Design and Analysis of High Performance Multicast Time-Multiplexed Switches

A thesis submitted in partial fulfillment of the
requirements for the degree of Master of Philosophy
in the Department of Electronic Engineering
of The Hong Kong Polytechnic University

By

Chan Man-Chi, BEng(Hons)

1998

Acknowledgments

The author wishes to express his gratitude to his supervisor, Dr. P. C. K. Liu, for his invaluable advice, constant guidance and encouragement during the period of study. Thanks are also due to Dr. K. C. Li and Dr. K. T. Lo for their support and advice on the project. He also thanks the academic staff of Department of Electronic Engineering, The Hong Kong Polytechnic University, for their concerns and technical supports. In addition, it is the author's pleasure to acknowledge the inspiring discussions with his fellow students.

Abstract

This thesis focuses on the time slot assignment (TSA) problems associated with multicast time multiplexed switches (TMS). It begins with a brief review of the problems and existing solutions. Based on the review, existing methods in solving these problems are systematically classified and organized according to their pros and cons. Having recognized the demand for more promising approach to tackle the TSA problems, especially, for the multicast mixed traffic, the project aims to study how a multicast TMS can be designed to overcome these problems. By understanding the nature of the problems and the shortcomings of the existing methods, an effective switching technique, namely, zone switching, is proposed as a hint for designing a high-performance multicast TMS. The idea of zone switching is to provide more rooms for call scheduling via a proper switch design. The switch using this technique in principle can eliminate most of call blocking inherently without the need of sophisticated control algorithm, which would limit the growth of switch in terms of size and speed. To implement this switching concept, two efficient switch architectures are developed for multicast TMS. The first design has a demux-mux architecture, which is a building block for cross-connect nodes in various broadband communication systems. By exploiting trunk grouping at both input and output ports, the demux-mux switch can remove most slot contentions and achieves better performance. The second one is a simple multistage switch. It is designed to implement a novel scheduling scheme, window scheduling, which is the simplest and an efficient way to implement zone switching. Since the multistage switch consists only of several switching planes and a set of delay elements, it is very suitable for using in an all-optical network (AON). Although zone switching has very outstanding

performance, it does not work well when one or more involved lines are heavily loaded. Another efficient design technique, internal bandwidth expansion, is thus proposed. As illustrated from its name, this technique reduces the level of slot contention by increasing the internal bandwidth, which in turn provides more alternatives to connect a call request. Based on this concept, a time-space-time (TST) switch with intermediate line dilation is developed. This switch has internal bandwidth doubled and can eliminate almost all the slot contentions. However, the tradeoff may be the requirement of buffer stages at both input and output time slot interchangers (TSI). This greatly reduces the feasibility of implementing it in lightwave systems. Indeed, multicast TMS designed based on either one of these techniques can achieve substantial performance improvement. For cost-efficient design, it is suggested to use internal bandwidth expansion, if its implementation is feasible. Otherwise, zone switching is a good alternative to provide the same extent of improvement. Throughout this thesis, the switch performance is measured as the average call blocking probability and is evaluated via purely analytical means. The analytical results are each comparable to those predicted via computer simulations, which shows that the analytical models are valid for use.

Table of Contents

List of Figures

Figure **Page**

List of Abbreviations

AON	All-Optical Network
ATM	Asynchronous Transfer Mode
B-ISDN	Broadband Integrated Service Digital Network
ITU	Internal Telecommunication Union
MMBB	Mass Market Broadband service
QoS	Quality of Service
TDM	Time-Division Multiplexed
TMS	Time-Multiplexed Switch
TSA	Time Slot Assignment
TSI	Time Slot Interchanger
TST	Time-Space-Time switch

List of Symbols

$\binom{m}{n}$	Combination function of selecting n out of m iterms
F	Frame size (in terms of the number of time slots in it)
t_{ij}	Number of time slots requested by input port i to output port j per frame
T	Request matrix (traffic matrix)
N	Total number of input (output) ports in a switch
$O(\cdot)$	Computation time complexity function
z	Number of lines per input (output) trunk in a demux-mux switch
f	Sub-frame size (in terms of the number of time slots) in a demux-mux switch
λ	Average arrival rate
μ	Exponential mean of the holding time
D	Average call fanout (average number of destinations of a call)
B	Average call blocking probability
w	Window size used in window scheduling (or the number of switching planes in a multistage switch)

Chapter 1 Introduction

The next generation of networks will integrate services for different kinds of applications, which can be classified into delay-insensitive asynchronous applications like fax, mail and file transfer, and delay-sensitive applications with real-time requirements, such as audio and video. To support such a wide variety of applications efficiently, future public switched networks must be developed in an application-independent manner [TURN 86], [PRYC 91]. Particularly, they must be able to support applications of different connectivity requirements (e.g., point-to-point, point-to-multipoint, and broadcast connections).

To date, time-division multiplexed (TDM) technique has widely been employed in both satellite communications [SCAR 83] and terrestrial networks [JOEL 79] because of its simplicity and guaranteed high quality of service (QoS). Although asynchronous transfer mode (ATM) is recommended by the International Telecommunication Union (ITU) as the transfer mode for implementing broadband integrated service digital network (B-ISDN), a lot of work has still to be done for the realization of the B-ISDN based on ATM. During the transition period from ISDN to B-ISDN, integrating broadband services in the existing TDM systems seems to be the best and the only strategy to meet the oncoming market demands. A high performance multicast time-multiplexed switch (TMS) which routes time-multiplexed traffic is hence important to support such kinds of applications.

1.1 Motivation

In a TMS, each input or output port carries multiplexed signals from several sources. These signals are divided into repetitive frames, each of which is further partitioned into a fixed number of time slots. A time slot is the basic unit of connection and a point-to-point call of basic data rate is established by associating an empty slot (idle slot) on the input frame and another on the output frame in the same temporal position. The time slot assignment (TSA) problem associated with TMS is to schedule a call to the matched idle slots in all the requested frames.

Recently, much of the research in solving such kinds of TSA problems has been focused on finding a conflict-free assignment of traffic-units to time slots such that the frame-length is minimized [INUK 79], [BONG 81a], [GOPA 82b], [BONU 89], [CHAL 90]. These TSA algorithms provide conflict-free assignments for any new request not violating capacity constraints and hence can fully realize the potential performance of a TMS. They are, however, computationally intensive and always involve the rescheduling of existing call connections. So that they are not commonly used in current switching systems. Therefore, some promising approaches for TSA (call scheduling) with no connection rearrangement were then developed [KIM 92a]. These algorithms target to minimize the level of constraint imposed on the later call connections by finding a suitable TSA for each new call. Since they only concern with the insertion of a new request to an existing schedule without any rearranges, the control structure of a switch can be much simplified. The trade off is higher call blocking rate, which is due to the slot contention with existing calls.

In the recent years, the design issue of high-performance packet switch has been widely studied in the literature and many promising switch designs were developed for multicast traffic [HUAN 84], [LEE 88], [TURN 88]. Their excellent performances give an insight that a good switch design may be one promising method to solve TSA problem in TMS. It would be desirable to study and understand how a multicast TMS can be designed to overcome these problems.

1.2 Objectives of the thesis

This thesis reported the recent research works in solving the TSA problem for various TDM switching systems and proposed two efficient design methods, namely, zone switching and internal bandwidth expansion, to remove slot contention effectively in a TMS. To implement these methods, three cost-efficient switch architectures for multicast TMS are considered. The performance of multicast circuit-switched traffic in them are evaluated via purely analytical means, while working with a random scheduling algorithm.

1.3 Organization of the thesis

The rest of the thesis is structured as follows. Chapter 2 introduces the basics of TDM switching system and the general problems associated with it. Existing solutions to these problems are discussed and two efficient methods, zone switching and internal bandwidth expansion, are then proposed for the design of a high-performance multicast TMS.

In Chapter 3, 4 and 5, based on our design methods, three efficient switch architectures for multicast TMS are developed and presented. The switch designed in Chapter 3 has a demux-mux architecture, which makes use of trunk grouping at both input and output ports to reduce the level of slot contention. The one presented in Chapter 4 is a multistage switch. It is proposed to implement a novel call scheduling scheme, window scheduling, in an all-optical network (AON). The final design in

Chapter 5 is a time-space-time (TST) switch with intermediate dilation. It takes the advantage of internal bandwidth expansion to achieve near-optimal performance. In this case, the switch performance is measured as the average call blocking probability and is evaluated via the arrival modulation technique in [LUND 48] and [SYSK 60].

Chapter 6 summaries the work reported in this thesis and comments on the efficiency of our proposed techniques. Finally, the thesis will be concluded with identifying a number of issues for further investigations.

Chapter 2 Overview of TSA Problems in Multicast TMS

In this chapter, the basics of TMS are detailed and the TSA problems in it are elaborated. Several promising approaches in solving these problems are reviewed and discussed. Based on our understanding, two design methods are proposed to remove call blocking effectively.

2.1 Basics of TMS

A TMS routes time-multiplexed traffic from input ports to output ports. In this system, each link is logically partitioned into frames each of which contains time slots of duration equal to one packet transmission time through a non-blocking switching fabric. In this case, all packets are assumed to be of equal size. The operation of the switching fabric is then synchronized in time slots and is carried out in such a way that at most one packet can be transmitted from an input port to a set of output ports during each time slot. The interconnection patterns between different input ports and output ports are repetitive every frame but change from slot to slot within a frame. A time slot connection in a TMS therefore constitutes a circuit of some fixed capacity in the system. This arrangement remains in effect until the user no longer needs the circuit. Figure 2.1 shows the general structure of TMS which consists of a central non-blocking multicast switch with N input ports and N output ports wherein each

input/output frame comprises F slots. TMS having this structure is termed as a simplex TMS or a simplex switch for short.

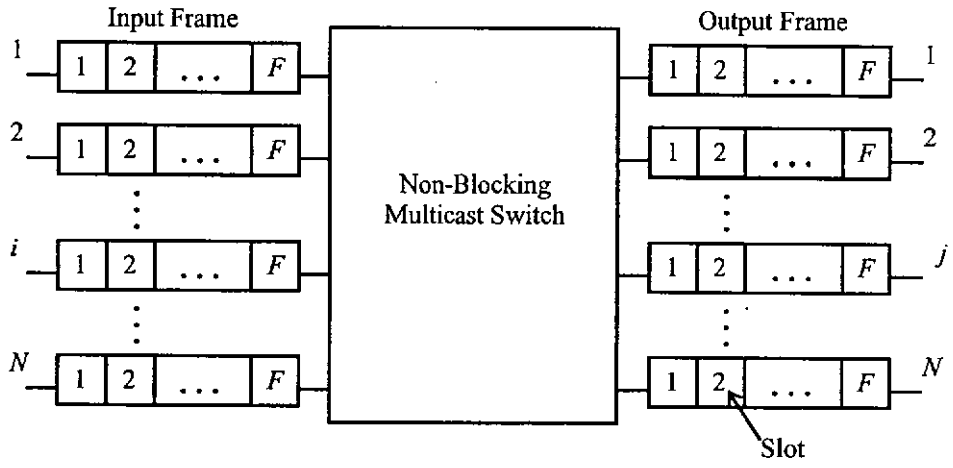


Figure 2.1 The general structure of TMS.

An inherent problem in time-multiplexed switching is that two incoming packets from different input ports cannot be destined for the same output port simultaneously, otherwise it would result in the loss of the packets or in call blocking. To circumvent

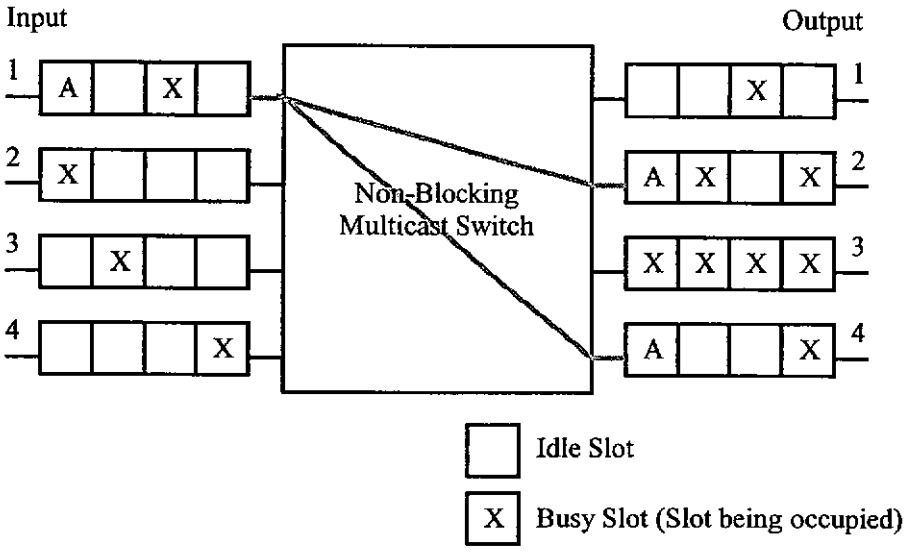


Figure 2.2 Example of call connection (scheduling).

this problem, a TMS makes use of a time slot interchanger (TSI) in each input line to schedule calls (packet arrivals) in some appropriate time slots on each frame among users, in order to conform with corresponding interconnection patterns executed by it [KIM 96]. For example, to set up a point-to-point call, a circuit switched connection is established by associating two idle slots coincided in time in the addressed input frame and output frame. Whereas, the establishment of a connection for a multicast call requires idle slots available at the same position both in the input frame and in every requested output frame. In general, a multicast call always encounters higher blocking probability than a point-to-point call because of the increased output port contention. Figure 2.2 shows the example of connecting a call A from input #1 to output #2 and #4 through the first slot.

In a TMS, assignment of time slots for a call is always constrained by the availability of input and output time slots. Considering the TMS in figure 2.1, which has N input ports and N output ports, and each input (output) frame consists of F slots. Let $T = [t_{ij}]$ be a request matrix (traffic matrix) where t_{ij} is the number of time slots requested by input port i for transmission to output port j per frame. Given a call request from an input port i to D output ports, $j_1, j_2, \cdots, j_D$, the capacity constraints for it are

$$\sum_{k=1}^{N} t_{ik} < F, \tag{2.1}$$

and

$$\sum_{k=1}^{N} t_{kj_m} < F, \quad m = 1, \cdots, D. \tag{2.2}$$

These conditions correspond to the constraints on the row and column sums of $\boldsymbol{T}$, where no row or column sum of $\boldsymbol{T}$ may exceed F, and guarantee that at least one idle slot exists in the input frame and in the every requested output frame. *Overflow blocking* is call blocking due to the lack of idle slots in one or more of involved input (output) ports [KIM 92a].

It has previously been shown that if $\boldsymbol{T}$ satisfies the capacity constraint then it is possible to assign time slots without conflict. Considering the insertion of a single request to a traffic matrix $\boldsymbol{T}$ which satisfies the capacity constraint. If this new request causes the capacity constraint to be violated then it must be dropped. If not, then the call can be scheduled, although the scheduling may involve rearrangement of calls already in progress. However, the capacity constraint is not sufficient for scheduling multicast calls due to the requirement of connecting the input port to all the requested output ports in an identical time slot. Even though idle slots are available both in the input frame and in every requested output frame, a multicast call still can be blocked when these idle slots are not matched properly. This type of blocking is *slot contention blocking*, which is due to the mismatching of idle slots in the requested input and output frames even each of them has at least one idle slot [KIM 92a]. It significantly degrades the switch performance, especially, when connection rearrangement is not allowed. In the previous example (shown in figure 2.2), a call B connecting input #1 to output #2 and #3 encounters overflow blocking due to the saturation of busy slots in output frame #3. On the other hand, a call C connecting input #2 to output #1 and #2 faces slot contention blocking because of the mismatched idle slots.

2.2 Overviews of TSA problem

In a TMS, TSA is faced by either overflow blocking or slot contention blocking. The first one is due to the capacity constraint and is unavoidable. The TSA problem here is thus to connect a call without slot contention.

Recently, there have several promising approaches in solving such kind of TSA problem in various TDM switching systems [ACAM 78], [BONG 81a], [BONG 81b], [BONU 89]. Most of them are algorithm based and can be divided into two categories depending on whether call rescheduling is involved or not. In the literature, the ones involving connection rearrangement are always termed as *TSA algorithms*, while the others are called *call scheduling algorithms*. For the sake of clarification, we refer these names to the corresponding types of algorithms in the later descriptions. They will be discussed separately in the following two sub-sections.

2.2.1 TSA algorithms

TSA algorithms have been widely investigated for various TDM switching systems. The objective of them is to find a sequence of switching configurations to transmit packets in each frame such that the overall transmission duration is minimum. A switching configuration is a set of switch settings by which packets can be transmitted without conflicts. In principle, TSA algorithms provide an assignment for any new request not violating capacity constraints and can fully realize the potential performance of a TDM switching system. Thus, they are always termed as optimal scheduling.

Recently, several best-known TSA algorithms were presented in the literature [INUK 79], [ROSE 89]. They use different methods to find a conflict-free switching configuration. For example, some TSA algorithms find a switching configuration by optimizing various max-min flow problems [GOPA 82a], [LEWA 83], [RAMA 84], while some compute it based on the neural network model [FUNA 94]. For clarity in comparing their performance, TSA algorithms in this case are divided into three classes; sequential TSA algorithms, parallel TSA algorithms and adaptive TSA algorithms. The general working principle of these classes will be detailed and discussed in the following paragraphs together with their pros and cons. It should be noted that most of these TSA algorithms can realize the potential performance of a TMS. The performance comparison between them is thus in terms of time complexity.

A. Sequential TSA algorithms [INUK 79], [BONG 81a], [BONG 81b], [GOPA 82a], [BONU 89], [LIEW 89], [CHAL 90]

This class of TSA algorithms is inherently iterative and sequential in nature. Although most of them run in polynomial time, their time complexities, which equal $O(N^5)$ or $O(N^{4.5})$, are still so high that they are not very suitable for using in many practical applications. This may limit the growth of the switch size and its speed.

B. Parallel TSA algorithms [ROSE 89], [CHAL 93], [CHAL 94], [FUNA 94]

Parallel proposal is one promising approach for a fast solution to the TSA problem. It makes use of special-purpose parallel processors to reduce the time spent in finding a switching configuration, in order to overcome the computational bottleneck. The major problem, however, is the requirement of a large number of processors, which increases with (N^2), where N is the number of ports. This may incur extra cost of the switch in a TDM system.

C. Adaptive TSA algorithms [VARM 92], [CHEN 95]

Contrary to the former two classes, in which much effort is wasted in recomputing the entire TSA for each frame, adaptive TSA algorithms generate an optimal TSA for the traffic demands in the current frame by suitably changing the known, optimal TSA for the previous frame. Intuitively, these adaptive algorithms will be faster than previous TSA algorithms with interdependence traffic in which the change of traffic demands between any two consecutive frames is always small in amount.

Intuitively, adaptive TSA algorithms have better performance than the other two classes of algorithms whenever the traffic is inter-frame dependent, which is expected to be the case in many practical applications. Because they would modify some existing connections in computing an optimal TSA for the current frame, many existing calls in such a case can be interrupted. Thus they are not popularly used in current switching systems.

2.2.2 Call scheduling algorithms

In contrast to TSA algorithms, call scheduling algorithms are targeted to minimize the level of constraint imposed on later call connections by finding a proper TSA for each new call. As it only concerns with scheduling a new request over existing switching configurations without rearranging them, the control structure of a switch can be much simplified, which facilitates the growth of the switch size and speed.

Recently, the performance of optimal scheduling and random scheduling for unicast traffic have been addressed by Brata [BART 84] and Rose [ROSE 87], where random scheduling is the simplest call scheduling algorithm whereby a new request may be granted by randomly selecting a set of idle slots coincided in time on each addressed lines. If such a kind of matched slot does not exist, the call will be blocked or lost. Their results showed that the performance improvement from the optimal scheduling over the random scheduling is not very significant, especially, when there is a larger number of connections time-division multiplexed on the data path of the switch. However, for the case of multicast traffic, Kim and Lee found that the performance gap between them increases substantially with the average call fanout [KIM 92a]. This large gap points to the fact that slot contention is a major cause for most multicast call blocking when calls are placed randomly into frames. This indicates the need of efficient call scheduling algorithms for multicast switches.

In order to reduce slot contention blocking, Kim and Lee developed two classes of call scheduling algorithms, *call packing algorithms* and *call splitting algorithms*, to place multicast calls into time slots effectively [KIM 92a]. These

algorithms are all simple in control and can demonstrate performance improvement over the random scheduling all the time. To have a clear understanding on these algorithms, a brief discussion for them is given as follows.

A. Call packing algorithms (e.g. First Fit, Best Fit, Worst Fit)

Call packing algorithms are analogous to the storage placement strategies of computer operating systems [CALI 82]. They are based on several criteria, for example, first find first use, to determine the idle slot searching sequence for each call connection, in order to reduce the potential slot contention for the later call requests. Although they are simple in control, they do not effectively improve the performance of a multicast switch [KIM 92a].

B. Call splitting algorithms (e.g. Fixed Splitting, Greedy Splitting)

Call splitting algorithms is one promising approach to remove slot contention. It partitions the original outputs of a multicast call and generates a number of sub-calls with smaller output subsets. Because sub-calls have fewer outputs, it is easier to establish a connection for each sub-call, and call splitting algorithms may effectively improve the performance of multicast switch. The problem of it, however, is the additional load introduced to the input port which is due to the excessive call splitting, and could degrade the switch performance. Although a simple adaptive call splitting algorithm, greedy splitting (GS), proposed by Kim and Lee, can overcome this problem and achieves near-optimal performance, the price is additional complexity in the structure of a switch. For instance, the input ports must provide extra buffers to

store all sub-packets generated from the splitting of multicast packet. Thus the splitting of multicast calls is not always very suitable.

In summary, call scheduling algorithms seem not to be the best approach to reduce the call blocking probability in a TMS. It would be desirable to develop a new approach to solve this problem.

