a decomposition algorithm to obtain the queue-length distributions at the input and output ports of a number of photonic ATM switch architectures based on a single-hop WDM architecture. Specifically, in Chapter 4, a quantitative analysis for the performance of a TTFR single-hop photonic switch architecture was derived. This analysis was presented assuming uncorrelated traffic which was transmitted using a protocol which supports fixed length transmission units (e.g., ATM cells). This analysis included two primary studies. First, a derivation of the delay distribution for cells traversing this switch was found and presented in Section 4.5. Second, given that this switch architecture employs input and output queueing (to eliminate "Head of Line" blocking), cell loss was calculated for both input and output queues.

As part of this study, the relationship between the number of wavelengths in such a switch, the schedule for sharing the wavelengths between input ports, and the loss probabilities was explored. It was found that there exists a complex interaction between switch parameters such as the length and allocation of the schedule to an input queue, the allocation of output ports to wavelengths, and buffer length, all of which can have a significant impact on the loss encountered by such a switch. The presented loss analysis was subsequently validated by simulation results and shown to be accurate to with 2% within our study. A discussion of the most significant approximation made in this study is given in Appendix A.

In Chapter 5, we presented a parallel study for a FTTR architecture, the dual to the TTFR presented in Chapter 4. Focusing again on loss characteristics and motivated by the potential for supporting multicast traffic, we found that the "relocation" of the tunability feature within the switch did not effect the performance analysis of the switch. In this study, it was shown that, by transforming the system such that members of the same receiving multicast group were treated as a single entity (a "virtual receiver"), the analysis of the FTTR architecture reduced to be a simple extension of the TTFR case.

In Chapter 6, we presented a further extension of the basic TTFR switch by allowing for variable length transmission units (i.e., packets). By slightly augmenting the switch's architecture to allow for segmentation and reassembly to occur within the switch, we significantly extended the previous analyses to accommodate the new traffic patterns. Loss calculations were also presented and compared to simulation results, both in terms of segment loss and packet loss. In this work, we saw the significant impact that traffic patterns, in addition to the previously mentioned parameters, would have on this type of photonic switch. The loss calculations (and the errors shown in Figures 6.7 through 6.10) in Chapter 6 highlight the issues associated with using steady-state values under highly correlated conditions; therefore, a heuristic for more accurately estimating the output port's losses is presented in Appendix C.

Directions for Future Research

Aside from the physical layer issues which were highlighted in Chapter 3, a number of problems still remain at the architectural level. One obvious issue remaining is the improvement of the performance analysis of the systems operating under *highly correlated* traffic patterns.

Additionally, while dedicated buffer architectures are convenient for the purposes of analysis, the implementation of such systems have been shown in the electronic domain as not being efficient. The introduction of the photonic domain will not have any bearing on this conclusion: therefore, the analysis of architectures which utilize shared buffers between input queues within the same input port is another immediate direction for this work.

As was mentioned in Chapter 2, the models in this work assumed that schedules would be fixed and changes "... will more likely take place over larger scales in time." This comment begs the question that, given that schedules for allocating bandwidth within a photonic switch is an implemented strategy, how can schedules be evolved as a network of such switches operate? Schedules should not necessarily need to be rebuilt based upon every connection's being accepted or terminated; therefore, the building of schedules which are not only optimally sized, but also tolerant of some amount of change to the set of supported connections, should be examined.

Congestion and admission control (CAC) algorithms will also need to be reconsidered. While photonic switching technology will allow for an increase in the amount of switched bandwidth by an order of magnitude, current-day CAC algorithms (e.g., [58], [21]) cannot directly address the allocation of connections to separate wavelengths. In other words, while a switch may be able to support the bandwidth required to accept a new connection, the currently supported connections may not be able to be reallocated to different wavelengths in order to provide enough free bandwidth on a single channel. (Reallocation of connections to different wavelengths must be possible in such a switch since, if reallocation is is not possible, the fragmentation of allocated bandwidth over time could result in significant amounts of wasted bandwidth within a schedule.) Additionally, given that connections may be reallocated within a switch, other issues, such as (1) the connection's ability to withstand the temporary delay which might be encountered during reconfiguration and (2) the propagated effects of reconfiguration of a single switch throughout a network, become significant.

Bibliography

[1] A. S. Acampora and M. J. Karol. An overview of lightwave packet networks. *IEEE Network*, pages 29–41, January 1989.

[2] H. Ahmadi and W. E. Denzel. A survey of modern high-performance switching techniques. *IEEE JSAC*, pages 1227–1237, September 1989.

[3] E. Arthurs, J. M. Cooper, M. S. Goodman, H. Kobrinski, M. Tur, and M. P. Vecchi. Multiwavelength optical crossconnect for parallel-processing computers. *Electronic Letters*, 24:119–120, 1988.

[4] E. Arthurs, M. S. Goodman, H. Kobrinski, and M. P. Vecchi. HYPASS: An optoelectronic hybrid packet switching system. *IEEE Journal on Selected Areas in Communications*, 6:1500–1510, December 1988.

[5] Edward Arthurs, Matthew S. Goodman, Chong K. Kim, and Tony T. Lee. U.S. Patent 04896934, January 1990.

[6] Edward Arthurs, Matthew S. Goodman, Chong K. Kim, and Tony T. Lee. U.S. Patent 05005167, April 1991.

[7] Edward Arthurs, Matthew S. Goodman, Haim Kobrinski, and Mario P. Vecchi. HYPASS: An optoelectronic hybrid packet switching system. *IEEE Journal on Selected Areas in Communications*, 6(9):1500–1510, December 1988.

[8] M. Azizoglu, R. A. Barry, and A. Mokhtar. Impact of tuning delay on the performance of banwidth-limited optical broadcast networks with uniform traffic. *IEEE Journal on Selected Areas in Communications*, 14(5):935–944, June 1996.

[9] I. Baldine and G. N. Rouskas. Reconfiguration in rapidly tunable transmitter, slowly tunable receiver single-hop WDM networks. Technical Report TR-96-10, North Carolina State University, Raleigh, NC, 1996.

[10] K. Bogineni, K. M. Sivalingam, and P. W. Dowd. Low-complexity multiple access protocols for wavelength-division multiplexed photonic networks. *IEEE Journal on Selected Areas in Communications*, 11(4):590–604, May 1993.

[11] C. Brackett. Dense wavelength division multiplexing networks: Principles and applications. *IEEE Journal on Selected Areas in Communications*, 8(6):948–964, August 1990.

[12] R. M. Bulley, M. S. Goodman, H. Kobrinski, C. N. Lo, K. W. Loh, M. P. Vecchi, and C. A. Brackett. Experimental demonstration of LAMBDANET: A multiwavelength optical network. In *Proceedings of ECOC '87: 13th European Conference on Optical Communication*, pages 345–348, 1987.

[13] Mon-Song Chen, N. R. Dono, and R. Ramaswami. A media-access protocol for packet-switched wavelength division multiaccess metropolitan area networks. *IEEE Journal on Selected Areas in Communications*, 8(6):1048–1057, August 1990.

[14] R. Chipalkatti, Z. Zhang, and A. S. Acampora. Protocols for optical star-coupler network using WDM: Performance and complexity study. *IEEE Journal on Selected Areas in Communications*, 11(4):579–589, May 1993.

[15] I. Chlamtac and A. Ganz. Channel allocation protocols in frequency-time controlled high speed networks. *IEEE Transactions on Communications*, 36(4):430–440, April 1988.

[16] Arturo Cisneros and C. A. Brackett. A large ATM switch based on memory switches and optical star couplers. In *Proceedings of ICC '91*, pages 721–728, 1991.

[17] R. L. Cruz and J-T. Tsai. COD: Alternative architectures for high speed packet switching. *IEEE/ACM Transactions on Networking*, 4(1):11–21, February 1996.

[18] N. R. Dono, P.E. Green Jr., K. Liu, R. Ramaswami, and F. F. Tong. A wavelength division multiple access network for computer communication. *IEEE Journal on Selected Areas in Communications*, 8(6):983–994, August 1990.

[19] P. W. Dowd. Random access protocols for high speed interprocessor communication based on an optical passive star topology. *Journal of Lightwave Technology*, LT-9:799–808, June 1991.

[20] P. W. Dowd, J. A. Perreault, J. C. Chu, D. C. Hoffmeister, and D. Crouse. Lightning: A scalable dynamically reconfigurable hierarchical WDM network for high-performance clustering. In *Proceedings of the IEEE Fourth International Symposium on High Performance Distributed Computing*, August 1995.

[21] Anwar Elwalid and TITLE = Debasis Mitra".

[22] Kai Y. Eng. A photonic Knockout switch for high-speed packet networks. *IEEE Journal on Selected Areas in Communications*, SAC-6(7):1107–1116, August 1988.

[23] Kai Y. Eng and Mark J. Karol. Gigabit-per-second ATM packet switching with the growable switch architecture. In *Proceedings of Globecom '91*, pages 1075–1081, 1991.

[24] E. Hall *et al.* The Rainbow-II gigabit optical network. *IEEE Journal Selected Areas in Communications*, 14(5):814–823, June 1996.

[25] E. M. Foo and T. G. Robertazzi. A distributed global queue transmission strategy for a WDM optical fiber network. In *Proceedings of INFOCOM '95*, pages 154–161. IEEE, April 1995.

[26] A. Ganz. End-to-end protocols for WDM star networks. In *IFIP/WG6.1-WG6.4 Workshop on Protocols for High-Speed Networks*, pages 219–235, May 1989.

[27] A. Ganz and Z. Koren. WDM passive star - protocols and performance analysis. In *Proceedings of INFOCOM '91*, pages 991–1000. IEEE, April 1991.

[28] M. S. Goodman and H. Kobrinski. Dynamic wavelength tuning for broadband optical packet switching. In K. Tada and H. S. Hinton, editors, *Photonic Switching II: Proceedings of the International Topical Meeting*. Springer–Verlag, 1990.

[29] M. S. Goodman, H. Kobrinski, M. P. Vecchi, R. M. Bulley, and J. L. Gimlett. The LAMBDANET multiwavelength network: Architecture, applications, and demonstrations. *IEEE Journal on Selected Areas in Communications*, 8(6):995–1004, August 1990.

[30] Matthew S. Goodman. Multiwavelength networks and new approaches to packet switching. *IEEE Communications Magazine*, pages 27–35, October 1989.

[31] Matthew S. Goodman, Haim Kobrinski, Mario P. Vecchi, Ray M. Bulley, and James L. Gimlett. The LAMBDANET multiwavelength network: Architecture, applications, and demonstrations. *IEEE Journal on Selected Areas in Communications*, JSAC-8(6):995–1004, August 1990.

[32] P. E. Green. *Fiber Optic Networks*. Prentice-Hall, Englewood Cliffs, New Jersey, 1993.