2.3 Design of Multicast Switch

In principle, a proper switch design provides more alternatives for call connection and reduces call blocking inherently without using any sophisticated algorithm, which may limit the growth of the switch size and its speed as well. Recently, there has rigorous research on the issue of switch design for various multicast traffic. Many promising switch structures, such as the Starlite system designed by Huang and Knauer [HUAN 84], the broadcast packet switch developed by Turner [TURN 88], and the multicast packet switch proposed by Lee [LEE 88] etc., were then developed. Their outstanding performance and strong support to the flexible multicast transmission give us an insight that a good switch design may be the most promising approach to solve the TSA problem in a multicast TMS.

Recall that in a TMS, a call of basic rate is established by connecting the requested ports via a non-blocking switching fabric in an identical time slot. When a

new request is arrived but the scheduler cannot find any conflict-free circuit (which consist of idle slots coincided in time on each addressed frame) for it, the call is blocked or lost, assuming no call rescheduling is allowed. Actually, such kind of one-shot switching operation is the major cause for most call blocking. It introduces a certain extent of restrictions on call scheduling. To relieve these restrictions, new approaches to switch designs will be required.

In this thesis, we propose a new switch design concept, *zone switching*, to overcome the problem. In zone switching, packets from a multicast call each arrives the outputs in one of a set of identical time slots in a frame. In other words, a circuit here is composed of idle slots at different requested lines, which are no longer necessary to be matched in position but distributed within a set of identical positions (switching zone). The connection constraint therefore will be substantially reduced. This idea is similar to that of call splitting algorithms where a multicast call is divided into several sub-calls each of which connects the input to a sub-set of the addressed outputs to reduce the number of conflicts in every call establishment [KIM 92a]. The difference is that with zone switching, a call can be established once the requested output zones are available even though there is only one input idle slot, in this case it is impossible to split a call into any sub-calls.

Switching zone in zone switching is a defined set of slot positions in every TDM frame with respect to a particular input idle slot, where a new call connecting this input can be established if one or more idle slots are available in each requested output frame at these positions. Different input idle slot may have different switching zone. The distribution of switching zone is much depended on the switch design.

Figure 2.3 describes one way to implement zone switching. In this example, there is no matched idle slot between the input frame and all the addressed output frames. Thus, a call D connecting these frames will be blocked in a simplex multicast switch. However, assuming a switch is well-designed such that giving an input idle slot, the switching zone of it in each request output frame is the two successive slot positions counting inclusively from it. As a result, the call is connected. This shows the potential of zone switching to improve switch performance.

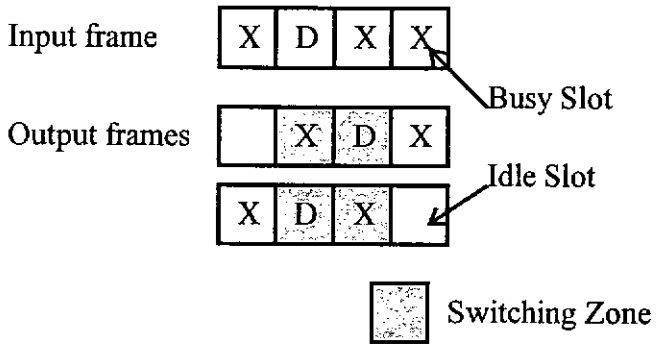


Figure 2.3 One way to implement zone switching.

Although zone switching has very high potential to remove slot contention, it does not work efficiently when some of requested lines are heavily loaded. In this case, it is hard to find any conflict-free TSA for a new request because of severe slot contentions. To overcome this problem, another efficient strategy, internal bandwidth expansion, is developed to reduce slot contention effectively.

Time-multiplexed switching is always implemented by a three-stage time-space-time (TST) interconnection network [KIM 96]. In the input "T" stage, each TSI exchanges time slots between incoming input frame and outgoing internal frame(s). After switching by a space-division switch, output internal frame(s) will be collected

by the corresponding output TSI, in which occupied slots (packets) in each internal frame will be extracted to form an output frame. The internal bandwidth here is the number of time slots in the internal frame(s) connecting to each input (output) port [HOU 96]. Similarly, the external bandwidth is the number of slots on each input (output) frame. Giving the number of busy slots in an input (output) frame is fixed, the increment in the internal bandwidth relatively increases the number of idle slots in the corresponding internal frame(s). This enhances the occurrence of matched idle slots (free ports in a space switch) among different internal frames and provides more rooms to connect a call via them. The level of output contention thus can be significantly reduced. Figure 2.4 shows one of the possible mappings of busy slots between a given external frame and the corresponding internal frame. It should be noted that common idle slots are most likely to be found between the given internal frame and other internal frames. This illustrates the potential performance of internal bandwidth expansion.

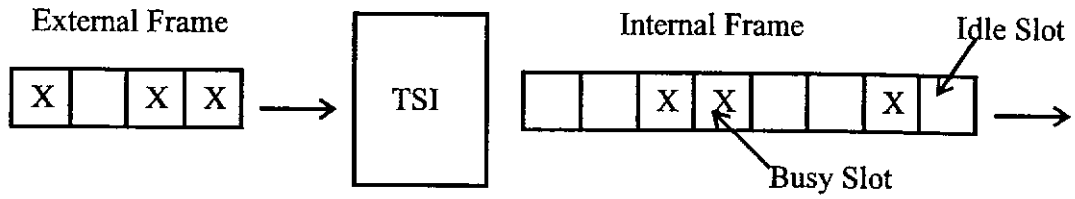


Figure 2.4 Example of transformation from the external frame to the internal frame.

Indeed, the two strategies, zone switching and internal bandwidth expansion, each has high potential to remove slot contention blocking significantly by providing more freedom for call scheduling (TSA). A multicast TMS having one of these properties

can greatly reduce the number of blocking even working under a simple random scheduling algorithm.

2.4 Chapter Summary

This chapter gave a brief introduction for a TMS and addressed the TSA problem associated with it. Several promising approaches in solving this problem were reported and discussed. Two approaches, zone switching and internal bandwidth expansion, hence were presented to argue how a multicast TMS can be designed to overcome scheduling conflicts and realize switch potential performance without using any complex control algorithm. The zone switching will be implemented by a general demux-mux architecture in Chapter 3 and a simple multistage architecture in Chapter 4. Then, a TST switch with intermediate dilation proposed in Chapter 5 would take the advantage of internal bandwidth expansion to achieve better performance.

Chapter 3 Demux-Mux Switch Architecture

In this chapter, a TMS with a demux-mux architecture is presented. This switch structure is widely employed in many existing optoelectronic communication systems to inter-connect a large number of high-speed data trunks. With trunk grouping, it provides more alternative routes for connecting a call and hence the level of call blocking can be reduced.

3.1 Characteristics of Demux-Mux Architecture

A demux-mux switch architecture is extensively used in various communication systems to inter-connect high-speed data trunks [GERS 96], [BUTN 96]. In this system, every input frame composed of F channels (slots) is first time-division demultiplexed into z sub-frames with f $(= F / z)$ channels each, then to a $Nz \times Nz$ non-blocking multicast switch which exchanges channels between different input sub-frames and output sub-frames by interconnecting these ports in a time slot. At the output, the traffic of z sub-frames will be time-division multiplexed into F channels in the connected output line. Once there is a new request arrival, the switch controller assigns matched free channels (idle slots) in the sub-frames of different addressed ports, in order to establish the requested call connection. The general structure of demux-mux switching system is shown in figure 3.1.

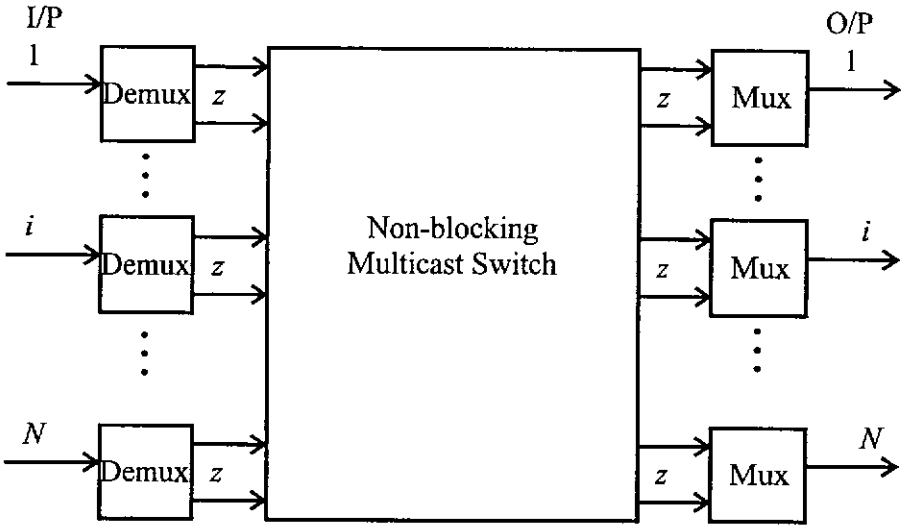


Figure 3.1 The structure of demux-mux switch.

In time-multiplexed switching, call scheduling faces different kinds of call blocking and is restricted by capacity constraint. The improvement on switch performance is only achieved via the reduction of slot contention blocking [KIM 92a]. The demux-mux switch reduces this kind of blocking by exploiting trunk grouping at both input side and output side [LI 91]. The concept of trunk grouping is to route each packet through a switching fabric on a trunk-by-trunk level instead of link-by-link level, where a trunk refers to the lines (sub-frames) from the same input (output) port. By doing so, the idle slot mismatching between the individual input sub-frame and output sub-frame, which is used to limit the system throughput and causes call blocking, is now grouped into the trunk consideration. Without trunk grouping, a call is blocked whenever an assigned input sub-frame has idle slots mismatched with the output ones. While with trunk grouping, it is blocked only when none of the sub-frames in the input group contains idle slot of position matched to that in the other requested output groups. Local contentions of individual sub-frames are therefore expected to be removed. In principle, trunk grouping increases the occurrence of matched slots

between different input sub-frames and output sub-frames which hence provides more alternatives to connect a call. Considering an example shown in figure 3.2, a call B connecting input #2 to output #1 and #2 faces slot contention blocking because of mismatched idle slots in the requested frames. Considering if the central space switch is now replaced by a demux-mux switch in which every input (output) frame is demultiplexed into two sub-frames (z=2) before switching, the call will be connected (see figure 3.3). This simple example illustrates the potential of the demux-mux switch to remove slot contention.

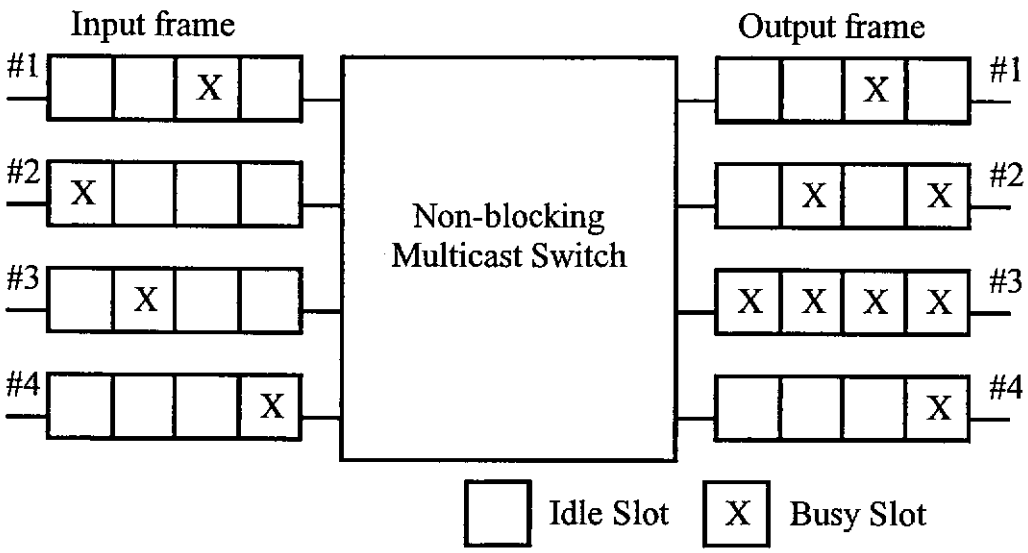


Figure 3.2 Examples of call blocking in a simplex switch.

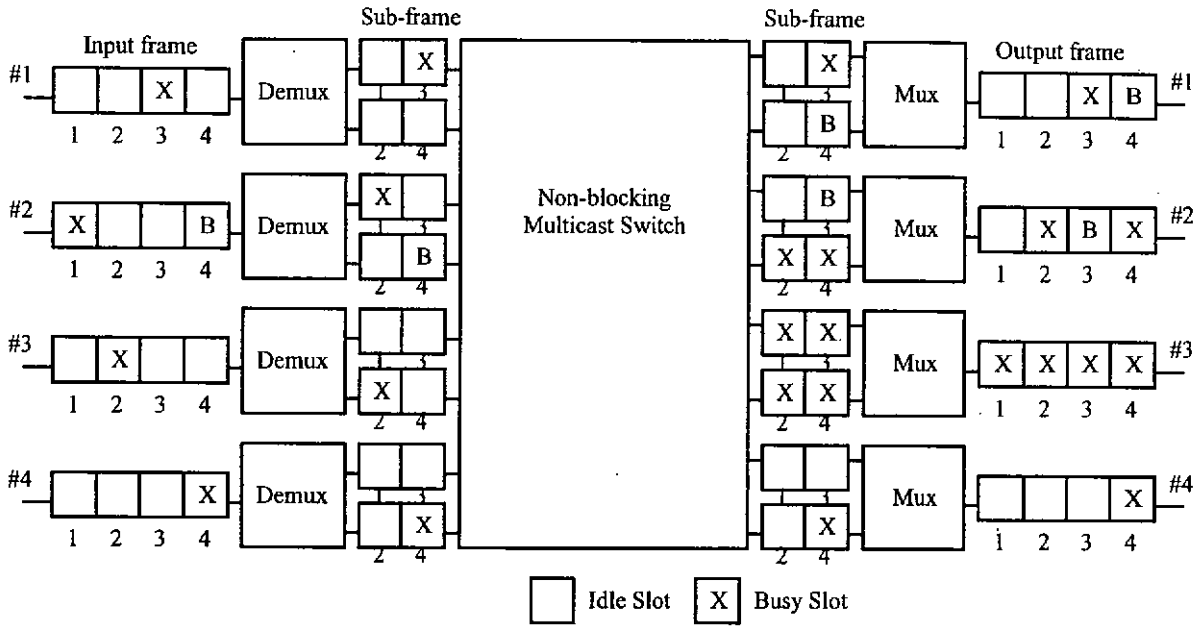


Figure 3.3 Call scheduling in demux-mux switch.

The demux-mux switch design is an alternative way to implement the zone switching described in Chapter 2. Every switching zone (defined in Chapter 2) created in this case contains z positions and their distributions are based on the design of the demultiplexer and multiplexer. For example, in figure 3.3, the switching zone of the first input slot (in the input frame) consists of the first and the second slot in each requested output frame. In order to connect a new call, it is necessary to have one or more idle slots in the switching zone of a particular input idle slot in every requested output frame. The greater the number of lines per trunk z the more the alternative routes for the switch to connect a call, and better performance is hence achieved.

3.2 Blocking Analysis

Call blocking probability of multicast traffic in a demux-mux switch with z lines per input (output) trunk is analyzed and presented in this section. For simplicity, the traffic is assumed to be homogeneous and uniformly distributed among all the input ports and output ports. Each call occupies one slot per TDM frame and has a fixed number (D) of destination, call fanout. The arrival rate of a call is assumed Poisson distributed with average arrival rate λ and its holding time is exponential with mean $1/\mu$. In this case, the switch is working under a random scheduling whereby a new call is connected by randomly allocating matched idle slots from the addressed sub-frame groups. A new request here will be granted if there exists common idle slots in different addressed sub-frames, otherwise, the request will be blocked. As the performance of this random scheduling is always poor than that of sophisticated scheduling algorithms, the corresponding analysis hence can provide a measure on the upper bound blocking probability.

Based on these assumptions, z sub-frames (sub-frame group) either from the same input demultiplexer or to the same output multiplexer can be modeled by a time-continuous Markov chain of z dimensions. The state of the chains is a vector $l=(l_1,l_2,\ldots l_z)$ which represents the distribution of busy slots in a given sub-frame group. The l_i is the total number of slot column with i busy slots each, where a slot column is constituted by the slots at the same position in the sub-frames of a given port. A sub-frame group of f slots per frame contains f slot columns with z slots each. Figure 3.4 shows the way to represent the distribution of busy slots in a sub-frame group by a distribution vector l. In the example, there are four busy slots in a given

sub-frame group. Slot columns at the second and the fourth position each contains one busy slot ($l_1 = 2$) and the one at the third position is saturated with busy slots ($l_2 = 1$). The corresponding distribution vector l is thus (2, 1).

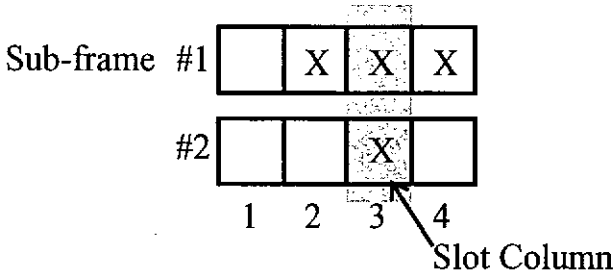


Figure 3.4 The way for a distribution vector l to represent the distribution of busy slots in a given sub-frame group.

Since a call is scheduled into slots coincided in time on every addressed input (output) frame, there is hence a certain extent of dependency between the distribution of busy slots in different input sub-frame groups and output sub-frame groups. In [KIM 92a], Kim and Lee studied the effect of switch size N and call fanout D on the interdependence of busy slots amongst the input frames and the output frames in a simplex switch. Their results showed that the effect of the dependency becomes negligible if the switch size (N) is considerably larger than the average call fanout (D) ($N / D > 8$). In order to ignore the similar interdependency amongst different sub-frame groups, we assume N is much greater than D in the analysis. The slot columns (with different number of busy slots) in each sub-frame groups are then assumed to be distributed by random and independent from each other.

Figure 3.5 shows the possible transitions into or from the state l in the input (output) Markov chain. There are at most $2 \cdot z$ states that have flows to the state l and

relatively at most $2 \cdot z$ transitions from it to the others. Assuming the given state has distribution vector $l = (l_1, l_2, \ldots l_z)$. The possible distribution vectors (states) connected to the state l can be denoted by l^{x+} and l^{x-} $(0 < x \leq z)$, where the l^{x+} is a distribution vector to which l would change in case of assigning a new call to a slot column with $x-1$ busy slots where $l_{x-1} > 0$ and $l_x < f$. On the other hand, the l^{x-} $(0 < x \leq z)$ is a distribution vector that l would change to upon a call departure from any slot column containing x busy slot given that $l_{x-1} < f$ and $l_x > 0$. The l^{x+} and l^{x-} $(0 < x \leq z)$ is hence given as

$$\begin{aligned} l^{x+} &= (l_1, \cdots l_{x-2}, l_{x-1} - 1, l_x + 1, l_{x+1}, \cdots l_z) \qquad \text{if } l_{x-1} > 0 \text{ and } l_x < f, \\ l^{x-} &= (l_1, \cdots l_{x-2}, l_{x-1} + 1, l_x - 1, l_{x+1}, \cdots l_z) \qquad \text{if } l_{x-1} < f \text{ and } l_x > 0, \end{aligned} \tag{3.1}$$

where l_0 is the number of slot columns each of which contains no busy slot.

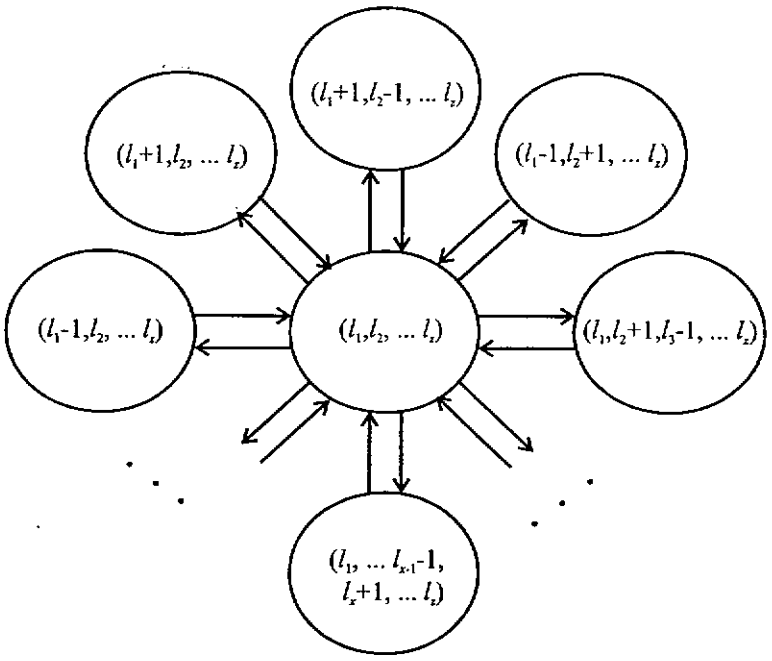


Figure 3.5 Possible transitions from state l to the other states in the input (output) Markov chain.