[33] P. E. Green. Optical networking update. *IEEE Journal Selected Areas in Communications*, 14(5):764–779, June 1996.

[34] I. M. I. Habbab, M. Kavehrad, and C.-E. W. Sundberg. Protocols for very high-speed optical fiber local area networks using a passive star topology. *Journal of Lightwave Technology*, LT-5(12):1782–1793, December 1987.

[35] P. A. Humblet, R. Ramaswami, and K. N. Sivarajan. An efficient communication protocol for high-speed packet-switched multichannel networks. *IEEE Journal on Selected Areas in Communications*, 11(4):568–578, May 1993.

[36] Andrezej Jajszczyk and H. T. Mouftah. Photonic fast packet switching. *IEEE Communications Magazine*, pages 58–65, February 1993.

[37] F. J. Janniello, R. Ramaswami, and D. G. Steinberg. A prototype circuit-switched multi-wavelength optical metropolitan-area network. In *Proceedings of ICC '92*, pages 818–823, 1992.

[38] A. Jaszczyk and H. T. Mouftah. Photonic fast packet switching. *IEEE Communications Magazine*, pages 58–65, February 1993.

[39] J. Jue, M. Borella, and B. Mukherjee. Performance analysis of the Rainbow WDM optical network prototype. *IEEE Journal Selected Areas in Communications*, 14(5):945–951, June 1996.

[40] Mark J. Karol and Michael G. Hluchyj. The Knockout packet switch: Principles and performance. In *Proceedings of the 12th Conference on Local Computer Networks*, pages 16–22, October 1987.

[41] Mark J. Karol, Michael G. Hluckyj, and Samuel P. Morgan. Input versus output queueing on a space-division packet switch. *IEEE Transactions on Communications*, COM-35(12):1347–1356, December 1987.

[42] Mark J. Karol and Chih-Lin I. Performance analysis of a growable architecture for broadband packet (ATM) switching. *IEEE Transactions on Communications*, 40(2):431–439, February 1992.

[43] H. Kobrinski, R. M. Bulley, M. S. Goodman, M. P. Vecchi, and C. A. Brackett. Demonstration of high capacity in the lambdanet architecture: A multiwavelength optical network. *Electronic Letters*, 23(16):824–826, July 1987.

[44] T. T. Lee, M. S. Goodman, and E. Arthurs. A broadband optical multicast switch. In *Proceedings of ISS '90*, 1990.

[45] D. A. Levine and I. F. Akyildiz. PROTON: A media access control protocol for optical networks with star topology. *IEEE/ACM Transactions on Networking*, 3(2):158–168, April 1995.

[46] N. Mehravari. Performance and protocol improvements for very high-speed optical fiber local area networks using a passive star topology. *Journal of Lightwave Technology*, 8(4):520–530, April 1990.

[47] A. Muir and J. J. Garcia-Luna-Aceves. Distributed queue packet scheduling algorithms for WDM-based networks. In *Proceedings of INFOCOM '96*, pages 938–945. IEEE, March 1996.

[48] B. Mukherjee. WDM-Based local lightwave networks Part I: Single-hop systems. *IEEE Network Magazine*, pages 12–27, May 1992.

[49] B. Mukherjee. WDM-Based local lightwave networks Part II: Multihop systems. *IEEE Network Magazine*, pages 20–32, July 1992.

[50] K. Nakagawa, S. Nishi, K. Aida, and E. Yoneda. Trunk and distribution network application of erbium-doped fiber amplifier. *Journal of Lightwave Technology*, LT-9, February 1991.

[51] Z. Ortiz, G. N. Rouskas, and H. G. Perros. Scheduling of multicast traffic in tunable-receiver WDM networks with non-negligible tuning latencies. In *Proceedings of SIGCOMM '97*. ACM, September 1997. (To appear).

[52] D. Park, H. G. Perros, and H. Yamashita. Approximate analysis of discrete-time tandem queueing networks with bursty and correleated input traffic and customer loss. *Operations Research Letters*, 15:95–104, 1994.

[53] V. Paxson and S. Floyd. Wide area traffic: The failure of poisson modeling. *IEEE/ACM Transactions on Networking*, 3(3):226–244, June 1995.

[54] H. G. Perros and K. M. Elsayed. Call admission control schemes: A review. *IEEE Communications Magazine*, 34(11):82–91, 1996.

[55] G. R. Pieris and G. H. Sasaki. Scheduling transmissions in WDM broadcast-and-select networks. *IEEE/ACM Transactions on Networking*, 2(2):105–110, April 1994.

[56] G. Pujolle and H. G. Perros. Queueing systems for modelling ATM networks. In *Int'l Conf. on the Performance of Distributed Systems and Integrated Comm. Networks*, pages 10–12, Kyoto, Japan, September 1991.

[57] R. Ramaswami. Multiwavelength lightwave networks for computer communication. *IEEE Communications Magazine*, pages 78–88, February 1993.

[58] Hamid Ahmadi Roch Guerin, TITLE = "Equivalent Capacity Mahmoud Maghshineh", and JOURNAL = Its Application to Bandwidth Allocation in High-Speed Networks".

[59] G. N. Rouskas and M. H. Ammar. Analysis and optimization of transmission schedules for single-hop WDM networks. *IEEE/ACM Transactions on Networking*, 3(2):211–221, April 1995.

[60] G. N. Rouskas and V. Sivaraman. On the design of optimal TDM schedules for broadcast WDM networks with arbitrary transceiver tuning latencies. In *Proceedings of INFOCOM '96*, pages 1217–1224. IEEE, March 1996.

[61] G. N. Rouskas and V. Sivaraman. Packet scheduling in broadcast WDM networks with arbitrary transceiver tuning latencies. *IEEE/ACM Transactions on Networking*, 5(3):359–370, June 1997.

[62] A. Sneh and K. M. Johnson. High-speed tunable liquid crystal optical filter for WDM systems. In *Proceedings of IEEE/LEOS Summer Topical Meetings on Optical Networks and their Enabling Technologies*, pages 59–60, Lake Tahoe, CA, July 1994.

[63] W. Stewart. *Numerical Solutions of Markov Chains.* Princeton University Press, Princeton, New Jersey, 1994.

[64] K. Tada and H. S. Hinton, editors. *Photonic Switching II*, pages 292–295. Springer–Verlag, Berlin, 1990.

[65] L. Thylen, G. Karlsson, and O. Nilsson. Switching technologies for future guided wave optical networks: Potentials and limitations of photonics and electronics. *IEEE Communications Magazine*, 34(2):106–113, February 1996.

[66] F. A. Tobagi. Fast packet switch architectures for broadband integrated services digital networks. *Proceedings of IEEE*, pages 133–167, January 1990.

[67] S. Tridandapani, J. S. Meditch, and A. K. Somani. The MaTPi protocol: Masking tuning times through pipelining in WDM optical networks. In *Proceedings of INFOCOM '94*, pages 1528–1535. IEEE, June 1994.

[68] Y.-S. Yeh, Michael H. Hluchyj, and Anthony S. Acampora. The knockout switch: A simple, modular architecture for high-performance packet switching. *IEEE Journal on Selected Areas in Communications*, SAC-5(8):1274–1283, October 1987.

[69] A. O. Zaghloul and H. G. Perros. Approximate analysis of a discrete-time polling system with bursty arrivals. *Modelling and Performance Evaluation of ATM Technology (Perros, Pujolle, Takahashi Eds.)*, 1993.

Appendix A

A Discussion of the Arrival Process Approximation for the TTFR Switch Supporting Cells

The approximation of an MMBP arrival process which is split and distributed to multiple destinations by several appropriately thinned MMBP processes is not trivial. This approximation preserves the mean interarrival time distribution, the autocorrelation of interarrival times, and the autocorrelation of arrival probabilities to the individual queues, but does not preserve the same values for the arrival stream to the input port, thus affecting the occupancy probabilities of the input queues. In this section, we discuss the implications of this approximation of the arrival process on the mentioned functions..

Based on the analysis presented in [52], the interarrival time distribution to input port i in our model of the actual system may be calculated. Define $\tilde{\pi}_i$ as the steady state occupancy probability for source i (satisfying $\tilde{\pi}_i \mathbf{Q}_i = \tilde{\pi}_i$ per our previous description of MMBP's). Therefore, the state occupancy probability conditioned on a cell being generated ($\vec{p}_a$) and, as a result, the interarrival time probability distribution may be easily expressed [1].

$$\vec{p}_a = \frac{\tilde{\pi}_i \mathbf{A}_i}{\tilde{\pi}_i \mathbf{A}_i \vec{\mathbf{1}}} \tag{A.1}$$

[1]In this section, $\vec{\mathbf{1}}$ refers to an appropriately dimensioned vector of 1's.

$$p(n \text{ slot interarrival time}) = \begin{cases} \vec{p}_a(\mathbf{A}_i\mathbf{Q}_i)^{(n-1)}(\mathbf{I}-\mathbf{A}_i)\mathbf{Q}_i\vec{\mathbf{1}}, & n > 0 \\ 0, & n = 0 \end{cases} \tag{A.2}$$

For the approximation involving multiple thinned MMBP processes, however, determining the actual distribution is more complicated. This case requires the consideration of 4 separate probabilities:

- the probability that an arrival occurs at queue c exactly n slots after the last arrival to port i which also arrived to queue c;
- the probability that an arrival occurs at queue c exactly n slots after the last arrival to port i which arrived to a queue other than queue c;
- the probability that no arrival occurs at queue c at least n slots after the last arrival to port i, which was an arrival to queue c; and,
- the probability that no arrival occurs at queue c at least n slots after the last arrival to port i, which was an arrival to a queue other than queue c.

Designate these probabilities as $e_{ic}(n)$, $f_{ic}(n)$, $\bar{e}_{ic}(n)$, and $\bar{f}_{ic}(n)$, respectively.

Additionally, the state occupancy probabilities conditioned on a cell being generated ($\vec{p}_{a_c}$) and not being generated ($\vec{p}_{\bar{a}_c}$) are necessary for this evaluation.

$$\vec{p}_{a_c} = \frac{\tilde{\pi}_i r_{ic}\mathbf{A}_i}{\tilde{\pi}_i r_{ic}\mathbf{A}_i\vec{\mathbf{1}}} = \vec{p}_a \ \forall \ c \tag{A.3}$$

$$\vec{p}_{\bar{a}_c} = \frac{\tilde{\pi}_i(I - r_{ic}\mathbf{A}_i)}{\tilde{\pi}_i(I - r_{ic}\mathbf{A}_i)\vec{\mathbf{1}}} \tag{A.4}$$

Therefore, the derivations of following expressions are straightforward.