According to the previous definitions, the transition rate from state l to state l^{x-} is the call departure rate from slot columns with x busy slots each $\mu_x(l)$, which is directly proportional to the number of busy slots found in these slot columns and is thus given by

$$\begin{aligned}\mu_x(l) &= \left(\text{No. of busy slots in the slot columns with } x \text{ busy slots each}\right)\cdot \mu \\ &= l_x \cdot x \cdot \mu.\end{aligned} \tag{3.2}$$

In contrast, the transition rate from state l to state l^{x+} is the call arrival rate to a slot column of $(x-1)$ busy slots. Considering call blocking probability in addition to the possibility of connecting a new call via the slot columns other than the desired ones, an effective call arrival rate $\lambda_x(l)$ in this case is used instead of the average one λ to

calculate the call acceptance probability. The $\lambda_x(l)$ is the probability that an arrival call is accepted and connected via a slot column with x $(< z)$ busy slots in a given sub-frame group. It is equal to

$$\lambda_x(l) = \lambda \cdot \left(\begin{array}{l} \text{Prob. of an arrival call being accepted and connected via a slot} \\ \text{column with } x \text{ busy slots in a given sub - frame group} \end{array} \right).$$

(3.3)

In order to compute the effective call arrival rate $\lambda_x(l)$, we have to calculate the probability of having idle slots at the same position in the requested sub-frames (each comes from different addressed ports). The occurrence of these common idle slots is dependent on the distribution of slot columns saturated with busy slots in all the requested sub-frame groups. Supposing sub-frame groups in port A and B contain i and j saturated slot columns, respectively. The occurrence of the common idle slots can be indicated by finding unsaturated slot columns matched in position amongst these two groups, which in turn suggests that there exists at least one sub-frame in port A contains idle slots matched to that at port B in the positions of these common unsaturated slot columns.

In the analysis, we make use of a logical frame, AND frame as defined by Kim, to find these matched unsaturated slot columns [KIM 92a]. The AND frame of the sub-frame group in port A and B is generated by carrying a logic AND operation between their slot columns at the same position with considering saturated slot column as logic "0" and unsaturated slot column as logic "1". The idle slot (logic "1")

in it indicates the existence of common unsaturated slot columns. The generation of AND frame is shown in figure 3.6.

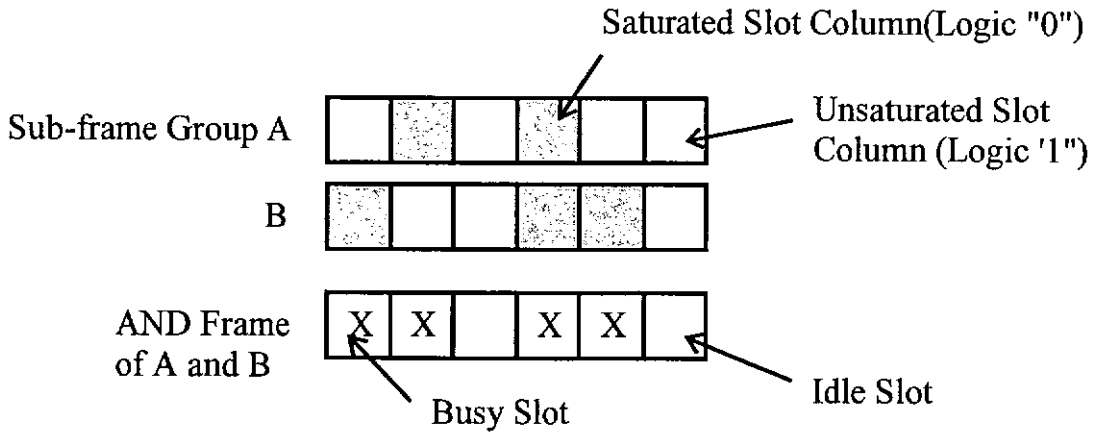


Figure 3.6 Generation of AND frame.

The occurrence of idle slots in the AND frame is depended on the distribution of saturated slot columns in each requested input (output) sub-frame group. The probability distribution of having n saturated slot columns in the input sub-frame group $P(n)$ and in the output sub-frame group $Q(n)$ are given by

$$P(n) = \sum_{l \in L_n} p(l), \tag{3.4}$$

and

$$Q(n) = \sum_{l \in L_n} q(l), \tag{3.5}$$

where $p(l)$ and $q(l)$ are the steady-state probabilities of a distribution vector l in the input Markov chain and output Markov chain, respectively and L_n is a set of the distribution vectors $l = (l_1, l_2, \ldots l_z)$ with n saturated slot columns, i.e., $l_z = n$.

$$L_n = (l:\ l_z = n) \tag{3.6}$$

Giving two sub-frame groups with i and j saturated slot columns randomly distributed, the probability of finding k busy slots in the AND frame generated from them $\pi(k \setminus i, j)$ is

$$\pi(k \setminus i, j) = \begin{cases} \dfrac{\binom{i}{k-j} \cdot \binom{f-i}{k-i}}{\binom{f}{j}} & \text{if } \max(i, j) \le k \le \min(f, i+j), \\ 0 & \text{otherwise.} \end{cases} \tag{3.7}$$

The probability distribution of the common unsaturated slot columns in all the requested output sub-frame groups is hence found by a recursive probability function $Q_n(k)$ that there are exactly k $(0 \le k \le f)$ busy slots in the AND frame generated from n output sub-frame groups. The $Q_n(k)$ is derived and is given by the following equation with the initial condition $Q_1(k) = Q(k)$

$$Q_n(k) = \sum_{i=0}^{k} \sum_{j=k-i}^{k} \pi(k \setminus i, j) \cdot Q_{n-1}(i) \cdot Q(j) \quad \text{for } n \ge 2. \tag{3.8}$$

Supposing that there is a connection request from the input port containing n saturated slot columns and m desired slot columns (with x busy slots each) to the output ports from which the AND frame generated contains j busy slots. The call arrival rate to a

slot column with x busy slots is calculated by finding the probability $\theta(u,k \setminus n,m,j)$ that the given input sub-frame group has $(f-k)$ unsaturated slot columns in the positions of the idle slots of the output AND frame where u of them are the desired slot columns. Figure 3.7 shows the way to derive $\theta(u,k \setminus n,m,j)$. Based on the random distribution of different slot columns in the input and output ports, it will suffice to consider the distribution of input slot columns fixed and the output arrangement random. Thus, we let the first n slot columns in the input sub-frame group being saturated and the next m ones with x busy slots each, and let the j busy slots in the given output AND frame be randomly distributed (as seen in the figure). The $\theta(u,k \setminus n,m,j)$ is thus derived by counting the number of possible ways to arrange these j busy slots in the AND frame, in order to have such kind of matching of the desired slot columns. It is given by the following equation.

$$\theta(u,k \setminus n,m,j) = \frac{\begin{pmatrix}\text{No. of ways in distributing}\\ (n-(k-j)) \text{ busy slots in}\\ n \text{ positions}\end{pmatrix} \cdot \begin{pmatrix}\text{No. of ways for}\\ \text{distributing } (m-u)\\ \text{busy slots in } m \text{ positions}\end{pmatrix} \cdot \begin{pmatrix}\text{No. of ways to distribute}\\ (k-n-(m-u)) \text{ busy slots}\\ \text{in } (f-n-m) \text{ positions}\end{pmatrix}}{\begin{pmatrix}\text{No. of ways for distributing}\\ j \text{ busy slots in an AND frame}\\ \text{of } f \text{ slots}\end{pmatrix}}$$

$$= \frac{\binom{n}{n-(k-j)} \cdot \binom{m}{m-u} \cdot \binom{f-n-m}{k-n-(m-u)}}{\binom{f}{j}}$$

$$= \frac{\binom{n}{k-j} \cdot \binom{m}{u} \cdot \binom{f-n-m}{f-u-k}}{\binom{f}{j}}$$

Therefore,

$$\theta(u,k \setminus n,m,j) = \begin{cases} \dfrac{\binom{n}{k-j}\cdot\binom{m}{u}\cdot\binom{f-n-m}{f-u-k}}{\binom{f}{j}} & \text{if } \max(n,j) \le k \le \min(n+j,f) \\ & \text{and } \max(0,m-(k-n)) \le u \le \min(m,f-k) \\ 0 & \text{otherwise.} \end{cases} \quad (3.9)$$

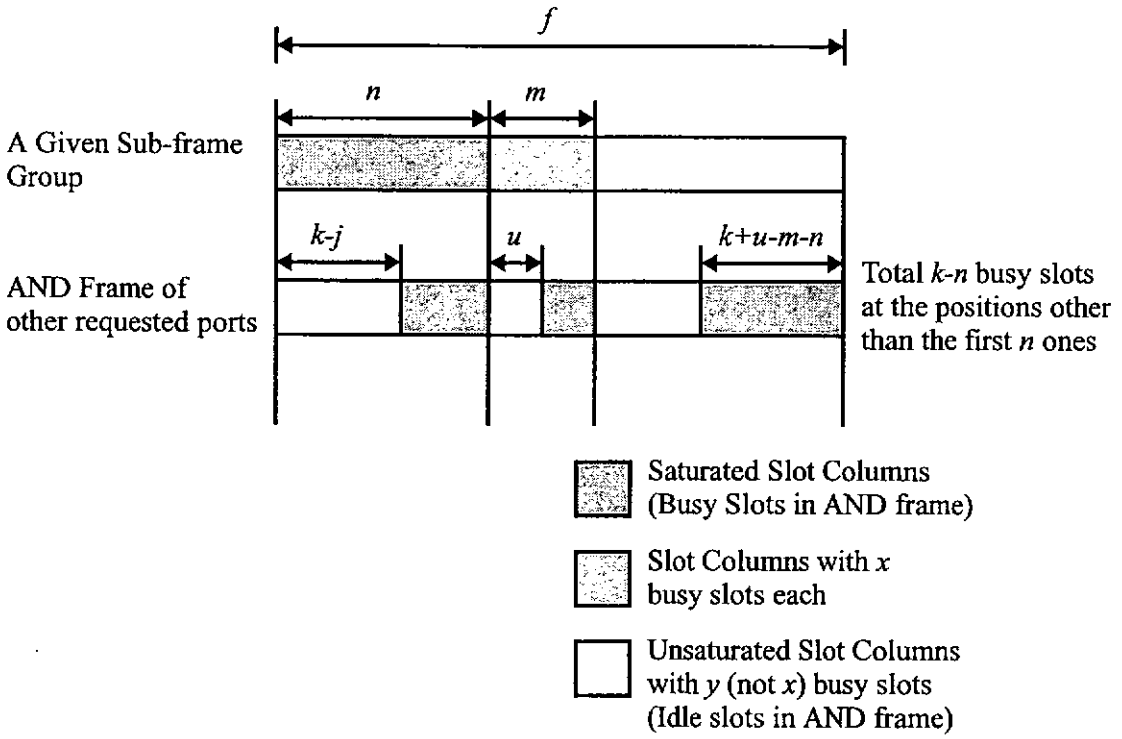


Figure 3.7 Derivation of probability function $\theta(u,k \setminus n,m,j)$.

Giving that the input sub-frame group has a distribution vector $l = (l_1, l_2, \ldots l_z)$, an arrival call is accepted and assigned to the slot column of x busy slots if there is at least one idle slot in the AND frame generated from the corresponding input sub-frame group and all the (D) requested output sub-frame groups in the position of

desired slot columns and the switch finally selects it to connect the call. Therefore, the effective arrival rate to an input slot column with x busy slots $\lambda_{I,x}(l)$ $(0 \le x < z)$ is

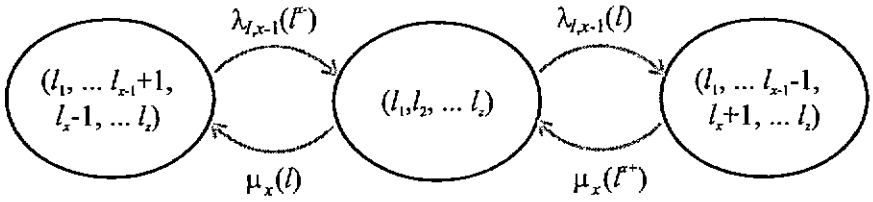
$$\lambda_{I,x}(l) = \lambda \cdot \left(\sum_{k=0}^{f-1} \sum_{j=0}^{f-1} \sum_{u=1}^{l_x} \left(\begin{array}{l} \text{Prob. of choosing slot} \\ \text{column of } x \text{ busy slots} \\ \text{to connect a call} \end{array} \right) \cdot \theta(k, u \setminus l_z, l_x, j) \cdot Q_D(j) \right) \text{ for } l \notin L_f \text{ and } l_x > 0$$

$$= \lambda \cdot \left(\sum_{k=0}^{f-1} \sum_{j=0}^{f-1} \sum_{u=1}^{l_x} \frac{u}{f-k} \cdot \theta(k, u \setminus l_z, l_x, j) \cdot Q_D(j) \right). \tag{3.10}$$

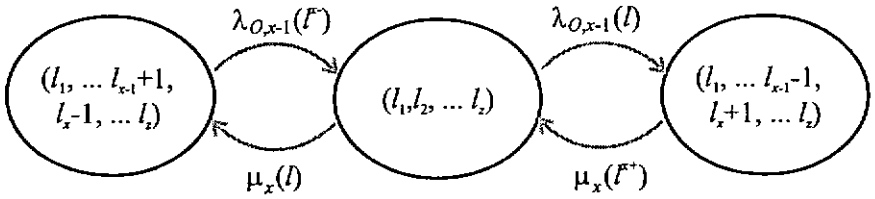
For the output sub-frame group with the same distribution vector l, the corresponding effective arrival rate $\lambda_{O,x}(l)$ $(0 \le x < z)$ is the probability that the call destined for it finds one or more idle slots in the AND frame generated by the corresponding output sub-frame group and the other $(D-1)$ requested output sub-frame groups together with the input one in the positions of the desired slot columns and is connected via it. The average arrival traffic here is D times as much as the input one because in this case, each call connects one input port to D output ports. The $\lambda_{O,x}(l)$ is thus given by

$$\lambda_{O,x}(l) = D \cdot \lambda \cdot \left(\sum_{k_1=0}^{f-1} \sum_{k_0=0}^{f-1} \sum_{u=1}^{l_x} \frac{u}{f-k_1} \cdot \theta(k_1, u \setminus l_z, l_x, k_0) \cdot \left(\sum_{i=0}^{f-1} \sum_{j=0}^{f-1} \pi(k_0 \setminus i, j) \cdot P(i) \cdot Q_{D-1}(j) \right) \right)$$
$$\text{for } l \notin L_f \text{ and } l_x > 0. \tag{3.11}$$

With the effective call arrival rates, the steady-state probabilities in the input (output) Markov chain can be computed via the equation that governs the flow of the corresponding state probabilities. The general flow of each state in the input and output Markov chain is shown in figure 3.8.



The Input Markov Chain



The Output Markov Chain

Figure 3.8 General flows amongst states l, l^{x+} and l^{x-} in the input (output) Markov chain.

By inspection, the flow equation of the input and output Markov chains may be obtained as,

$$\dot{p}(l) = \sum_{x=1}^{z}\left(\xi(l^{x+})\cdot\left(p(l)\cdot\lambda_{I,x-1}(l) - p(l^{x+})\cdot\mu_x(l^{x+})\right) + \xi(l^{x-})\cdot\left(p(l)\cdot\mu_x(l) - p(l^{x-})\cdot\lambda_{I,x-1}(l^{x-})\right)\right), \tag{3.12}$$

and

$$\dot{q}(l) = \sum_{x=1}^{z}\left(\xi(l^{x+})\cdot\left(q(l)\cdot\lambda_{O,x-1}(l) - q(l^{x+})\cdot\mu_x(l^{x+})\right) + \xi(l^{x-})\cdot\left(q(l)\cdot\mu_x(l) - q(l^{x-})\cdot\lambda_{O,x-1}(l^{x-})\right)\right), \tag{3.13}$$

for all possible l's, where $\xi(l')$ is

$$\xi(l') = \begin{cases} 1 & \text{if } \sum_{i=1}^{z} l'_i \le f \\ 0 & \text{otherwise} \end{cases} \tag{3.14}$$

in which $l' = (l'_1, l'_2, \ldots l'_z)$.

As the Markov chains are ergodic, for a given set of non-zero $\lambda_{I,x}(l)$ and $\lambda_{O,x}(l)$ $(0 \le x < z)$, $\dot{p}(l)$ and $\dot{q}(l)$ will approach zero and steady-state $p(l)$ and $q(l)$ exist. However, the calculation of $\lambda_{I,x}(l)$ is depended upon the unknown steady-state probability $p(l)$ and $q(l)$, which greatly complicates the calculation of steady-state distributions using the above equations. In order to solve this problem, we make use of a numerical iterative method in [ROSE 87] whereby in each iteration, the sets of $p(l)$ and $q(l)$ will be calculated via the equation (3.12) and (3.13) with presenting $\lambda_{I,x}(l)$ and $\lambda_{O,x}(l)$ in terms of $p'(l)$ and $q'(l)$, which are the previous outputs of the corresponding steady-state computation. This procedure will be repeated again and again until the following inequality is satisfied at the end of the iteration.

$$\sum_{l \in L} \left(\left(p(l) - p'(l) \right)^2 + \left(q(l) - q'(l) \right)^2 \right) < 10^{17} \tag{3.15}$$

where L is a set of possible l's with $\xi(l) = 1$.

Then, the sets of $p(l)$ and $q(l)$ output in the final iteration will be considered as steady-state probabilities in the input and output Markov chains.

In the numerical iterative method, we are used to find a set of initial values for different $\lambda_{I,x}(l)$ and $\lambda_{O,x}(l)$, in order to generate the state probability $p(l)$ and $q(l)$ in the first iteration. The determination of these initial values is quite arbitrary and can be considered as an art. For the sake of simplicity and similarity, the initial values of $\lambda_{I,x}(l)$ and $\lambda_{O,x}(l)$ are determined by substituting $P(n)$ and $Q(n)$ with the probabilities of having n busy servers in the well-known f-server loss systems of the same average arrival rate and holding time as the input frame and the output frame, respectively. These probabilities can be computed directly from some general equations [KLEI 75].

Finally, the call blocking probability P_B is the probability that there is no idle slot in the AND frame generated from the input sub-frame group and D requested output sub-frame groups and is given as

$$P_B = \sum_{i=0}^{f} \sum_{j=f-i}^{f} \pi(f \setminus i, j) \cdot P(i) \cdot Q_D(j) . \qquad (3.16)$$

The above equations thus derived is used to measure the performance of the demux-mux switching system.

3.3 Performance Evaluation

In this section, the performance of multicast traffic in a demux-mux switch is evaluated by varying the parameters including the number of lines per trunk z, frame size F and call fanout D.

In figure 3.9, the call blocking probability of multicast traffic in demux-mux switch is plotted for various values of z and D, fixing frame size F at 24. For a given call fanout D, the call blocking probability decreases as z increases. It is because as z increases, the occurrence of common idle slot between the addressed sub-frame groups increases too, which provides more alternatives to connect a call and hence improves switch performance. In the special case where $z = 24\,(F)$, the call blocking probability is exactly equal to the lower bound blocking probability derived by Kim [KIM 92a] in Appendix A. It is due to the fact that each sub-frame in this case composes of only one time slot. So the common unsaturated slot columns are always found in the requested sub-frame groups unless one or more of the lines overloaded. The switch therefore can completely eliminate slot contention and the call blocking encountered in this case is only due to the line overloaded which gives a measure on the lower bound blocking probability.

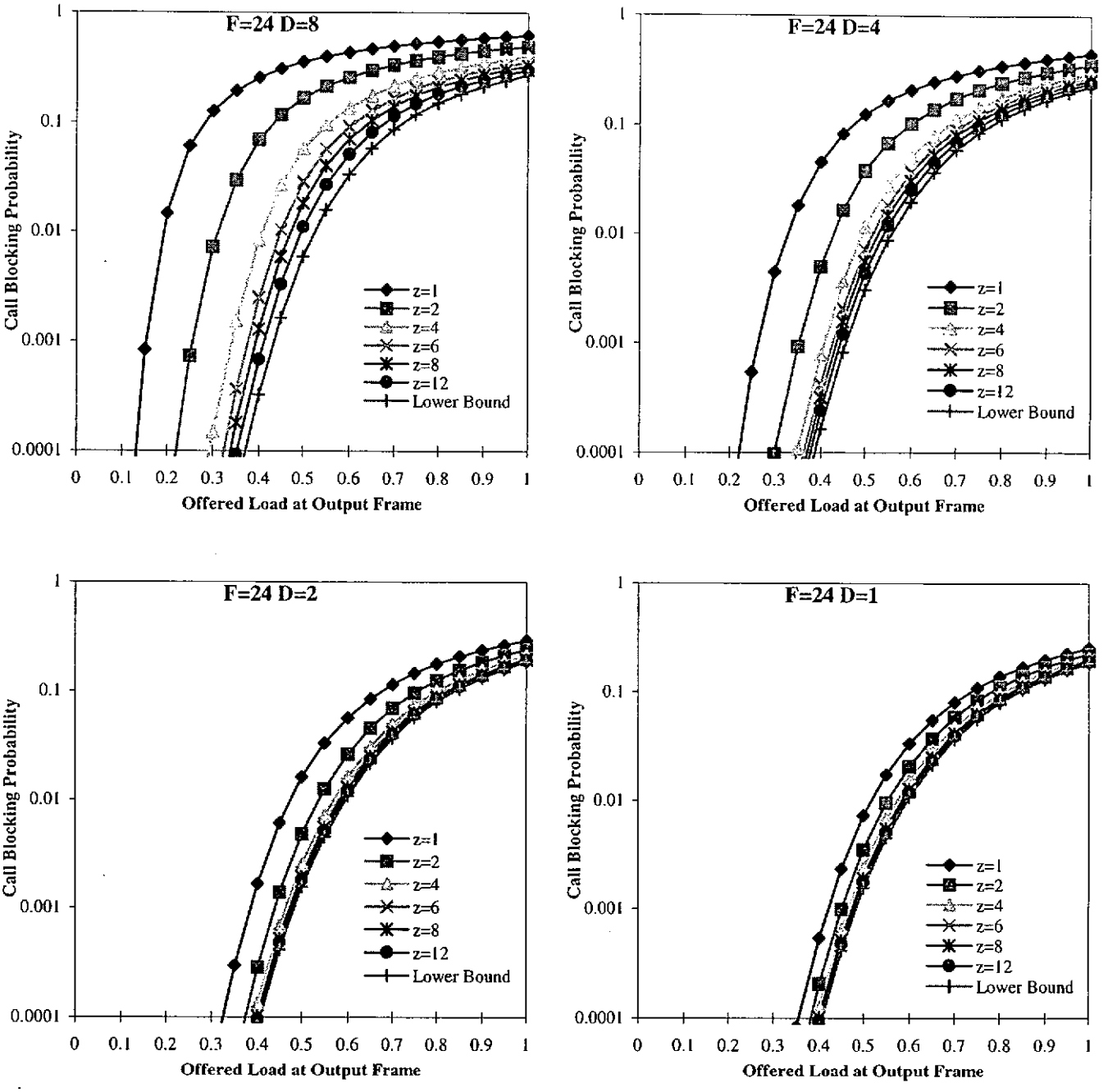


Figure 3.9 The call blocking probability of multicast traffic in a demux-mux switch with different number of lines per trunk z.