$$e_{ic}(n) = \vec{p}_a\mathbf{X_{ic}}^{(n-1)}\mathbf{Y}_{ic}\vec{\mathbf{1}} \tag{A.5}$$

$$f_{ic}(n) = \vec{p}_{\bar{a}_c}\mathbf{X}_{ic}^{(n-1)}\mathbf{Y}_{ic}\vec{\mathbf{1}} \tag{A.6}$$

$$\bar{e}_{ic}(n) = \vec{p}_a\mathbf{X}_{ic}^{n}\vec{\mathbf{1}} \tag{A.7}$$

$$\bar{f}_{ic}(n) = \vec{p}_{\bar{a}_c}\mathbf{X}_{ic}^{n}\vec{\mathbf{1}} \tag{A.8}$$

To obtain the interarrival distribution for port i in first part of equation A.9, the outermost summation is made over the queue which receives a cell (input queue c) and terminates the interarrival period being examined. The first term of this equation corresponds to the condition that input queue c also received the previous cell. The second term (containing the interior summation) corresponds to the possibility that input queue c' received the previous cell at this input port. Finally, the entire expression is evaluated for all values of c.

Simultaneous arrivals are equivalent to two input queues having an arrival exactly n slots from the previous arrival. Given that the value of n need not be specified, the expression begins with a summation over all valid values. The two inner summations run over the possible ranges of queues which may have arrivals. The four cases which may occur is that either, both, or neither of the two queues (c and c') may or may not have been the last queue(s) to have received an arrival. Furthermore, all of the intervening queues (those between c and c') do not receive cells during the n slots even though they may or may not have received the last arrival to the switch.

$$p(n \text{ slot interarrival time}) = \begin{cases} \sum_{c=1}^{C}[e_{ic}(n)\prod_{\dot{c}\neq c}\bar{f}_{i\dot{c}}(n) + \\ \quad \sum_{c'\neq c}[f_{ic}(n)\bar{e}_{ic'}(n)\prod_{\dot{c}\neq c,c'}\bar{f}_{i\dot{c}}(n)]], & n > 0 \\ \sum_{n=0}^{\infty}\sum_{c=2}^{C}\sum_{c'=1}^{c-1}[(e_{ic}(n)+f_{ic}(n))(e_{ic'}(n)+f_{ic'}(n))\times \\ \quad \prod_{\dot{c}=c'+1}^{c-1}(\bar{e}_{i\dot{c}}(n)+\bar{f}_{i\dot{c}}(n))], & n = 0 \end{cases} \tag{A.9}$$

It should be noted that these expressions simplify considerably if r_{ic} is constant $\forall\ c$.

In order to evaluate the autocorrelation coefficient for the port's interarrival times in our approximate system, the definition of the state of the arrival processes is necessary. The state of the arrival processes may be defined using a C-dimensional tuple, $(k_1, k_2, \cdots, k_C)$, such that k_c represents the state of the single arrival process which corresponds to the cth queue of the input port. Since the separate arrival processes in the approximate system operate independently, the probability of the arrival processes occupying a single aggregate state may be easily calculated. If the value of the cth element of the vector $\vec{p}_a$ calculated in Equation A.1 can be found by the expression $\vec{p}_a\mathbf{1_c}$ [2], the probabilities of state occupancy given the arrival of a cell to the input port in the approximate systems

[2] $\mathbf{1_c}$ denotes a transposed vector of appropriate dimension which contains a single non-zero element. The element's value is 1.0 and is the cth element in the vector.

may be defined as $\vec{p}_A$.

$$\vec{p}_A = \prod_{c=1}^{C} \vec{p}_a \mathbf{1}_{\mathbf{k_c}} \tag{A.10}$$

Appendix B

An Alternative Input Queue Analysis for the TTFR Switch Supporting Packets

In this section, we consider the case of fixed-size packets in the TTFR switch presented in Chapter 6. As will be seen, the computational aspect of the input queue's analysis may be considerably simplified. This simplification is possible because it suffices to only consider the Markov chain $\mathbf{S}_{ic}$ at boundaries of the arrival slots for which either $v_{ic}(x) > 0$ or $v_{ic}(x \oplus 1) > 0$. This method cannot be used for the case involving varying packet lengths since the calculation of the probability of loss for the input queue (presented in the next section) is dependent on the order of arrivals. As will be seen, this approach does not impact the analysis of the output queues.

First, the traffic model is modified to define the following arrival state descriptors.

$$\hat{\mathbf{Q}}_{ic} = \begin{array}{c} \left[\begin{array}{ccccc} q_{ic}^{(0,0)} & 0 & \cdots & 0 & 1 - q_{ic}^{(0,0)} \\ q_{ic}^{(0,0)} & 0 & \cdots & 0 & 1 - q_{ic}^{(0,0)} \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{array} \right] \end{array} \begin{array}{c} 0 \\ 1 \\ 2 \\ \vdots \\ T \end{array} \qquad \text{(B.1)}$$

$$\hat{\mathbf{A}}_{ic} = \mathbf{A}_{ic} \qquad \text{(B.2)}$$

The order of events is defined as in Figure 6.3. Transitions at all observed instants except the first observed instant during a frame occur with the probabilities shown in Table 6.1. Transitions probabilities between states (x, y, z) and $(\hat{x}, \hat{y}, \hat{z})$, (the period during which the corresponding input queue receives no service), are defined in Table B.1. Notice first that there are $M - (x - \hat{x})$ arrival slots between slots x and $\hat{x}$ (non-inclusive). During those slots, $\hat{y} - y$ segments arrived. There is no service during the first observed slot in this scheme; it is not until the second observed slot that service is witnessed. Additionally, given that the system was left in state z and returned in state $\hat{z}$, $\min(M - (x - \hat{x}), z)\mathbf{I}_{z>0} + \max(\min(M - (x - \hat{x}) - z, \hat{z}), 0)\mathbf{I}_{\hat{z}>0}$ segments must have occurred[1]. It should be noted that it is not possible for this number to be greater than $\hat{y} - y$. Furthermore, this number of segments will necessarily require $\min(M - (x - \hat{x}), \mid z \mid) + \max(\min(M - (x - \hat{x}) - \mid z \mid, T - \mid \hat{z} \mid), 0)$ slots of the queue's idle period.