For point-to-point communication $D = 1$, as shown in [BRAT 84] and [ROSE 87], the performance gap between the lower bound and a simplex switch $(z = 1)$ is small. This implies that the potential performance improvement attained by the demux-mux switch is not very significant. There is less incentive for us to use the demux-mux switch with a large number of ports per address to get a slight improvement. However, when D is fairly large, this performance gap increases rapidly. Our results show that the increase in z improves the switch performance substantially and bridges the gap. For clarification, the performance improvement achieved by the demux-mux switch with various values of z is presented in terms of the maximum offered load achieved at the output time frame at fixed call blocking probability $B = 0.01$. The plots are shown in figure 3.10 for different call fanout D.

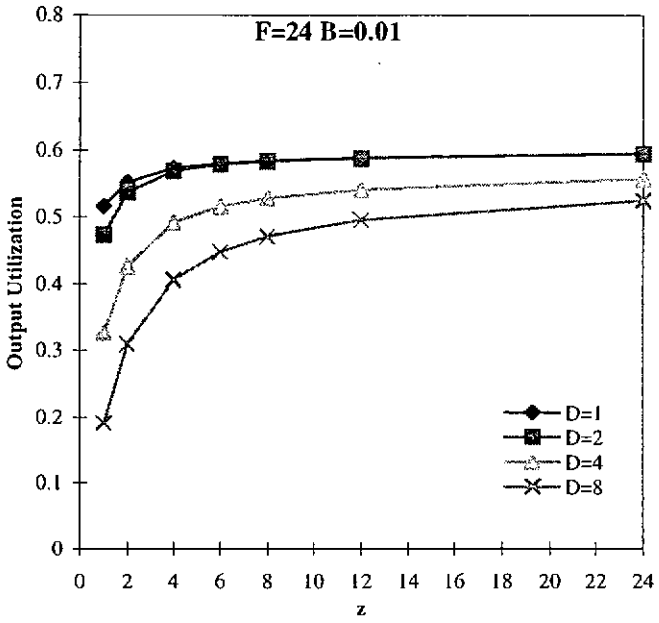


Figure 3.10 Maximum offered load at output frame with different call fanout D and z at fixed call blocking probability $B = 0.01$.

Similarly, the effect of frame size F is investigated by plotting the output achievable load as a function of F for the same call blocking probability and various values of z in figure 3.11. It is readily seen that a switch with a larger frame achieves better performance for a given z. This performance improvement is due to the fact that for larger F, there are more opportunities for the occurrence of common idle slots between a given input and output sub-frame each with $f = F / z$ slots. In the plot, it is also shown that the larger the F the greater the z required by the demux-mux switch to have the performance approaching the lower bound. It is because for a given z, the size of the corresponding sub-frames f increases with F, which in turn increases the possibility of slot contention.

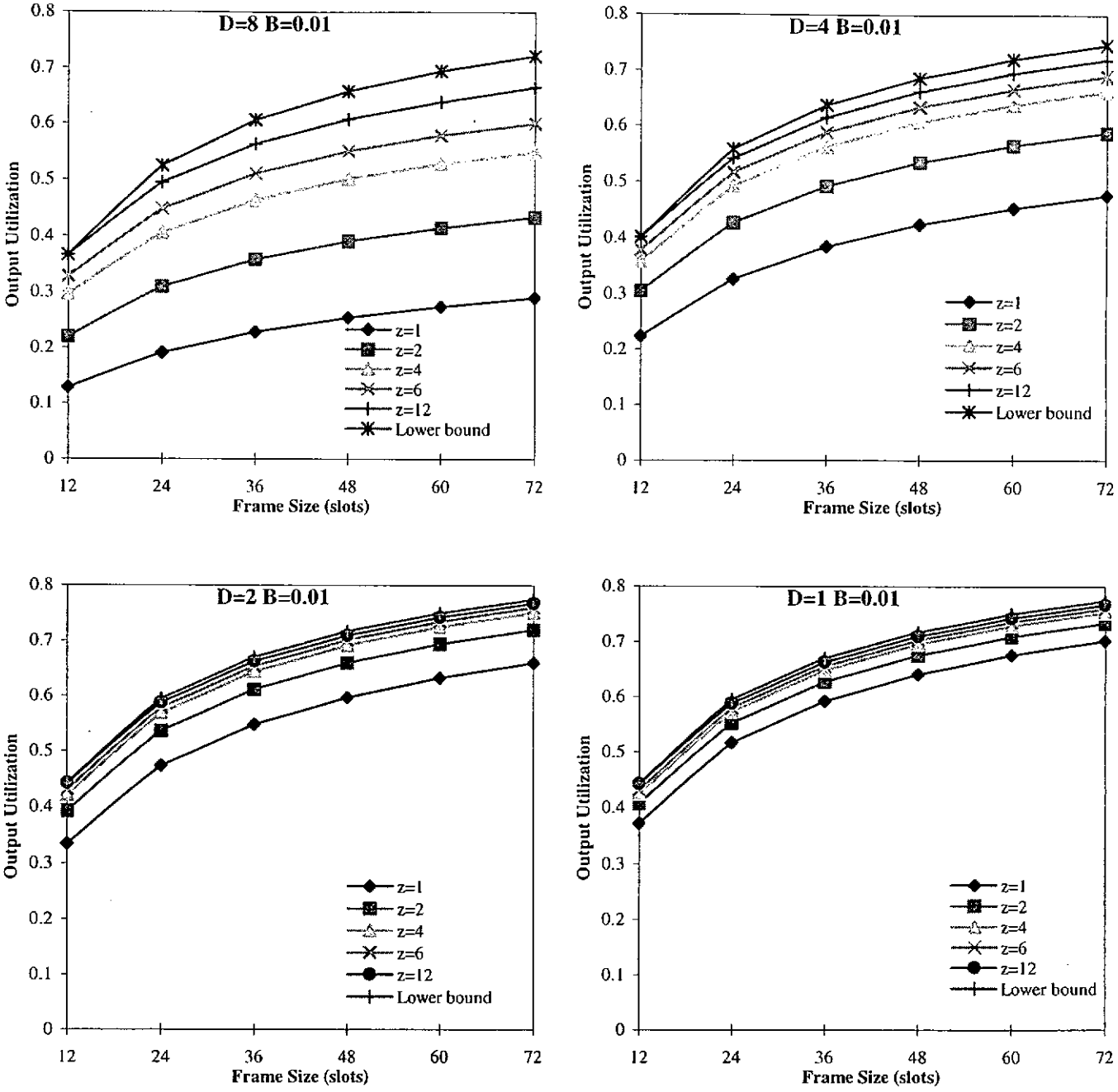


Figure 3.11 Maximum offered load at output frame with different frame size F and z at fixed call blocking probability $B = 0.01$.

It is also of great practical value to investigate how many lines per trunk are required for a demux-mux switch to attain near-optimal performance, which here is obtained by comparing the performance of a demux-mux switch with different values of z to its lower bound. For the sake of clarification, we make use of a parameter, percent

deviation, to measure the extent of these performance differences. The percent deviation is the percentage performance improvement achieved by the optimal scheduling over a demux-mux switch. In this case, the performance is in terms of achievable output utilization (load) at fixed call blocking probability. The calculation of percent deviation is given as

$$\text{Percent Deviation} = \left(\frac{U_{Optimal}}{U_{Demux\text{-}Mux,z}} - 1 \right) \times 100\% . \tag{3.17}$$

Note that $U_{Optimal}$ and $U_{Demux\text{-}Mux,z}$ are the maximum output utilization attained by the optimal scheduling and the demux-mux switch of z lines per trunk at fixed call blocking probability B, respectively.

The percent deviation of demux-mux switch is plotted as a function of F in figure 3.12 with fixing call blocking probability $B = 0.01$. In the plots, we observe that for any $z > 2$, there is only a slight change in percent deviation from optimal scheduling, especially when F is large in value, say $F > 36$. These results show that when F is sufficient large, the increment in it does not further enlarge the performance gap between the optimal scheduling and a demux-mux switch with $z > 2$. Therefore, in order to attain a desired percent deviation, the number of lines required per trunk in a demux-mux switch can be determined by considering the average call fanout only, no matter how fast the data rate in each external link (frame size) is. This gives us a hint on the cost-effective design of demux-mux switch in different communication systems.

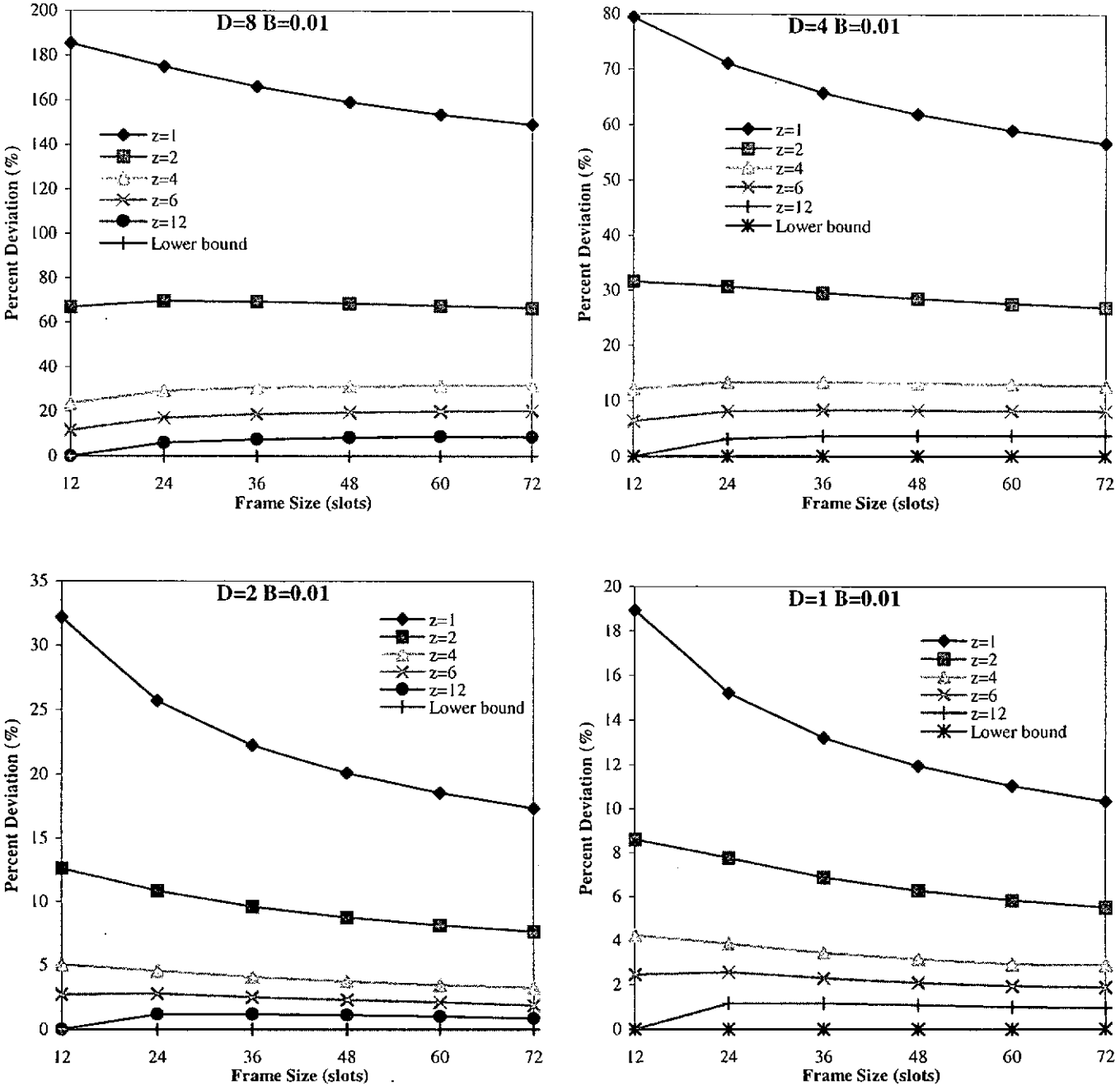


Figure 3.12 Percent Deviation with different frame size F and z at fixed call blocking probability $B = 0.01$.

To test the validity of the assumptions made in section 3.2, the performance of a demux-mux switch structure was simulated for $N = 256$ with various values of z and F. For random scheduling, call blocking probability as a function of offered load was

calculated. The results are compared to that found by the analysis for $F = 24$ and 48 in figure 3.13. It is shown that the simulation results always match the analytic ones for both $D = 4$ and $D = 8$. This indicates that our assumptions are reasonable.

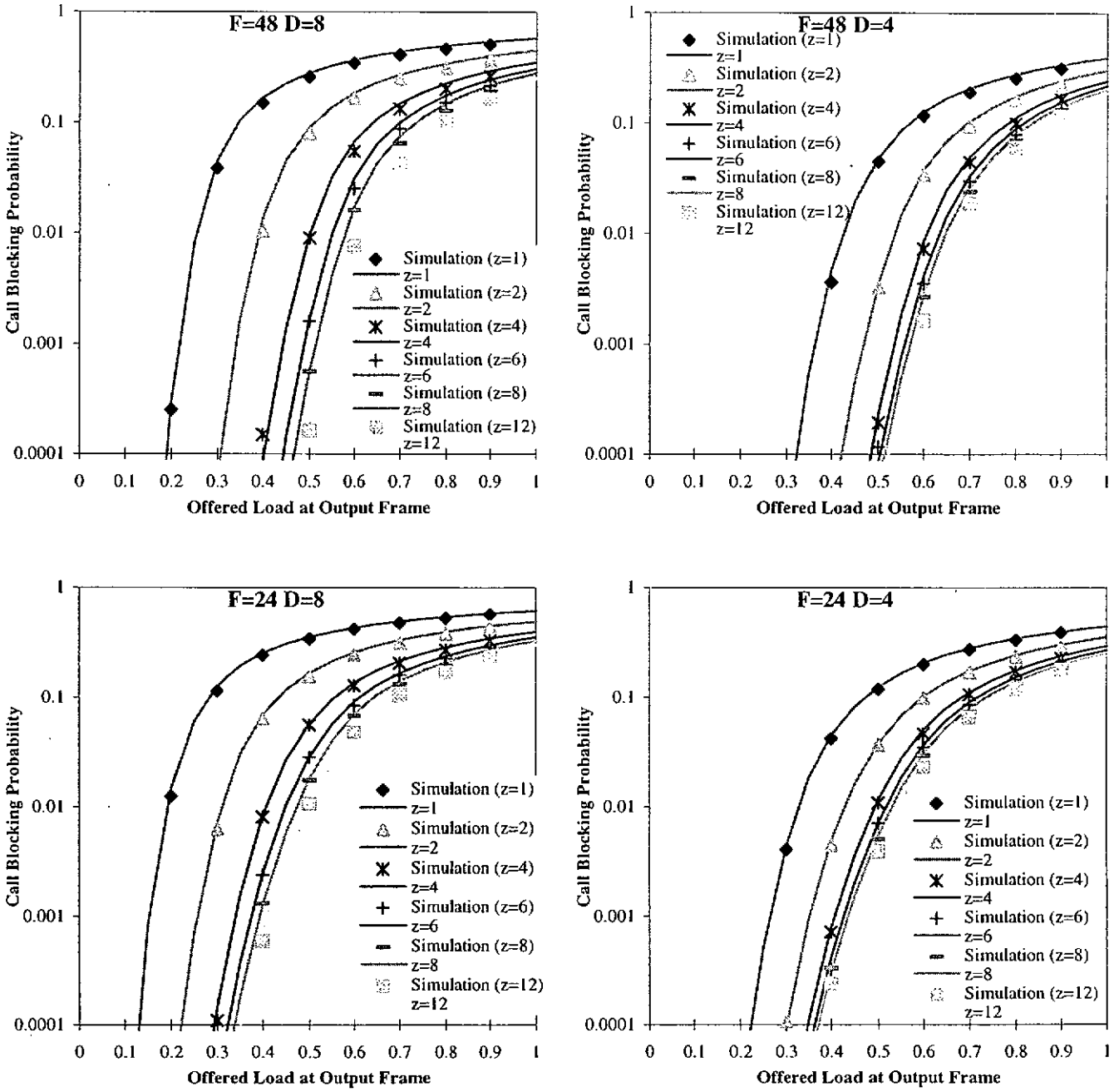


Figure 3.13 Comparison of the call blocking probability between simulation results and analytical results.

Note that the further comparison between these results for smaller switch size is given in Appendix B.

3.4 Chapter Summary

In this chapter, we proposed a demux-mux architecture for multicast TMS to reduce the level of call blocking. The performance of homogeneous multicast traffic in it was studied and analyzed via arrival rate modulation technique. Our results show that the larger the number of lines per trunk the lesser the call blocking encountered in the system throughout the range of interesting conditions. This performance improvement is resulted by increasing the rooms for us to connect a call between different addressed ports, which greatly reduces the occurrence of slot contention blocking in the switch and hence improves switch performance. In this system, the performance gap between a demux-mux switch and its lower bound can be bridged by increasing the number of lines per port only, without requiring any sophisticated scheduling scheme. Note that although a switching fabric in this case has been enlarged from $N \times N$ to $Nz \times Nz$, the speed at which each input to the fabric operates can be reduced by a factor z. The reduction of the speed of the switching fabric relative to the incoming speed increases the feasibility of the switch implementation. The trade off may be the requirement of the additional hardware at both input and output ports.

The demux-mux architecture is always used as cross-connect nodes in broadband communication networks. It is also a key building block for the existing electro-optical node and the oncoming full-conversion all-optical node. According to

present technology, the optical transmission rate is likely at 2.5G bps while the electronic switching rate is in the range of 150M bps, a typical ratio of optical transmission rate to electronic switching rate is then very large ($z \geq 16$). The performance improvement noted in this study shows that switch nodes with these values of z can eliminate most of the call blocking inherently without the need of any costly design to handle multicast traffic.

Chapter 4 A Multistage Switch using Window Scheduling

This chapter introduces a novel call scheduling scheme, window scheduling, which is the simplest and a direct way to implement the concept of zone switching proposed in Chapter 2. To realize this scheduling scheme in an all-optical TDM system, a simple multistage switch is developed. The switch consists of w switching planes which are connected in series by a set of fiber delay lines. While working with window scheduling, the switch has potential to eliminate most of output conflicts. Compared to the demux-mux switch in Chapter 3, the multistage switch has lower hardware requirement in achieving the same extent of performance improvement.

4.1 Principle of Window Scheduling

In a time-division multiplexing (TDM) switching system, the incoming traffic must be scheduled to avoid two or more packet streams converging simultaneously upon a single output, which would result in the loss of data packets. This is accomplished by carrying call scheduling on each new request. Figure 4.1 shows an example of call scheduling in which an arrival call is connected by finding idle slots in each requested input (output) frame at the same position such that the switch can connect the call at this time slot throughout its duration. The lack of these common idle slots would result in call blocking.

In this chapter, a novel call scheduling concept, window scheduling, is proposed to reduce call blocking. In the window scheduling, to establish a call connection, the switch is no longer necessary to allocate the idle slots of output frames in the position of input idle slot but allowing certain degree of backward displacement or delay which are measured in terms of time slots and are accomplished by means of delay lines. The maximum slot displacement allowed is $(w-1)$ where w is the window size. By exploiting the delay in packet arrival at the switch output, a call will be connected if there is at least one idle slot found in the first w slots in each requested output frame counting inclusive from the position of input idle slot. These w output slots constitute the window at the position of corresponding input idle slot. In other words, a window at slot $\#i$ $(0 \le i < F)$ consists of slots numbered from i to $(i+w-1) \bmod F$ in the same frame. As frames are transmitted successively, the windows at the last $(w-1)$ slots associate with the first few slots of the following frame. A window opens when there is at least one idle slot found in its associated slots; otherwise, it closes. To connect a call, it requires to match the open windows in every requested output frame to the input idle slot at the same position. Figure 4.2 shows how the window scheduling removes output conflicts. It illustrates the working principle of window scheduling. In the example, there is no matched idle slot in the input frame and output frames. The call will be blocked under traditional call scheduling. However, using window scheduling and letting windows of size $w=2$ (indicated in the figure by shaped slots) open in both output frames in the position of the input idle slot, the call will be connected.

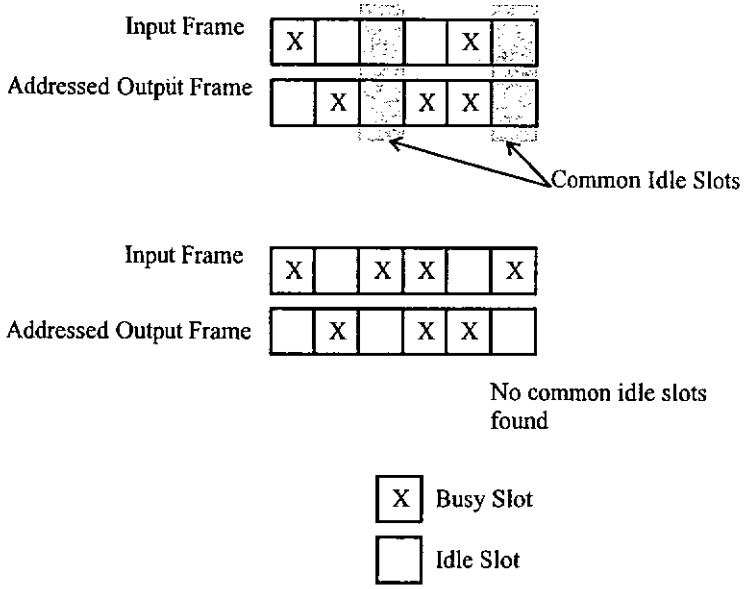


Figure 4.1 (a) The arrival call can be connected via any one of the common idle slot groups of its requested input and output time-multiplexed frames. (b) Call blocking because of the lack of common idle slots.

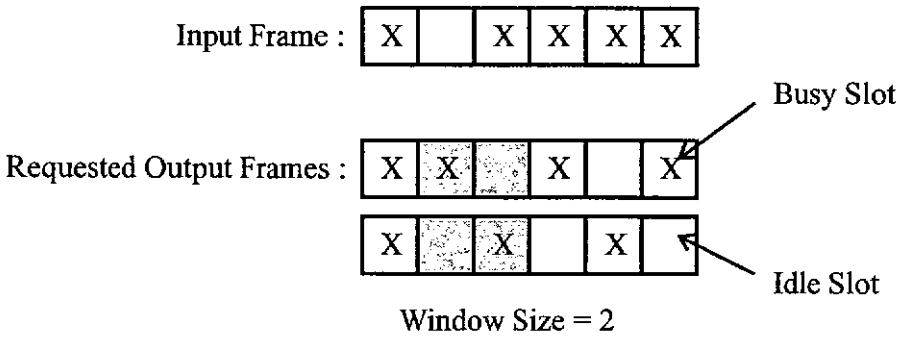


Figure 4.2 Example of window scheduling.

Window scheduling is the simplest and an efficient way to implement zone switching to reduce slot contention. The window in this case is exactly the switching zone defined in Chapter 2. Its implementation only requires some delay elements and no buffering is needed. Therefore, it is very suitable for using in a single hop all-optical network (AON) [GIPS 96].

4.2 Multistage Switch Design

In this section, a multistage architecture is proposed to implement window scheduling in an all-optical switching system. The designed switch has N input ports and N output ports. It consists of w $N \times N$ cross-bar switching planes which are connected in series by several fiber delay lines of one slot time delay each. The N outputs of each switching plane are connected to N concentrators, respectively. The concentrators collect data packets from any one of the connected lines and send them out each time slot. There is no buffering of transit packets during switching. Figure 4.3 shows the general structure of the multistage switching system.

In the switch, each incoming data packet is transmitted over successive switching planes via the fiber delay lines between them. Once the packet arrives the assigned switching plane, it is being copied to the destination output port by the corresponding switching element whose operation is according to the TSA for this call. For example, the data packet scheduled to be output at $(i-1)$ time slots later will be switched and transmitted at switching plane #$(i-1)$.

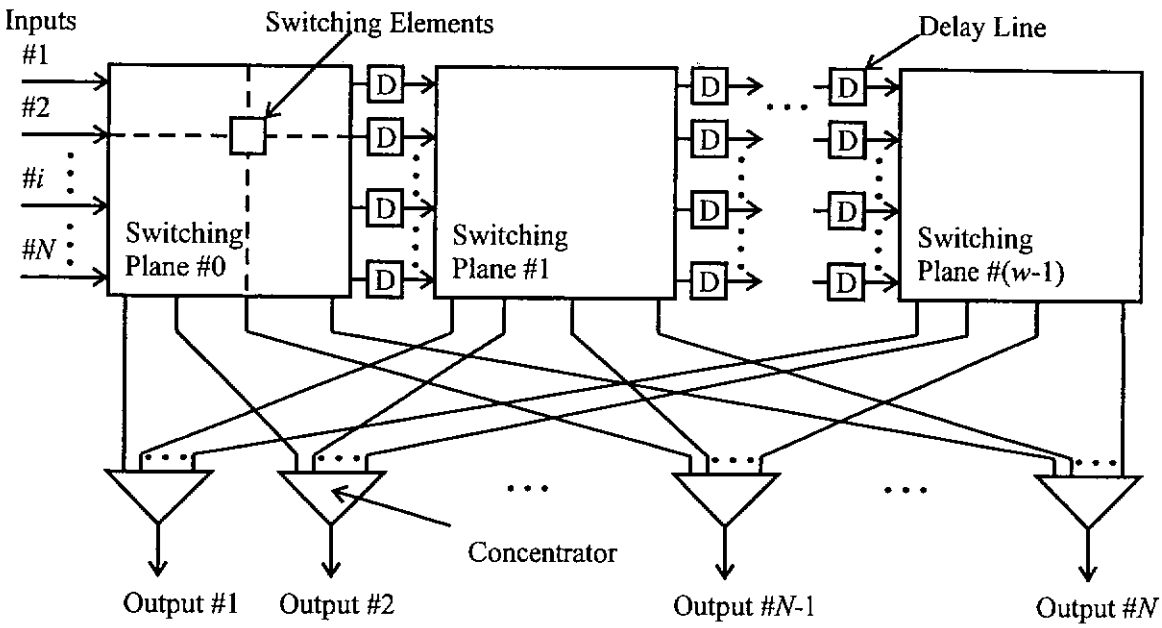
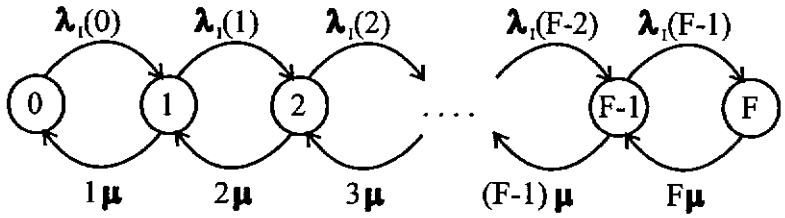


Figure 4.3 General structure of a multistage switch.

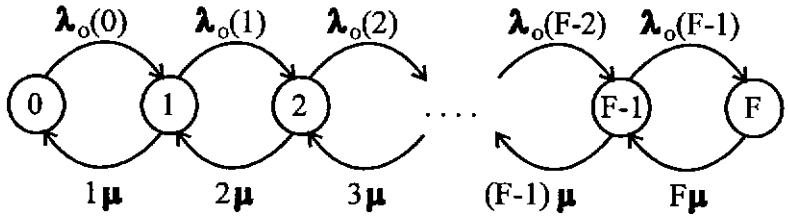
4.3 Blocking Analysis

To evaluate the switch performance, call blocking probability of multicast traffic in the multistage switch is analyzed via the arrival modulation technique used in [LUND 48], [SYSK 60]. In this case, the traffic is assumed homogeneous and uniformly distributed among all the input ports and output ports. Each call here occupies one slot per input (output) frame and has fixed number (D) of destinations, which is assumed to be very small compared to the total number of ports N in the switch to ignore the effect of interdependence amongst the input frames and the output frames [KIM 92a]. The arrival rate of a call is assumed Poisson distributed with average arrival rate of λ and its holding time is exponentially distributed with mean of $1/\mu$. Each input (output) frame has F slots and supports at most F calls concurrently.

In the analysis, the input frame and the output frame are modeled by two independent continuous-time Markov chains whose state is determined by the number of busy slot k found in each frame. As a call faces both overflow blocking and slot contention blocking, an effective arrival rate $\lambda(k)$ is employed instead of the average arrival rate λ as the transition probability in the Markov chains, where $\lambda(k)$ is the average number of call arrivals excluding the blocked call at each input (output) time frame with k busy slots. In this case, $\lambda_I(k)$ and $\lambda_O(k)$ are denoted as the effective call arrival rate at input and output frame with k busy slots, respectively. Both of them are based on the steady state probabilities of Markov chains obtaining through a numerical iterative method. The continuous-time Markov chains for both input and output time frames are shown in Figure 4.4.



Input Markov Chain



Output Markov Chain

Figure 4.4 The continuous-time Markov chain for input and output time frames.

To connect a call, it is necessary to find one or more matched idle slots and open windows in the input time frame and every output window frame. The probability of finding common idle slot and open windows are obtained by making use of a logic AND frame which has been widely used in [KIM 92a], [KIM 92b] and in Chapter 3. The AND frame is a logic frame generated by carrying a logic AND operation between two or more frames at each slot position according to the state of the corresponding window in the output frames as well as the state of the corresponding slot in the input frame, where close window and busy slot are considered as logic '0'. Similarly, open window and idle slot are considered as logic '1'. An idle slot in the AND frame indicates the existence of common idle slot and open windows. Figure 4.5 shows the generation of the AND frame.

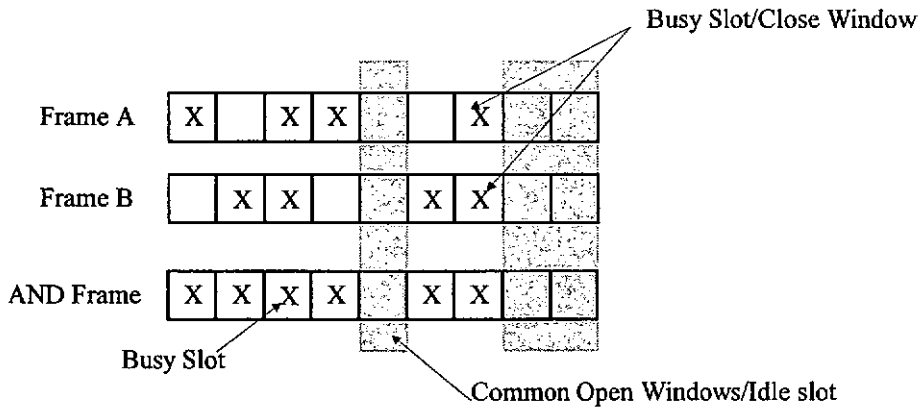


Figure 4.5 The generation of the AND frame with respect to the output close windows and the input busy slots.

Let's suppose output frame A and B contain l close windows and m busy slots, respectively. The probability distribution of having k busy slots in the AND frame generated from A and B is calculated by finding the probability that there are $(k-l)$ close windows in B outside the close window regions in A. For simplicity sake and in

most practical cases, we can assume that the l close windows in A are grouped together to form one close window region.

The probability of finding $(k-l)$ close windows in frame B outside the close window region in A is directly depended on the distribution of busy slots in the scanning region. Figure 4.6 depicts the situation where the scanning region in B with respect to the close region in A. When $l \geq w-1$, the scanning region contains $F-l+(w-1)$ slots and has $(w-1)$ slots inside the close window region. The inclusive of these $(w-1)$ slots is to determine if the window in B, which is in front of the close window region in A, is closed. On the other hand, when $l < w-1$, the scanning region is the entire frame. In this case, to compute the number of close windows outside the close window region, it is necessary to find the number of close windows overlap between the two frames. The probability distribution of close windows found in these two cases will be presented separately.

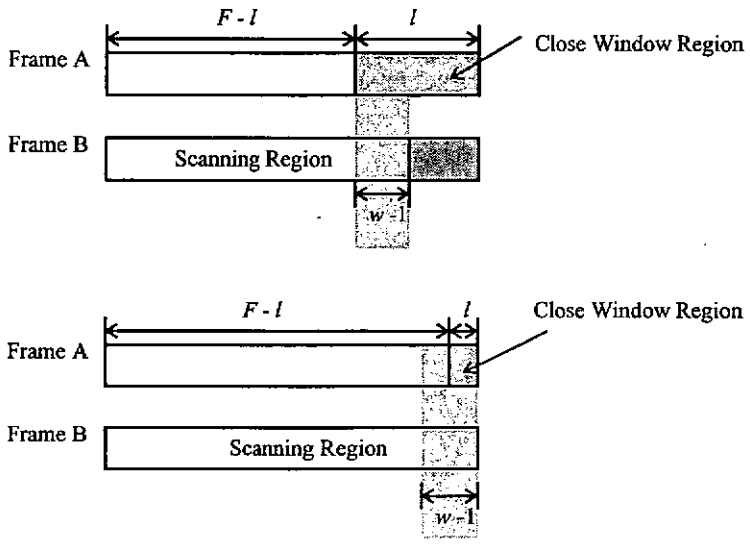


Figure 4.6 The scanning region in B with respect to the close window region of length l in A.

In the case of $l \geq w-1$, the occurrence of close windows in the scanning region is based on the distribution of busy slots in it. For the sake of clarification in the following analysis, busy slots in the scanning region are first represented in terms of different slot group g_i. Each g_i contains $(i-1)$ busy slots followed by an idle slot with $(1 \leq i \leq F)$. To ensure that all busy slots anywhere in the scanning region are being represented, an idle slot is attached at the end of the region to form a scanning frame. There may be more than one slot group in the scanning frame. Distribution of them in the scanning frame is represented by a vector $s=(n_1,n_2,\ldots n_{F-l+w})$ in which n_i denotes the number of slot group g_i found. Figure 4.7 is an example of this representation scheme. In the example, after attaching an idle slot at the end of the region, there are two g_1, one g_2, and two g_3 found. The corresponding distribution vector $s=(2,1,2,0,0,0,0,0,0,0)$ provided that $F-l+w=10$.

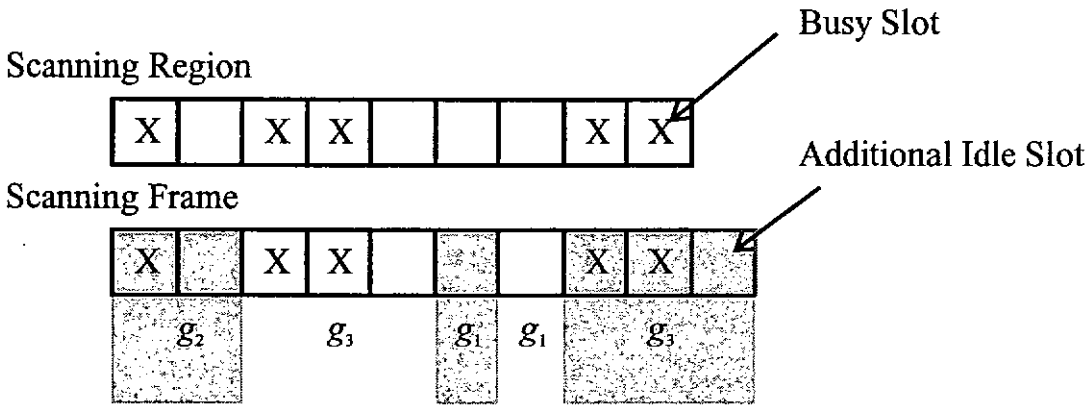


Figure 4.7 Scanning region and the corresponding scanning frame.

Consider the scanning frame in B, which is represented by the distribution vector $s=(n_1,n_2,\ldots n_{F-l+w})$, the total number of close windows in it $C(s)$ can be obtained by finding the number of close windows carried in each slot group. It is thus derived by

subtracting the number of open windows associated with the last idle slot in each slot group from the total number of slots in the scanning frame. The $C(s)$ is given as

$$\begin{aligned} C(s) &= \sum_{i=1}^{F-l+w} n_i \cdot (\text{No. of close windows found in slot group } g_i) \\ &= \sum_{i=1}^{F-l+w} n_i \cdot \max(i-w,0). \end{aligned} \tag{4.1}$$

In addition, the length of this scanning frame $N(s)$ as well as the number of busy slots found in it $B(s)$ are given by

$$N(s) = \sum_{i=1}^{F-l+w} n_i \cdot i \tag{4.2}$$

and

$$B(s) = \sum_{i=1}^{F-l+w} n_i \cdot (i-1). \tag{4.3}$$

Suppose that there are u $(u \le \min(F-l,m))$ busy slots in the scanning region such that $B(s)=u$ and $N(s)=F-l+w$. For clarity sake, we let $S_{u',u,F-l+w}$ denotes the set of s with $C(s)=u'$, $B(s)=u$, and $N(s)=F-l+w$.

$$S_{u',u,F-l+w} = \{s: C(s)=u', B(s)=u, N(s)=F-l+w\} \tag{4.4}$$

The number of ways that the scanning frame being represented by the distribution vector $s=(n_1,n_2,\ldots n_{F-l+w})$ is given by the following equation

$$\alpha(s) = \frac{\left(\sum_{i=1}^{F-l+w} n_i\right)!}{\prod_{i=1}^{F-l+w}(n_i!)}. \tag{4.5}$$

And the probability that there are u' close windows inside the scanning frame in B which contains u busy slots is

$$\gamma(u'\backslash u) = \frac{\begin{pmatrix}\text{No. of way to distribute } u \text{ busy slots}\\ \text{in the scanning frame (region) such}\\ \text{that there are } u' \text{ close windows in it}\end{pmatrix}}{\begin{pmatrix}\text{Total no. of possible distributions}\\ \text{of } u \text{ busy slots in the scanning}\\ \text{frame (region)}\end{pmatrix}} \tag{4.6}$$

$$= \begin{cases} \dfrac{\sum_{s \in S_{u',u,F-l+w}} \alpha(s)}{\binom{F-l+w-1}{u}} & \text{if } S_{u',u,F-l+w} \neq \phi, \\ 0 & \text{otherwise.} \end{cases}$$

Therefore, the probability of finding k busy slots in the AND frame generated by the frame A and B provided that $l \geq w-1$ is equal to that of finding $k-l$ close windows in the scanning frame in B. It is given as

$$\varphi(k \setminus l,m) = \sum_{u=\max(0,m-l+w-1)}^{\min(m,F-l+w-1)} \left(\begin{array}{l} \text{Prob. of having } u \text{ busy slots in the} \\ \text{scanning frame in B given that B} \\ \text{contains } m \text{ busy slots} \end{array} \right) \cdot \gamma(k-l \setminus u)$$

$$= \sum_{u=\max(0,m-l+w-1)}^{\min(m,F-l+w-1)} \frac{\binom{F-l+w-1}{u} \cdot \binom{l-w+1}{m-u}}{\binom{F}{m}} \cdot \gamma(k-l \setminus u)$$

when $l \geq w-1$.

(4.7)

For the case $l < w-1$: to calculate the probability distribution of the busy slots in the resultant AND frame, we first find the probability distribution of close windows in frame B and calculate the number of close windows overlapping between A and B. Similar to the case of $l \geq w-1$, the number of close windows found in B is obtained via a distribution vector s in the same way as that previously described. As the state of the last $(w-1)$ windows are related to the first few slots in the same frame, we consider frame B as a ring by connecting the last slot to the first one prior to representing it through a distribution vector s. Figure 4.8 shows an example how to represent a ring by a distribution vector s. After connecting the slot #7 to slot #0, the resultant ring contains one g_2 and two g_3, and is represented by the distribution vector $s = (0,1,2,0,0,0,0)$. Note that s cannot represent the particular case when there is no idle slot in a frame.

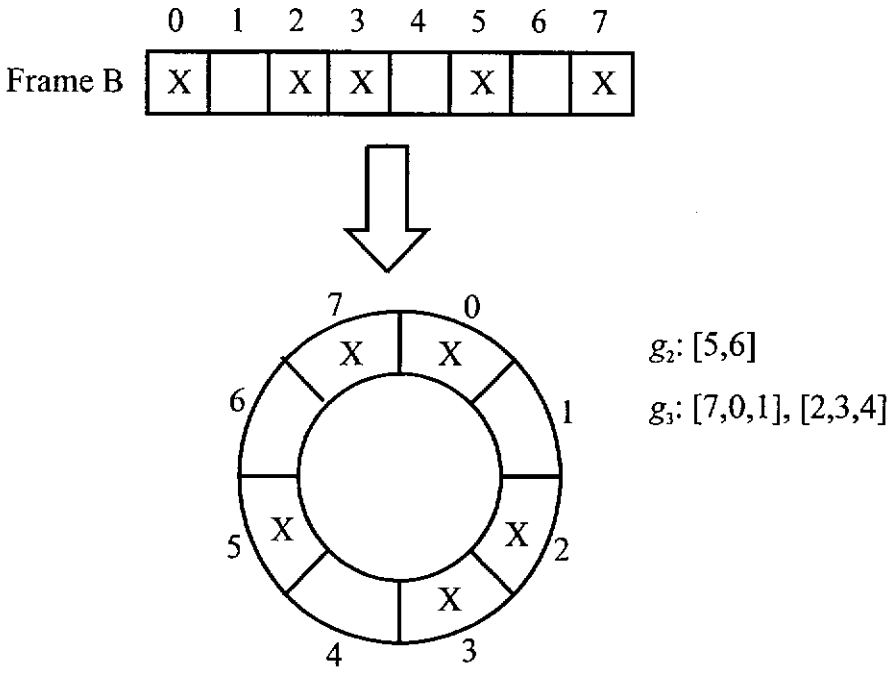


Figure 4.8 The representation of the frame B by a distribution sequence *s*.

The number of ways that B has a distribution vector $s = (n_1, n_2, \ldots n_F)$ is given by

$$\beta(s) = \frac{\left(\left(\sum_{i=1}^{F} n_i\right) - 1\right)!}{\prod_{i=1}^{F} (n_i!)} \cdot F \,. \tag{4.8}$$

To calculate the number of close windows overlapping in A and B, says l', we need the following Lemma.

Lemma 4.1: Considering a frame with a distribution vector $s = (n_1, n_2, \ldots n_F)$ has l' $(0 \le l' \le l)$ close windows overlap the close window region of size l $(l < w)$ in the other frame. The probability distribution of overlapping close windows $\delta(l')$ is given by

$$\delta(l'\backslash s)=\begin{cases}\dfrac{\sum_{i=l}^{w-1} n_i\cdot i+\sum_{i=w}^{F} n_i\cdot(w-l+1)}{F} & l'=0,\\ \dfrac{n_{w+l'}\cdot(l-l'-1)+\sum_{i\geq w+l'}^{F} n_i\cdot 2}{F} & 0<l'<l,\\ \dfrac{\sum_{i\geq l+w}^{F} n_i\cdot(i-w-l+1)}{F} & l'=l.\end{cases} \tag{4.9}$$

Its proof is presented in Appendix C.