The question now becomes "What is the probability of receiving exactly $\nu(\cdot)$ packets during $t(\cdot)$ slots?"[2] Obviously, $\nu(\cdot)$ packets is equal to $T\nu(\cdot)$ segments which may arrive during an equal number of arrival slots. This observation implies that there are exactly $t(\cdot) - T\nu(\cdot)$ idle arrival slots. Therefore, during the t slots, there exists exactly $\nu + t(\cdot) - T\nu(\cdot)$ instants during which the arrival process' state machine may potentially enter the "Idle" state. From (B.1), an idle arrival slot occurs with probability $q_{ic}^{(0,0)}$ given that the system is at an appropriate instant. Finally, to answer the above question, the probability of receiving exactly $\nu(\cdot)$ complete packets during $t(\cdot)$ slots is given as $\theta(\nu(\cdot), t(\cdot))$ in (B.3).

$$\theta(\nu(\cdot), t(\cdot)) = \begin{pmatrix} \nu(\cdot) + t(\cdot) - T\nu(\cdot) \\ t(\cdot) - T\nu(\cdot) \end{pmatrix} \times (q_{ic}^{(0,0)})^{(t(\cdot) - T\nu(\cdot))} \times (1.0 - q_{ic}^{(0,0)})^{\nu(\cdot)} \tag{B.3}$$

$$\nu(x, \hat{x}, y, \hat{y}, z, \hat{z}) = \begin{cases} \hat{y} - y - z\mathbf{I}_{z>0} - \hat{z}\mathbf{I}_{\hat{z}>0}, & M - a_{ic} \geq 2T \\ \frac{\hat{y} - y - (\min(M - (x - \hat{x}), z)\mathbf{I}_{z>0} + \max(\min(M - (x - \hat{x}) - z, \hat{z}), 0)\mathbf{I}_{\hat{z}>0})}{T}, & \text{otherwise} \end{cases} \tag{B.4}$$

[1]The complexity of this expression is necessary in order to account for unusually short frame sizes. Usually, the idle period (whose only activity during this period is arrivals, for which the corresponding probabilities may be computed directly) will be long enough for both the packet which may be seen at the beginning of the idle period to complete its arrival (accounting for the completion of the packet seen in z) as well as another full packet (which may be seen in $\hat{z}$). The simplifications associated with this situation are shown in (B.4) and (B.5).

[2]The formal arguments of these functions are omitted from the text of this discussion for the sake of clarity. They are appropriately shown in (B.4) and (B.5).

Table B.1: Transition probabilities out of state (x, y, z) of the Markov chain for fixed packet length

Current State	Next State	Transition Probability
(x, y, z)	$(\hat{x}, \hat{y}, \hat{z})$	$\theta(\nu(\cdot), t(\cdot) \times \mathbf{I}_{B_{ic}-\hat{y}\geq T}$
(x, y, z)	$(\hat{x}, \hat{y}, \hat{z})$	$\sum_{\nu'=\nu(\cdot)}^{\nu_{max}} \theta(\nu', t(\cdot)) \times \mathbf{I}_{B_{ic}-\hat{y}<T}$

$$t(x, \hat{x}, z, \hat{z}) = \begin{cases} M - (x - \hat{x}) - (\mid z \mid + (T - \mid \hat{z} \mid)), & M - a_{ic} \geq 2T \\ M - (x - \hat{x}) - [\min(M - (x - \hat{x}), \mid z \mid) + & \\ \quad \max(\min(M - (x - \hat{x}) - \mid z \mid, T - \mid \hat{z} \mid), 0)], & \text{otherwise} \end{cases} \tag{B.5}$$

This expression is used for the first set of transitions in Table B.1, as it does not account for possible losses due to overflow during the queue's unserviced period.

In accounting for losses, the same calculations may be used; however, $\hat{y}$ is, in fact, equal to some $\hat{y}' - T\nu'$ where ν' is the number of complete packets which were discarded during the period which is not being explicitly modeled. Therefore, it is easily seen that the actual number of segments which arrived to the queue may be any value between that number observed ($\nu(\cdot)$) and the maximum number which may be received during the unserviced period ($\nu_{max} = \lfloor \frac{t(\cdot)}{T} \rfloor$ packets). Accumulating the probabilities for each of the valid values within this range yields the expression shown in the second row of Table B.1.

These transition probabilities may be used to build a smaller version of $\mathbf{S}_{ic}$ containing only $M - (x - \hat{x})$ states. This version of $\mathbf{S}_{ic}$ may then be used to solve for the relevent values of $\pi_{ic}(x, y, z)$.