As a result, the probability of finding k busy slots in the AND frame generated by the frame A and B in the case of $l<w-1$ is

$$\varphi(k\backslash l,m)=\begin{cases}\dfrac{\sum_{0\leq l'\leq l}\sum_{s\in S_{k-l+l',m,F}}\delta(l'\backslash s)\cdot\beta(s)}{\binom{F}{m}} & \text{if } S_{k-l+l',m,F}\neq\phi \quad \forall 0\leq l'\leq l,\\ 0 & \text{otherwise.}\end{cases} \tag{4.10}$$

Considering the output frame containing j busy slots, the probability that there are j' close windows in it is given by,

$$\tau(j'\backslash j)=\begin{cases}\dfrac{\sum_{s\in S_{j',j,F}}\beta(s)}{\binom{F}{j}} & \text{if } S_{j',j,F}\neq\phi,\\ 0 & \text{otherwise.}\end{cases} \tag{4.11}$$

The probability of finding j' busy slots in the AND frame generated by n output frames given that one of them contains j busy slots is obtained by a recursive probability function $Q_n(j'\backslash j)$ with the initial condition $Q_1(j'\backslash j)=\tau(j'\backslash j)$. The probability function $Q_n(j'\backslash j)$ is shown as follows,

$$Q_n(j'\backslash j)=\sum_{l=0}^{j'}\sum_{m=0}^{F}\varphi(j'\backslash l,m)\cdot Q_{n-1}(l\backslash j)\cdot q(m) \qquad n\geq 2 \tag{4.12}$$

where $q(m)$ is the probability that the output frame contains m busy slots, and the resultant AND frame is generated based on the state of the slot in the participated AND frame with considering busy slot as logic '0'.

In addition, the probability that there are j busy slots in the AND frame generated by n output window frames is

$$Q_n(j)=\sum_{l=0}^{F}Q_n(j\backslash l)\cdot q(l) \qquad n\geq 2. \tag{4.13}$$

To calculate the call blocking probability, we compute the probability of finding a matched idle slot and open windows in both input frame and every output frame. The

existence of this common idle slot and open windows is indicated by finding an idle slot in the AND frame generated by the input frame and the AND frame of all the requested output window frames with respect to the state of their slots. Since busy slots are randomly distributed in the input frame, we use the probability function derived in [KIM 92a]. Consider that there are a and b busy slots in the input frame and the output AND frame, respectively. The probability that the AND frame generated from them contains c busy slots is

$$\pi(c \setminus a,b) = \begin{cases} \dfrac{\binom{a}{c-b}\cdot\binom{F-a}{c-a}}{\binom{F}{b}} & \text{if } \max(a,b) \le c \le \min(a+b,F), \\ 0 & \text{otherwise.} \end{cases} \tag{4.14}$$

Suppose the input frame contains k busy slots, an arrived call is connected if there is at least one idle slot in the AND frame generated by this input frame and the AND frame of every output window frame. The effective call arrival rate at the input frame is

$$\lambda_I(k) = \begin{cases} \lambda \cdot \left(\sum_{j=0}^{F} (1 - \pi(F \setminus k, j)) \cdot Q_D(j) \right) & 0 \le k < F, \\ 0 & k = F. \end{cases} \tag{4.15}$$

The effective call arrival rate at the output frame $\lambda_O(k)$ is derived in a similar way. The difference is that the AND frame concerned here is generated by the input frame and the other output window frames and the average arrival traffic in this case is D times the input one. $\lambda_O(k)$ is given as follows

$$\lambda_O(k) = \begin{cases} D \cdot \lambda \cdot \left(\sum_{i=0}^{F} \sum_{j=F-i}^{F} (1 - \pi(F \setminus i, j)) \cdot p(i) \cdot Q_{D-1}(j \setminus k) \right) & 0 \le k < F, \\ 0 & k = F. \end{cases} \quad (4.16)$$

where $p(i)$ is the probability that the input frame contains i busy slots.

The steady-state probabilities in the input Markov chains here are obtained by the following flow equations, which govern the flow in this chain.

$$\dot{p}(k) = \begin{cases} -\lambda_I(0) \cdot p(0) + \mu \cdot p(1) & k = 0, \\ \lambda_I(k-1) \cdot p(k-1) - \left(\lambda_I(k) + k \cdot \mu\right) \cdot p(k) + (k+1) \cdot \mu \cdot p(k+1) & 1 \le k \le F-1, \\ \lambda_I(F-1) \cdot p(F-1) + F \cdot \mu \cdot p(F) & k = F. \end{cases}$$

(4.17)

Similarly, the flow equation employed to determine the state probability in the output Markov chain is given as

$$\dot{q}(k) = \begin{cases} -\lambda_O(0) \cdot q(0) + \mu \cdot q(1) & k = 0, \\ \lambda_O(k-1) \cdot q(k-1) - \left(\lambda_O(k) + k \cdot \mu\right) \cdot q(k) + (k+1) \cdot \mu \cdot q(k+1) & 1 \le k \le F-1, \\ \lambda_O(F-1) \cdot q(F-1) + F \cdot \mu \cdot q(F) & k = F. \end{cases}$$

(4.18)

As the input and output Markov chains are ergodic such that for a given non-zero set of transition rates, both sets of $\dot{p}(k)$ and $\dot{q}(k)$ will approach zero and a steady-state exists. The calculation of $\lambda_I(k)$ and $\lambda_O(k)$ here are depended upon the unknown steady-state $p(k)$ and $q(k)$, which greatly complicates the computation process. In this case, we solve this problem via the numerical iterative method in Chapter 3, where the initial values of $\lambda_I(k)$ and $\lambda_O(k)$ are obtained by replacing $p(i)$ and $q(i)$ with the probability of having i busy servers in the F-server loss systems of the same average arrival rate and holding time as the input frame and the output frame, respectively [Kleinrock, 1975].

Finally, the call blocking probability P_B is the probability of finding no idle slot in the AND frame generated from the addressed input frame and output window frames. P_B is given by

$$P_B = \sum_{i=0}^{F} \sum_{j=F-i}^{F} \pi(F \setminus i, j) \cdot p(i) \cdot Q_D(j). \tag{4.19}$$

The above expressions thus derived are used to evaluate the switch performance.

4.4 Performance Evaluation

In Figure 4.9, the call blocking probabilities of random window scheduling are plotted for various values of w with fixing frame size F at 48. For a given $D = 8$, the call blocking probability decreases with w and approaches to the lower bound giving in [KIM 92a] when $w > 4$. It is because the larger the window the smaller the probability of it being closed. So the probability of finding the common idle slot and open windows increases.

To further evaluate the performance of window scheduling in multistage switch with different w, the maximum output utilization achieved by the switch to attain the desired call blocking probability, here it is 0.01, is investigated and plotted in Figure 4.10 for various call fanout D. It has been observed that in the case of $D = 1$, the improvement achieved by window scheduling is small because of the small performance difference between the normal random call scheduling $(w = 1)$ and the lower bound. There is less incentive to use window scheduling in this case. However, when D is fairly large, say > 4, the use of window scheduling improves the performance substantially. The result with $w = 6$ seems to be sufficient to achieve the performance approaching the lower bound in the range of fanout we considered.

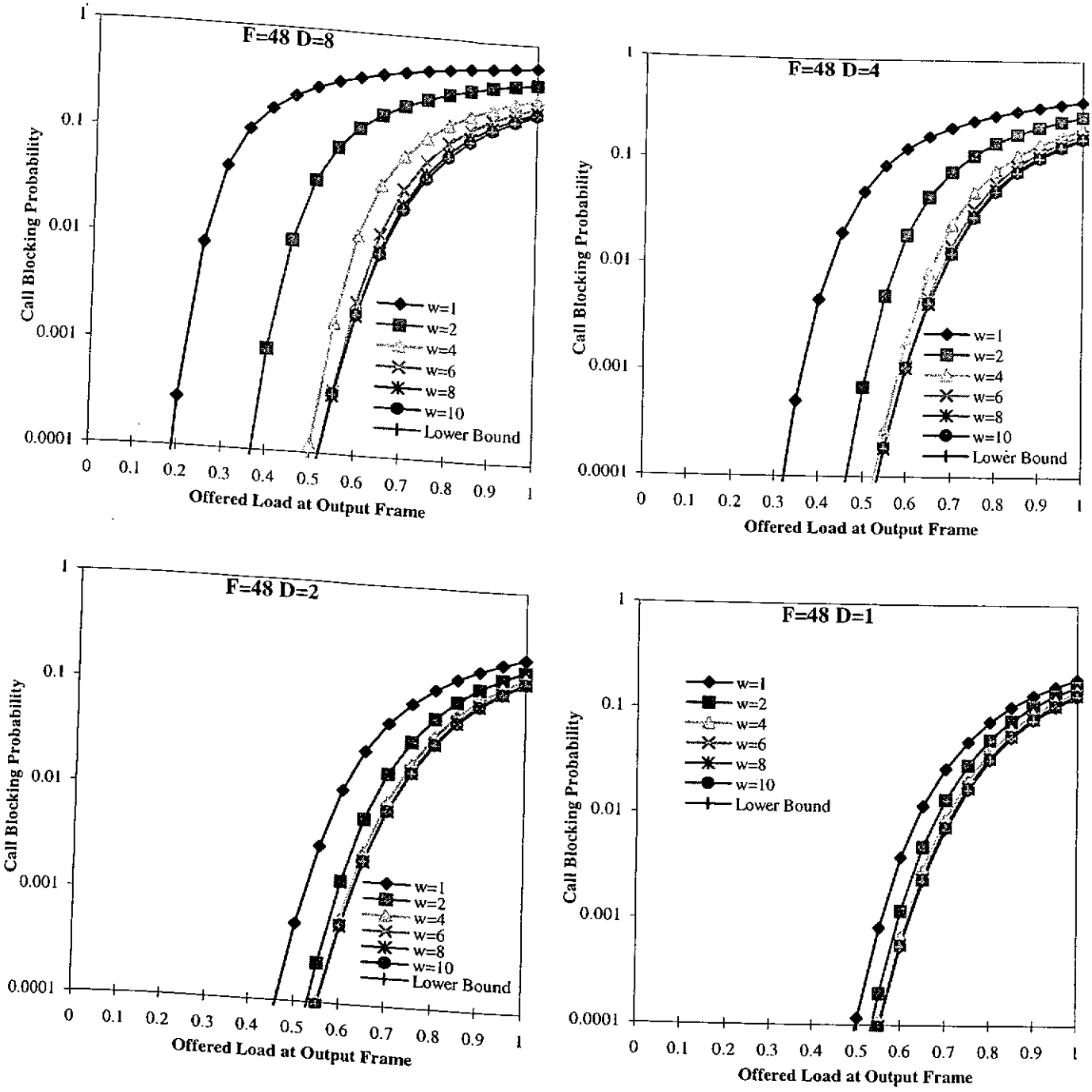


Figure 4.9 The call blocking probability encountered in a switch under random window scheduling with different window size w.

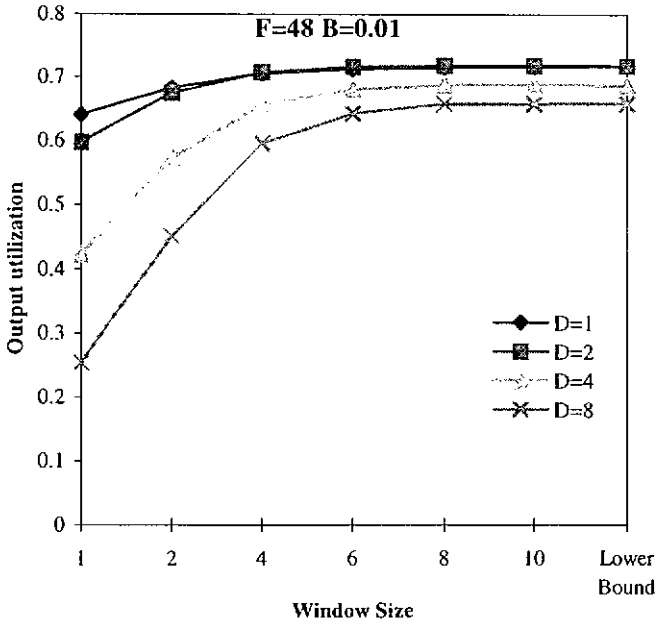


Figure 4.10 Output utilization against window size w with call blocking probability fixed at 0.01.

In order to study the effect of w on the frame size F, the output utilization with call blocking probability fixed at 0.01 is plotted in figure 4.11 for different values of F and w. It has seen that as the frame size increases, the window size w required to approach the lower bound increases too. It is because the larger the frame the greater the difficulty in matching the input idle slot to every output open window with a fixed window size.

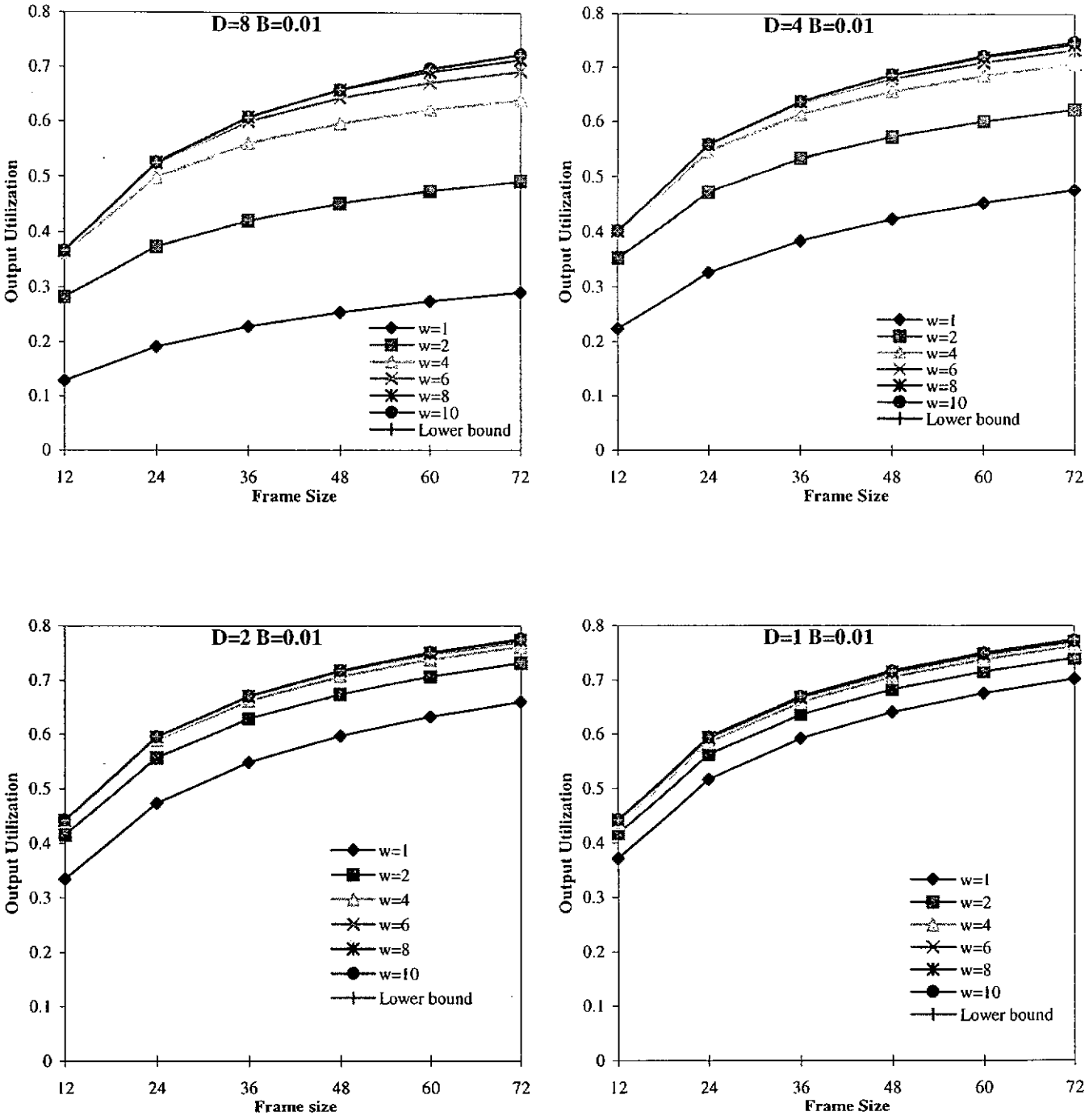


Figure 4.11 The effect of frame size on the random window scheduling.

For clarity sake, we make use of a parameter, percent deviation, to determine how large w should be in order to attain near-optimal results. The percent deviation of the multistage switch from the optimal scheduling is plotted in figure 4.12 with fixing call blocking probability $B = 0.01$ and varying the values of F, D and w. In the plots, we observe that for a given w, the change in the percent deviation decreases as F increases. The switch with $w > 1$ has a slight increment in percent deviation while frame size increases. When $w \geq 6$, the percent deviation achieved by the switch is always below 10% throughout all the interesting conditions. In order to design a cost-effective switch, we are better to select w within this range.

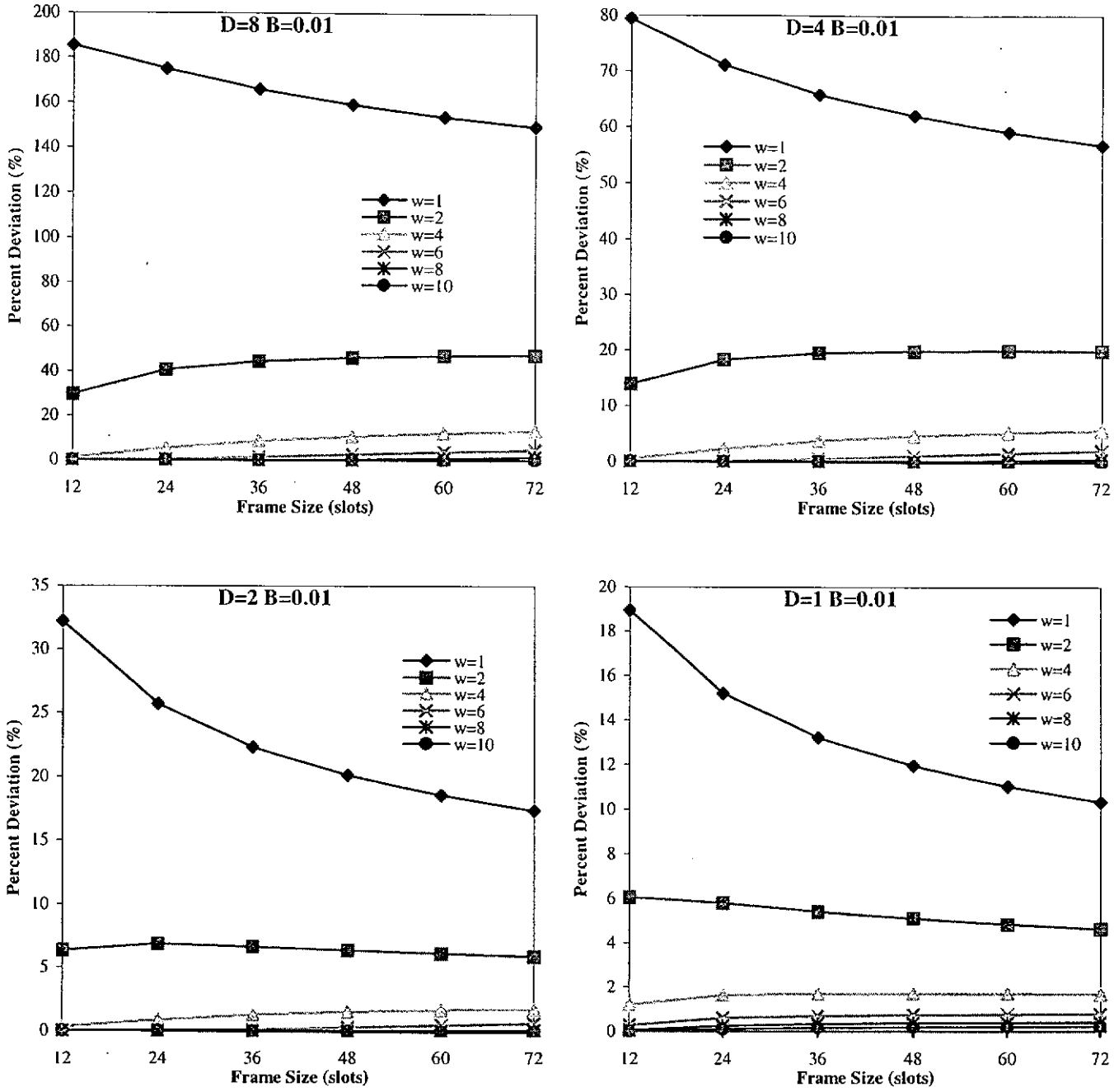


Figure 4.12 The effect of frame size on the percent deviation of the switch from the lower bound.

To verify the analytical results, we predict the call blocking probability of the switch of size $N = 256$ by means of computer simulations. It has been shown that there is an excellent match between the simulation results and the analytical results for various values of F, D and w. This implies that our analytical model is valid for use.