Appendix C

A Heuristic for Estimating Packet Transition Probabilities for the TTFR Switch Supporting Packets

As we mentioned in Section 6.3, the output queue analysis may be significantly affected by the introduction of justifiable heuristics for the functions $\mathcal{P}(\cdot)$, $\mathcal{F}(\cdot)$, and $L_i(\cdot)$. In this appendix, we demonstrate the impact of such a set of heuristics.

In these experiments, we use the same switch, schedules, and arrival processes given in Chapter 6. In our analytic calculations, however, $L_i(\cdot)$ is modified per Equation C.1.

$$\begin{aligned} L_i(s_i \mid x) &= r_{cj} f_{ic}(s_i) \times p_{ij}(\text{packet generated } \mid x) \\ &\quad \times \textstyle\prod_{x' \in \mathcal{A}_{ij}^{(s_i)}(x)} (1.0 - p_{ij}(\text{packet generated } \mid x')) \end{aligned} \tag{C.1}$$

$$\begin{aligned} p_{ij}(\text{packet generated } \mid x) &= M \textstyle\sum_{\zeta=0}^{T-1} \Phi_{ij}(\zeta, x) \times \sum_{\zeta_{add}=1}^{v_{ic}(x \oplus 1)} \\ &\quad \left\{ \textstyle\sum_{\zeta'=0}^{\zeta_{add}-1} \mathcal{P}(\zeta_{add} \mid x) \mathcal{F}(\zeta - \zeta' + \zeta_{add} \mid \zeta) \right\} \end{aligned} \tag{C.2}$$

The set $\mathcal{A}_{ij}^{(s_i)}(x)$ is the set of arrival slots during which re-assembly queue (i, j) may receive segments and which encompasses exactly $s_i - 1$ service slots prior to arrival slot x. Depending on the values involved, the same arrival slot may need to be considered multiple times, based on its being encountered in different frames. This set is used to condition

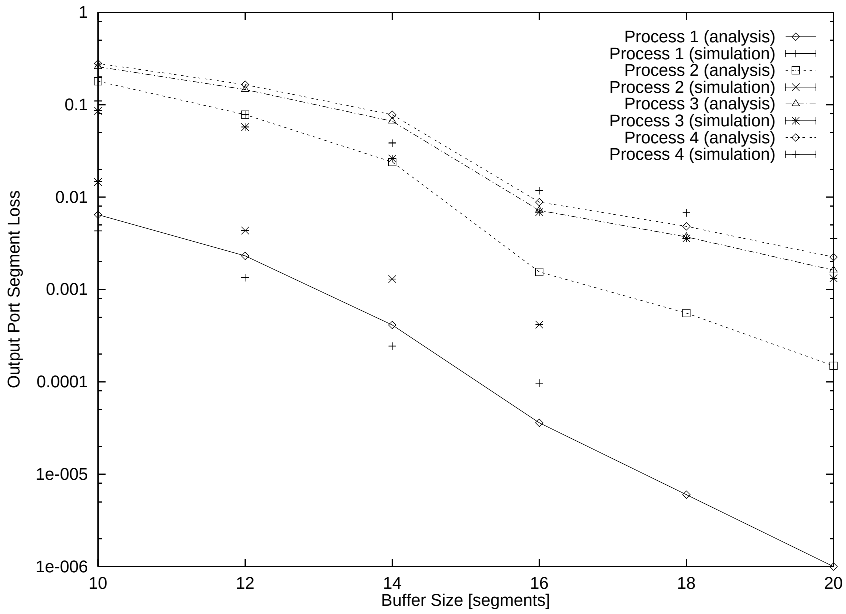

Figure C.1: Output Port Segment Loss Probability ω_1 for $C = 2$

the fact that an arrival is occurring during slot x on the fact that no arrival has occurred during the previous (appropriate) s_i service slots. By modifying the definition of $L_i(\cdot)$, we have attempted to preserve the inter-arrival distribution of packets by modifying the arrival process of packets arriving to the output queue.

This formulation for $L_i(\cdot)$ obviously produces more accurate results than that given in (6.21). The sharp change in loss probabilities as B increased from 14 to 16 is still apparant, but rather than the loss at $B = 16$ being approximately 4% of that at $B = 14$ (as shown in Figures 6.11 and 6.12), the loss at $B = 16$ is approximately 27% that at $B = 14$ (see Figures C.1 and C.2). The corresponding simulation results shown in both sets of figures shown the change in loss between the two buffer values being 36%.

The goal of this heuristic was to provide a formulation for $L_i(\cdot)$ which consistently overestimated the simulated loss values. This goal was not accomplished in the strictest sense: for instance, we observe underestimation at $B = 16$ while using Process 1 and two wavelengths on the order of 35% (i.e., the analytically determined values were 35% of the simulated values). While this error could be attributed to non-convergence by the simulation (recall that the simulation results are based on 30 runs of 100,000 service slots each), Figure C.3 shows that, with three wavelengths and $B = 18$ and using Process 4, errors of 28% were