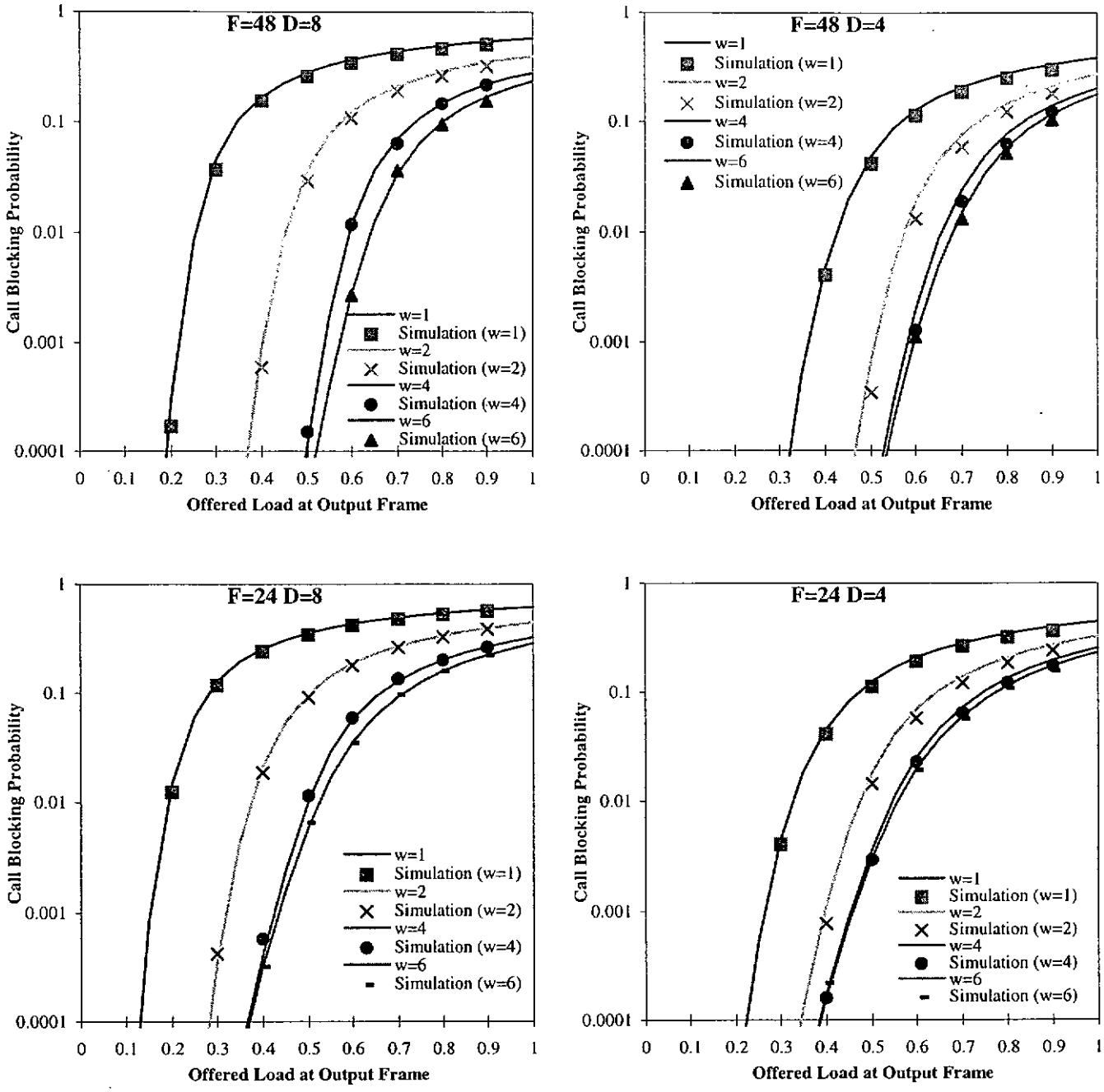


Figure 4.13 The comparison of analytical results and simulation results.

The further comparison between the simulation results and the analytical results are shown in Appendix D.

4.5 Chapter Summary

In this Chapter, we have developed a window scheduling to improve the performance of TDM switching and proposed a multistage switch for its implementation in all-optical TDM systems. The performance analysis of the window scheduling on homogeneous multicast circuit-switched traffic is given. It shows that the call blocking probability decreases substantially with w irrespective of the values of the call fanout D. Compared to the demux-mux switch presented in Chapter 3, this approach saves a large number of switching elements for achieving the same extent of output utilization at the desired call blocking rate.

Chapter 5 Time-Space-Time Switch with Intermediate Dilation

In this chapter, the performance of a time-space-time (TST) TMS with intermediate dilation is evaluated. By increasing the internal bandwidth as well as implementing trunk grouping on both sides of a central space-division switch, the switch can eliminate almost all the slot contention without working under any sophisticated scheduling algorithms.

5.1 Design of TST Switch

A switch concerned here is a time-space-time (TST) network and performs TDM switching. Its input stage and output stage consist of time slot interchangers (TSI) with fanout equals two. Each packet in the input frame is first switched by an input TSI to any appropriate position in one of the two input sub-frames. The following space switch with two ports per address exchanges these packets (time slots) to different output sub-frames, which are then collected by corresponding output TSI's to form output frames. Call scheduling in this case is carried on the addressed sub-frames where a new call will be connected via the matched idle slots in them. Figure 5.1 shows the structure of the proposed TST switch, or called dilated switch for short.

The dilated switch has very high potential performance. It makes use of trunk grouping at both sides of a central switch to relieve the contention of output idle slots (free ports) and provide more freedom to route packet streams. Besides, with the fanout in a TSI, the internal bandwidth is twice as much as the external one carried in each input (output) frame. This creates more capacity to schedule a call and hence increases the occurrence of matched idle slots in the requested sub-frames. Based on the Clos strictly non-blocking principle [CLOS 53], the dilated switch is strictly non-blocking for switching basic rate point-to-point call. For the case of multicast traffic, the switch can also eliminates most slot contentions and attains near-optimal performance. Figure 5.2 illustrates the way to connect a multicast call in a dilated switch. In the example, there are several ways to connect call H from input #2 to output #1 and #2, which encounters slot contention in a simplex TMS. This gives an insight on the potential performance of the dilated switch.

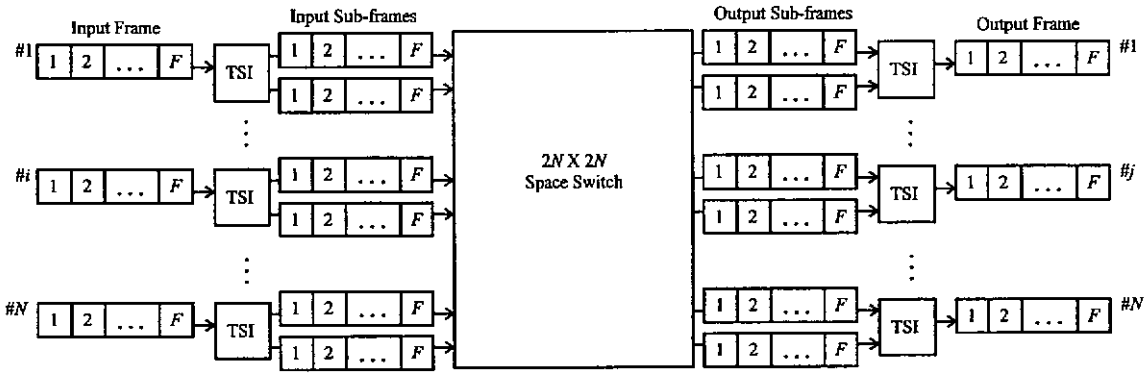


Figure 5.1 General structure of dilated switch.

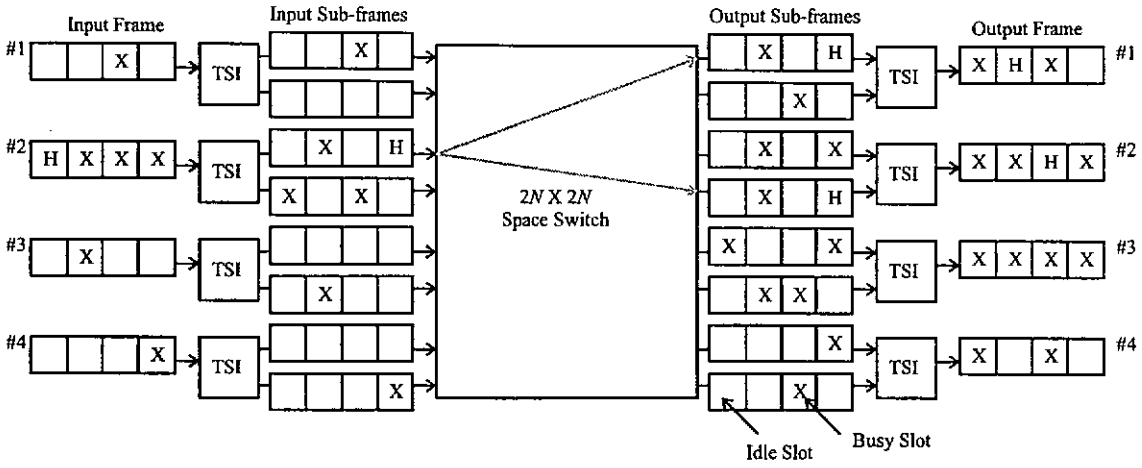


Figure 5.2 Example of call scheduling in a dilated switch.

5.2 Modeling and Analysis

In this section, call blocking probability of multicast traffic in the dilated switch is derived under purely random call scheduling. The switch performance analyzed here can be considered as upper bound call blocking probability because most sophisticated scheduling algorithms would have performance better than the random one.

For simplicity sake, the traffic is assumed to be uniformly distributed among the input ports and the output ports. The call arrival process at each input is Poisson distributed with average arrival rate λ and its holding time is an exponential distribution of mean $1/\mu$. Each input (output) frame has F slots and supports at most F calls concurrently. A call considered here is of a basic-rate and has a constant number of destinations (call fanout) D. In the analysis, we assume that the call fanout D is very small compared with the switch size N to ignore the effect of interdependency between the sub-frames of different address ports [ROSE 87], [KIM 92a].

Based on these assumptions, the conjugate sub-frame pairs at the input port and the output port can be modeled by two two-dimensional continuous-time Markov chains. The state of Markov chain, say (l_1,l_2), represents the distribution of busy slots in the corresponding sub-frame pair in a similar way as was demonstrated in Chapter 3. In this case, the sub-frame pair of F slots per frame contains F slot columns each of which is constituted by the slots matched in position (coincided in time) in the two sub-frames. The variable l_1 is the number of slot columns with one busy slot each. On the other hand, the l_2 shows how many slot columns saturated with busy slots.

Since a call can be blocked either because of the overload in one or more of the requested lines or due to the slot contention, we employ effective arrival rate $\lambda(l_1,l_2)$ instead of average arrival rate λ as the transition probability in the Markov chains. The effective arrival rate $\lambda_0(l_1,l_2)$ here shows the average number of call arrivals excluding the blocked call at one of the $(F-l_1-l_2)$ slot groups each contains no busy slot given that there are l_2 saturated slot columns. Similarly, the effective arrival rate at the slot column with one busy slot is given by $\lambda_1(l_1,l_2)$. In the analysis, we denote $\lambda_{I,}(l_1,l_2)$ and $\lambda_{O,}(l_1,l_2)$ as effective call arrival rate at the input sub-frame pair and the output sub-frame pair, respectively. Both of them are based on the steady state probabilities of the Markov chains which are generated through the numerical iteration method described in Chapter 3. The continuous-time Markov chains for both input and output conjugate sub-frames are shown in Figure 5.3.

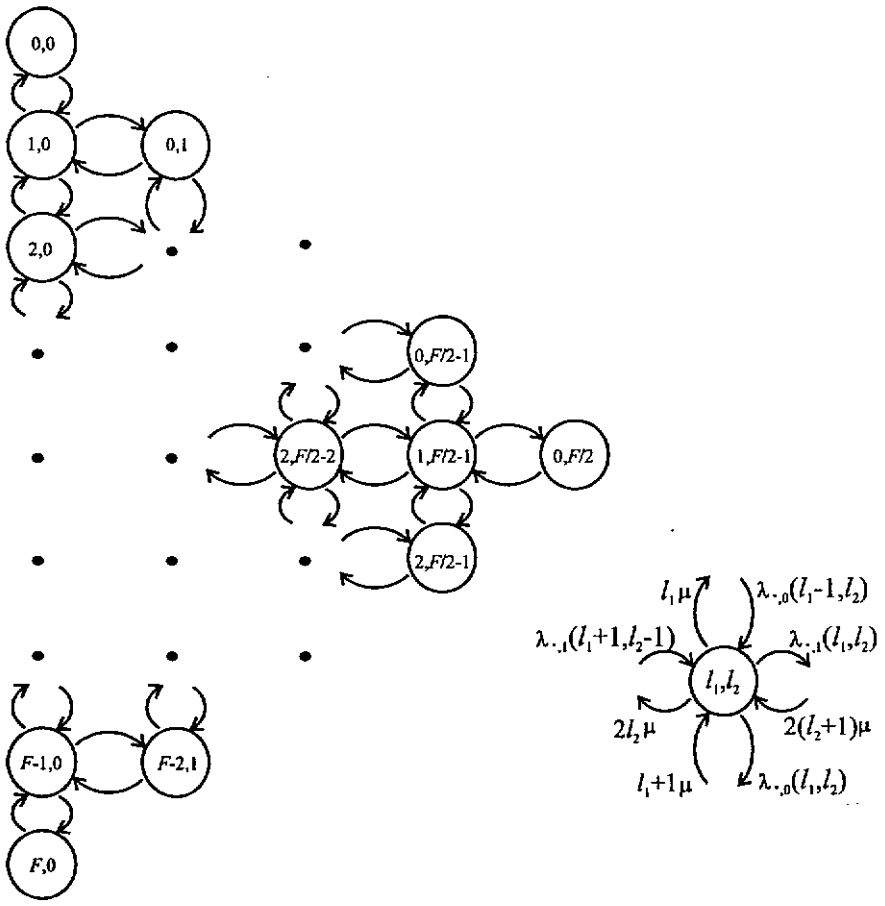


Figure 5.3 2-dimensional continuous-time Markov chain for input (output) conjugate sub-frame pair where F is a multiple of 2.

To connect a call, we have to find common idle slots from the sub-frames of different addressed port. By means of the input TSI, a call can be placed to any appropriate idle slot in the unsaturated slot columns of the same input sub-frame pair. The occurrence of unsaturated slot columns at the same position in these sub-frame pairs therefore indicates the existence of common idle slots in the corresponding sub-frames at these positions.

The number of common idle slots between the addressed sub-frame pairs is calculated by finding the probability distribution of saturated slot columns in each

sub-frame pair. For simplicity, we assume that slot columns are randomly distributed in each sub-frame pair. The probability of finding $(F-k)$ common unsaturated slot columns from the two sub-frame pairs with i and j saturated slot columns is derived by a probability function $\pi(k \setminus i, j)$ that there are k busy slots in the AND frame generated from these sub-frame pairs with considering saturated slot column as logic "0" and unsaturated slot column as logic "1". The AND frame is a logic frame and has been widely used in the previous chapters. The probability function $\pi(k \setminus i, j)$ is given by

$$\pi(k \setminus i, j) = \begin{cases} \dfrac{\binom{i}{k-j}\cdot\binom{F-i}{k-i}}{\binom{F}{j}} & \text{if } \max(i,j) \le k \le \min(i+j, F), \\ 0 & \text{otherwise.} \end{cases} \tag{5.1}$$

In order to derive the probability distribution of matched unsaturated slot columns, we calculate the probability distribution of unsaturated slot columns in the input sub-frame pair and every requested output sub-frame pair. The probability function that there are i saturated slot columns in the input sub-frame pair excluding the event of overload $P(i)$ is given by

$$P(i) = \begin{cases} \sum_{l_1=0}^{F-2i-1} p(l_1, i) & \text{if } 0 \le i \le \lfloor (F-1)/2 \rfloor, \\ 0 & \text{otherwise,} \end{cases} \tag{5.2}$$

where $\lfloor x \rfloor$ is the largest integer smaller than or equal to x and $p(l_1, l_2)$ is a steady state probability distribution in the input Markov chain.

In addition, we also define a similar probability function $Q(j)$ for the output sub-frame pair which are given by the following equation.

$$Q(j) = \begin{cases} \sum_{l_1=j}^{F-2\cdot j-1} q(l_1, j) & \text{if } 0 \le j \le \lfloor (F-1)/2 \rfloor, \\ 0 & \text{otherwise.} \end{cases} \tag{5.3}$$

Note that $q(l_1, l_2)$ is a probability distribution of different slot columns in the output sub-frame pair.

To find the probability distribution of the common unsaturated slot columns in the requested output sub-frame pairs, we derive a recursive probability function $Q_n(j)$ that there are exactly j busy slots in the AND frame generated by n output sub-frame pairs given that the participated output lines are all unsaturated. $Q_n(j)$ is given by the following equation with the initial condition $Q_1(j) = Q(j)$

$$Q_n(j) = \sum_{l=0}^{j} \sum_{m=j-l}^{j} \pi(j \setminus l, m) \cdot Q_{n-1}(l) \cdot Q(m) \qquad n \ge 2. \tag{5.4}$$

Let suppose that the input line has distribution vector $l = (l_1, l_2)$. The effective arrival rate to the empty slot columns $\lambda_{1,0}(l_1, l_2)$ is the probability that an arrived call finds at

least one idle slot in the AND frame generated from this sub-frame pair and all the (D) requested output sub-frame pairs in the positions of the empty slot columns and is placed to it by the switch. The $\lambda_{1,0}(l_1,l_2)$ is given by

$$\lambda_{1,0}(l_1,l_2)=\begin{cases}\lambda\cdot\left(\sum_{k=0}^{F-1}\sum_{j=0}^{F-1}\sum_{u=1}^{F-l_1-l_2}\begin{pmatrix}\text{Prob. of choosing}\\ \text{empty slot column}\end{pmatrix}\cdot\theta(k,u\setminus l_2,F-l_1-l_2,j)\cdot Q_D(j)\right) & \text{if } l_1+l_2<F,\\ 0 & \text{otherwise,}\end{cases}$$

$$=\begin{cases}\lambda\cdot\left(\sum_{k=0}^{F-1}\sum_{j=0}^{F-1}\sum_{u=1}^{F-l_1-l_2}\frac{u}{F-k}\cdot\theta(k,u\setminus l_2,F-l_1-l_2,j)\cdot Q_D(j)\right) & \text{if } l_1+l_2<F,\\ 0 & \text{otherwise.}\end{cases}$$

(5.5)

where $\theta(u,k\setminus n,m,j)$ is the probability function derived in Chapter 3. It gives the probability that a sub-frame pair with n saturated slot columns and m desired unsaturated slot columns (each contains a fixed number, say x $(0\le x<2)$, of busy slots) has $F-k$ unsaturated slot columns in the positions of the $(F-j)$ idle slots in the AND frame where u $(<m)$ of them are the desired slot columns. The $\theta(u,k\setminus n,m,j)$ is

$$\theta(u,k\setminus n,m,j)=\begin{cases}\dfrac{\binom{n}{k-j}\cdot\binom{m}{u}\cdot\binom{F-n-m}{F-u-k}}{\binom{F}{j}} & \text{if } \max(n,j)\le k\le\min(n+j,F)\\ & \text{and } \max(0,m-(k-n))\le u\le\min(m,F-k),\\ 0 & \text{otherwise.}\end{cases}$$

(5.6)

Similarly, the effective input arrival rate at the slot column with one busy slot $\lambda_{I,1}(l_1,l_2)$ is

$$\lambda_{I,1}(l_1,l_2)=\begin{cases}\lambda\cdot\left(\sum_{k=0}^{F-1}\sum_{j=0}^{F-1}\sum_{u=1}^{l_1}\left(\begin{array}{l}\text{Prob. of choosing}\\ \text{slot column with one}\\ \text{busy slots}\end{array}\right)\cdot\theta(k,u\setminus l_2,l_1,j)\cdot Q_D(j)\right) & \text{if } l_1+l_2<F \text{ and } l_1>0,\\ 0 & \text{otherwise,}\end{cases}$$

$$=\begin{cases}\lambda\cdot\left(\sum_{k=0}^{F-1}\sum_{j=0}^{F-1}\sum_{u=1}^{l_1}\frac{u}{F-k}\cdot\theta(k,u\setminus l_2,l_1,j)\cdot Q_D(j)\right) & \text{if } l_1+l_2<F \text{ and } l_1>0,\\ 0 & \text{otherwise.}\end{cases}$$

(5.7)

Given the distribution of busy slots $l=(l_1,l_2)$ in the output sub-frame pair, the effective arrival rate $\lambda_{O,0}(l_1,l_2)$ and $\lambda_{O,1}(l_1,l_2)$ are derived in a way similar to the input one except the resultant AND frame in this case is generated by the corresponding output sub-frame pair and the AND frame of the input sub-frame pair and the other $(D-1)$ requested output sub-frame pairs. The average call arrival rate at the output sub-frame pair is D times as much as that at the input ones. The $\lambda_{O,0}(l_1,l_2)$ and $\lambda_{O,1}(l_1,l_2)$ are shown as follows.

$$\lambda_{O,0}(l_1,l_2)=\begin{cases} D\cdot\lambda\cdot\left(\sum_{k_1=0}^{F-1}\sum_{k_0=0}^{F-1}\sum_{u=1}^{F-l_1-l_2}\frac{u}{F-k_1}\cdot\theta(k_1,u\setminus l_2,F-l_1-l_2,k_0)\cdot\left(\sum_{i=0}^{\lfloor\frac{F-1}{2}\rfloor}\sum_{j=0}^{F-1}\pi(k_0\setminus i,j)\cdot P(i)\cdot Q_{D-1}(j)\right)\right) & \text{if } l_1+l_2<F, \\ 0 & \text{otherwise,} \end{cases} \tag{5.8}$$

and

$$\lambda_{O,1}(l_1,l_2)=\begin{cases} D\cdot\lambda\cdot\left(\sum_{k_1=0}^{F-1}\sum_{k_0=0}^{F-1}\sum_{u=1}^{l_1}\frac{u}{F-k_1}\cdot\theta(k_1,u\setminus l_2,l_1,k_0)\cdot\left(\sum_{i=0}^{\lfloor\frac{F-1}{2}\rfloor}\sum_{j=0}^{F-1}\pi(k_0\setminus i,j)\cdot P(i)\cdot Q_{D-1}(j)\right)\right) & \text{if } l_1+l_2<F \text{ and } l_1>0, \\ 0 & \text{otherwise.} \end{cases} \tag{5.9}$$

Since a call connection faces both overflow blocking and slot contention blocking, we have to find their probabilities in order to calculate the overall call blocking probability. The probability function of slot contention blocking $P_{Slot\ Contention}$ is the probability that no idle slot in the AND frame generated by the input sub-frame pair and all the requested output sub-frame pairs given that all the addressed lines are unsaturated. The $P_{Slot\ Contention}$ is

$$P_{Slot\ Contention}=\sum_{i=0}^{F}\sum_{j=F-i}^{F}\pi(F\setminus i,j)\cdot P(i)\cdot Q_D(j). \tag{5.10}$$

In contrast, the overflow blocking probability $P_{Overflow}$ is the probability that at least one of the requested frames is saturated with busy slots and is given by the following equation.

$$P_{Overflow} = \left(1-(1-\Omega_I)\cdot(1-\Omega_O)^D\right). \tag{5.11}$$

Note that the Ω_I is the probability that no idle slot is found in the input frame and is shown as follows

$$\Omega_I = \sum_{l_2=0}^{\lfloor \frac{F}{2} \rfloor} p(F-2\cdot l_2, l_2) \tag{5.12}$$

Similarly, the probability of the output line being saturated Ω_O is

$$\Omega_O = \sum_{l_2=0}^{\lfloor \frac{F}{2} \rfloor} q(F-2\cdot l_2, l_2). \tag{5.13}$$

The average call blocking probability P_B is therefore obtained by adding (5.11) and (5.12) together.

$$\begin{aligned} P_B &= P_{Overflow} + P_{Slot\ Contention} \\ &= \left(1-(1-\Omega_I)\cdot(1-\Omega_O)^D\right) + \left(\sum_{i=0}^{F}\sum_{j=F-i}^{F}\pi(F\setminus i, j)\cdot P(i)\cdot Q_D(j)\right) \end{aligned}. \tag{5.14}$$

Based on the above expressions, the switch performance is thus evaluated.