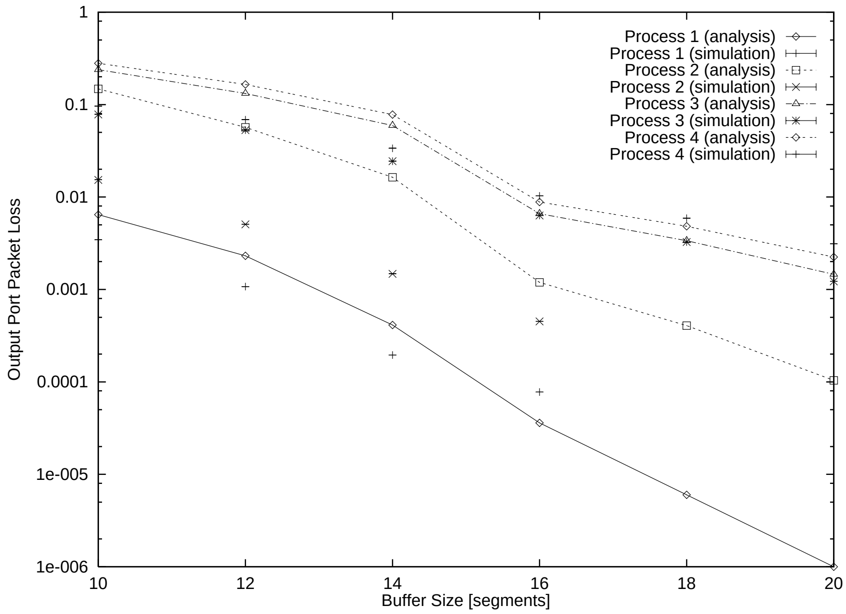

Figure C.2: Output Port Packet Loss Probability Ω_1 for $C = 2$

found. Observe also, though, the *decrease* in the error (the absolute difference between the values found via simulation and analysis) on either side of this value of B; this decrease (seen both for $C = 2$ and $C = 3$) would seem to indicate that the error is not necessarily diverging and needs to be further explored for larger buffer sizes. With only these two sets of points showing underestimation and a mean absolute relative error of 7.6% (compared to 19.5% for the found in the Figures 6.11 through 6.14)[1], this heuristic seems to provide a much better bound than that presented in Chapter 6.

[1]The mean absolute relative error is taken to mean the average absolute relative errors for all 96 pairs of points shown in the two sets of four figures: 6.11 through 6.14 and C.1 through C.4.

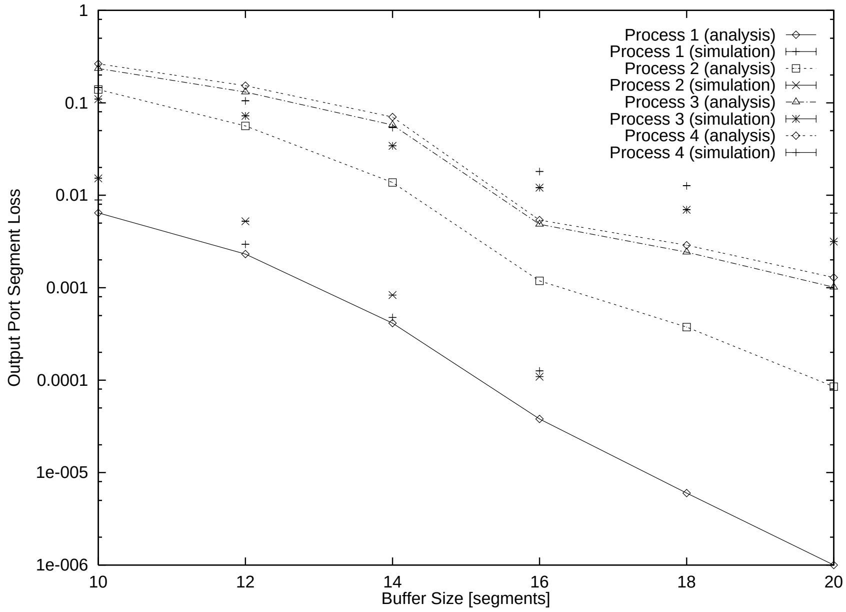

Figure C.3: Output Port Segment Loss Probability ω_1 for $C = 3$

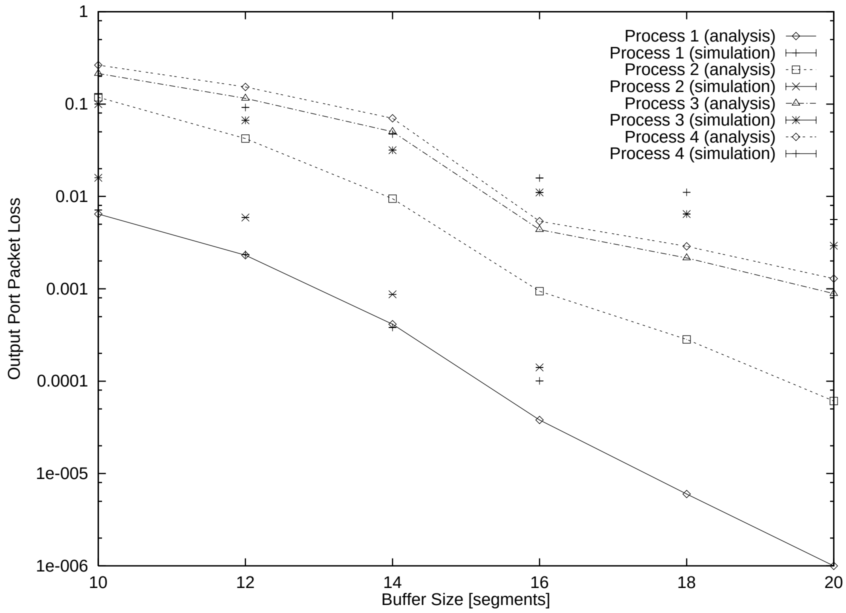

Figure C.4: Output Port Packet Loss Probability Ω_1 for $C = 3$