5.3 Performance Evaluation

In this section, the potential performance of dilated multicast switch is evaluated by varying the parameters including frame size F and call fanout D. To justify the performance of dilated switch, we plot the lower bound blocking probability together with that of a simplex switch on the same graph. The lower bound blocking probability is exactly the probability of overflow blocking. Its derivation have been detailed in [KIM 92] and is given in Appendix A.

In figure 5.4, the call blocking probability of the dilated multicast switch with various values of D are plotted for fixing frame size F at 48. It was seen that the switch always has performance approaching the lower bound. This shows that with exploiting intermediate trunk grouping as well as increasing the internal bandwidth, the dilated switch can remove almost all the slot contentions, which is a major cause for the performance gap between a simplex switch and the lower bound. Call blocking encountered in the dilated switch is thus mainly due to the overflow of the addressed frames.

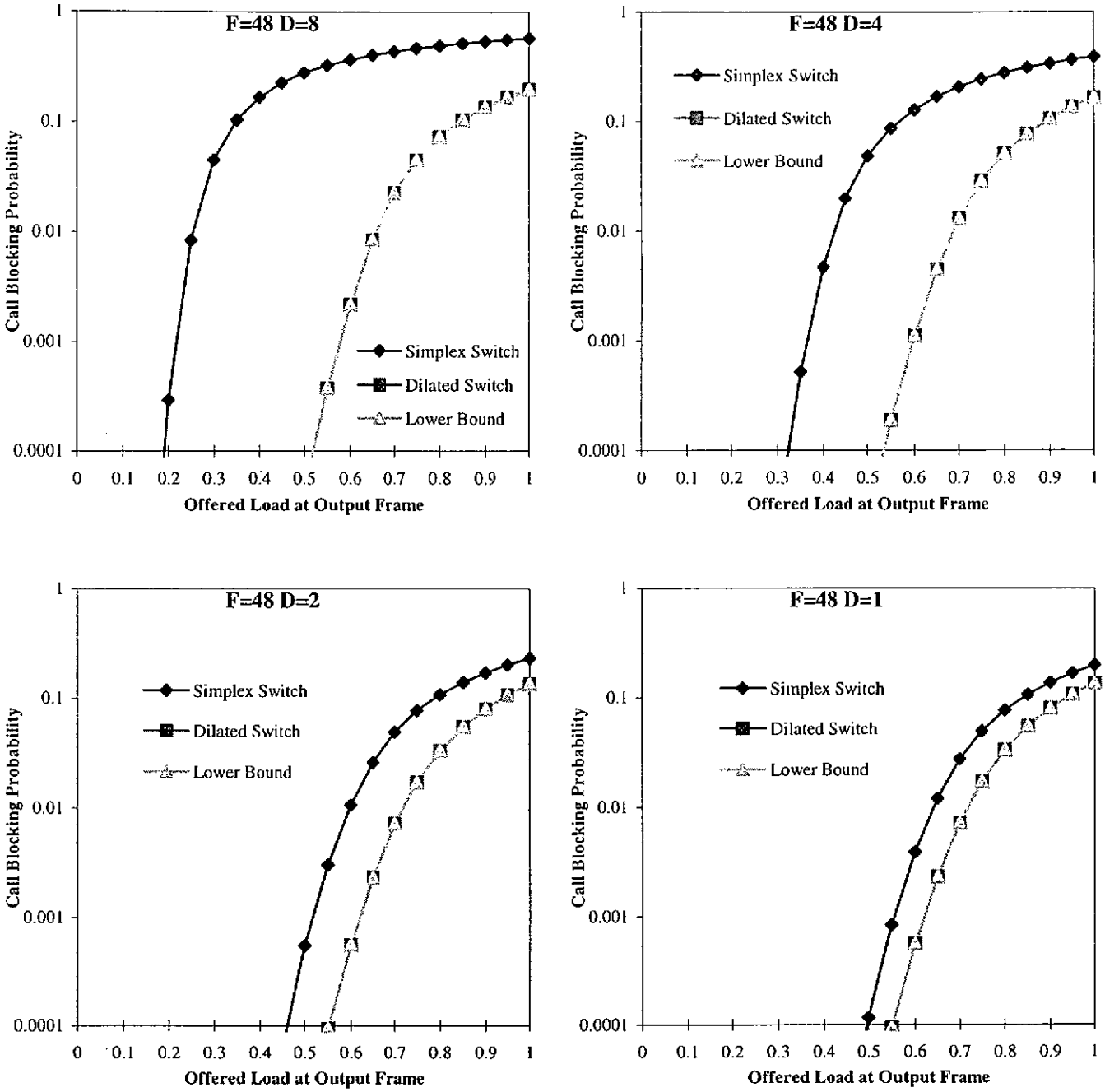


Figure 5.4 The call blocking probability in a dilated multicast switch under a random scheduling.

With fixing the call blocking probability at 0.01, the effect of frame size F is shown in figure 5.5. The plots show that throughout the range of interesting frame size and call fanout, the blocking probability of the dilated switch is always close to the lower bound. This directly shows the potential performance of the dilated switch and gives a

clear implication on the high efficiency of internal bandwidth increment on the performance improvement.

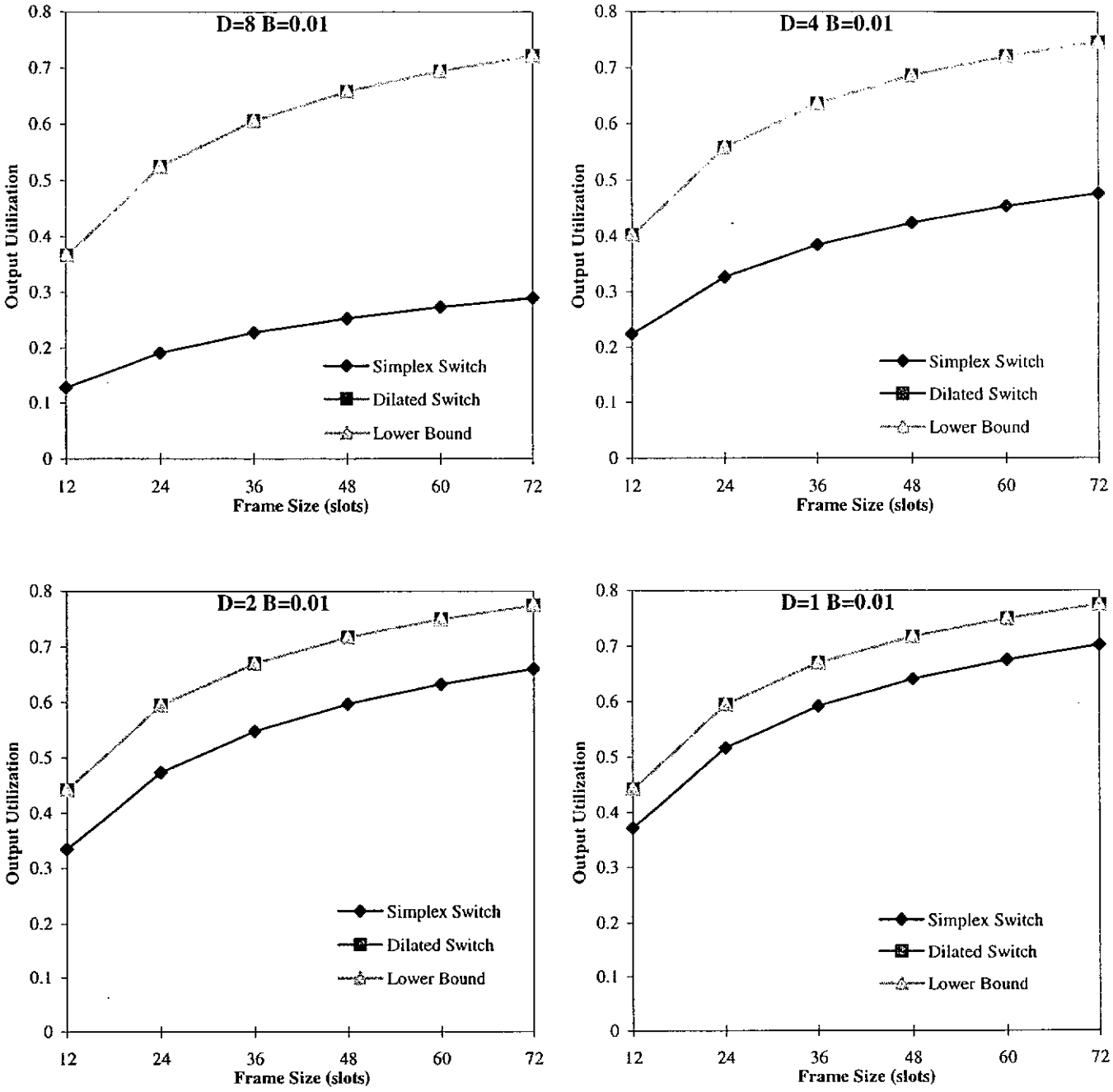


Figure 5.5 Maximum offered load at output frame with different frame size F when call blocking probability fixed at 0.01.

To check the validity of the assumptions made in the last section, the performance of dilated multicast switches with 256 ports and 32 ports were predicted via computer simulations for $F = 24$ and 48. The results are compared to the analytical ones in figure 5.6. As seen in the plot, there is a perfect match between the analytical results and the simulated ones. This suggests that the independence of the busy slots in the input and output sub-frames is a reasonable assumption.

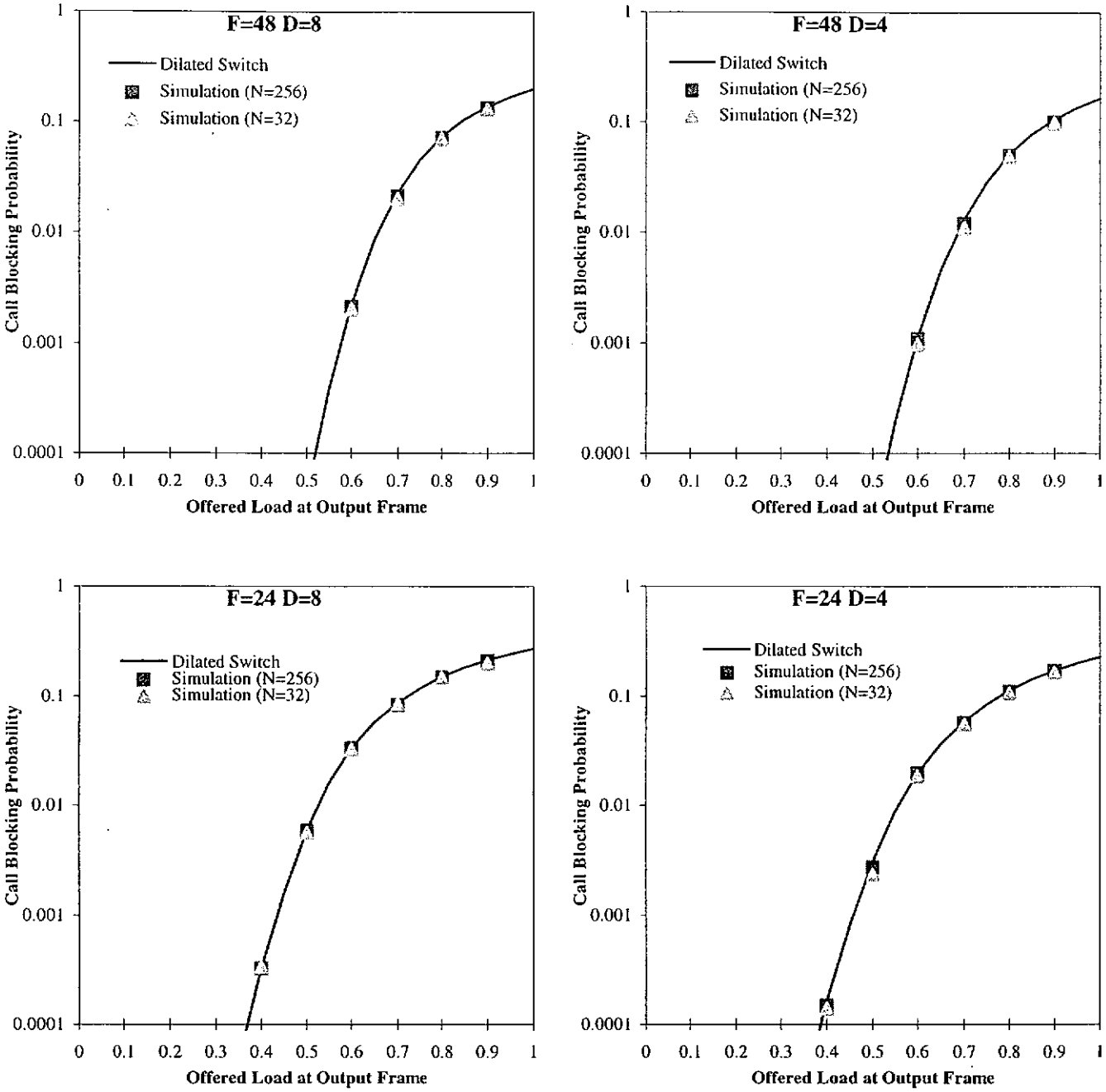


Figure 5.6 The Comparison of analytical results and simulation results for dilated switch.

5.4 Chapter Summary

The dilated switch proposed in this chapter has performance approaching the lower bound throughout the range of interesting conditions. The substantial performance improvement shows the high efficiency of the strategy, internal bandwidth expansion, in the provision of a high quality interconnection. This architecture can be applied in many existing TST switch nodes without any significant modifications in their designs. The trade off, however, is the requirement of packet buffering at both input and output stages. This greatly reduces the feasibility of implementing it in lightwave networks.

Chapter 6 Conclusions and Topics for Future Investigations

In this thesis, we tackled the TSA problems associated with multicast TMS and proposed two efficient design methods, namely, zone switching and internal bandwidth expansion, to remove call blocking effectively. In principle, both methods can significantly reduce the level of call blocking and achieve near-optimal performance. Based on these basic methods, we considered three switch architectures and evaluated their upper bound blocking probability for homogeneous multicast traffic via purely analytical means.

The first design has a demux-mux architecture, which is a building block in many existing electro-optical switching nodes. By means of demultiplexing and multiplexing operations at each input line and output line, the switch takes the advantage of both input trunk grouping and output trunk grouping which increases the occurrence of common idle slots among the addressed sub-frames and hence reduces the level of call blocking. Our results show that the greater the number of lines per trunk the better the performance achieved. It gives ramification of the less important of costly design for such switch nodes having this structure likes an electro-optical node and a full-conversion all-optical node, which can remove most of call blocking inherently, to handle multicast traffic.

The second design is a multistage switch. It is proposed to implement a novel call scheduling scheme, window scheduling, which is the simplest and an efficient way to implement zone switching. This switch design is very simple and does not require any buffering stage. Therefore, it is very suitable for using in any single hop lightwave networks. Compared with the demux-mux switch, the multistage switch has lower hardware requirement in achieving the same extent of performance.

The final switch design is a TST network with intermediate dilation. By exploiting the fanout in TSI stages, the internal bandwidth is twice as much as that carried in each input line and output line. This arrangement creates more capacity to connect a call and thus greatly reduces the output conflict without sophisticated switch control. According to the Clos Strictly non-blocking principle, the dilated switch can remove all the slot contention in switching unicast traffic. For multicast traffic, it can also eliminate most of call blocking and attains optimal performance. However, the buffering requirement at both input and output stages reduces the ease of implementing it in a lightwave system.

In summary, the designed switches greatly reduce the probability of blocking without using any sophisticated scheduling algorithms. The substantial improvement on their performance shows the high efficiency of our design methods. For cost-efficient design, it is recommended to use internal bandwidth expansion, if its implementation is feasible. Otherwise, zone switching is a good alternative to provide the same extent of improvement.

There are some possible extensions of this research. They are shown as follows:

A. Blocking Performance of General Heterogeneous Traffic

In the analysis throughout the thesis, we assume that the traffic is of basic rate and is homogeneous in call fanout. However, the real traffic is always heterogeneous in both of these fields, especially in the BISDN [KIM 96]. To provide more understanding on the switch performance in the future integrated networks, we need to investigate the blocking performance of heterogeneous traffic in our switches.

B. Performance Study for Mass Market Broadband Services

To support mass market broadband (MMBB) services, such as video on demand, and make it to be competitive, network resources must be carefully engineered to meet anticipated traffic demands cost-effectively [ERRA 94], [DELO 94]. Nowadays, many MMBB service providers employ TDM-based system to distribute their video signal because of its simple control and guaranteed grade of service. It is thus of great practical value to investigate the performance of our TMS's in MMBB applications. This study can also give us an insight on a cost-effectiveness of any switch design as well as the optimal allocation of network resources in different MMBB networks.

C. Performance Analysis of Call Scheduling Algorithms

The performance of the switches in this thesis are analyzed under a pure random call scheduling to evaluate their upper bound blocking probability. It would be interesting to investigate their performance under some sophisticated call scheduling algorithms.

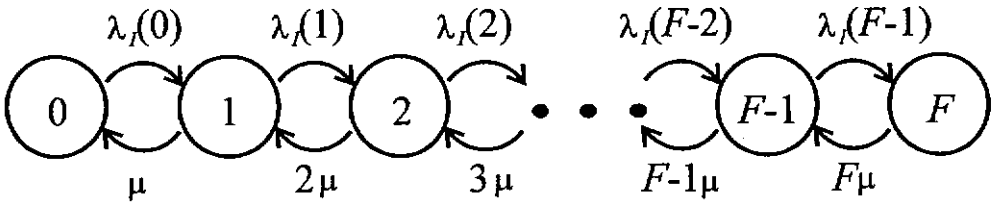
For example, with the increment in internal bandwidth, the dilated switch is very suitable to use call splitting algorithms to achieve better performance. In contrast, the demux-mux switch and multistage switch seem to best be employed call packing type algorithms to reduce the number of blocking.

D. Cost-effective Switch Design for Window Scheduling

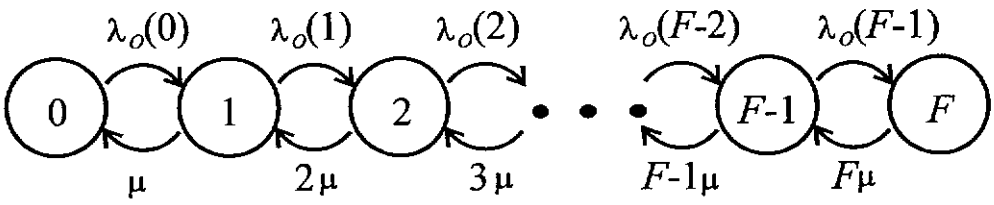
In the study of window scheduling, we found that this scheduling scheme is very efficient in reducing slot contention. The proposed multistage network is a simple and direct way to implement this scheduling scheme. However, it limits the size of windows and always requires many switch planes to attain desired performance. Therefore, it is advantage to consider other switch structures for window scheduling.

Appendix A

In this appendix, the derivation of the lower bound on the switch performance is presented for various frame size F and call fanout D. To simplify the analysis, we assume that traffic is uniformly distributed over input and output ports, and the call arrival process at each input is Poisson with arrival rate λ and the holding time of each call is exponentially distributed with mean $1/\mu$. The traffic here is assumed homogeneous where all calls have the same fixed bandwidth and fanout. For simplicity sake, the correlation between input and output frames is ignored in such a way that the number and position of busy slots in one frame are independent of other frames. Basing on these assumptions, each input/output frame can then be modeled by a continuous-time Markov chain, as shown in figure A.1. The number of busy slots in a time frame determines the state of each input or output port.



(a) Input Markov Chain



(b) Output Markov Chain

Figure A.1 The continuous-time Markov chains of input and output frame.

Since all calls in this case are blocked by frame overflow, the effective arrival rate to an input port ($\lambda_I(\cdot)$) and to an output port ($\lambda_O(\cdot)$) are the arrival rates of successful calls, which in turn depended on the blocking probability. Therefore, the steady-state probability distribution is determined via the numerical iterative method.

In order to compute the lower bound, the need to find the frame overflow probability to derive the effective arrival rates. Considering a call which arrives at an input port of state k $(0 \le k < F)$ is accepted if all the addressed D destinations have at least one idle slot. Thus, the effective arrival rate at the input of state k, $\lambda_I(k)$, is

$$\lambda_I(k) = \begin{cases} \lambda \cdot \left(1 - q(F)\right)^D & 0 \le k < F, \\ 0 & \text{otherwise.} \end{cases} \tag{A.1}$$

where $q(k)$ is the steady-state probability that there are k busy slots in the output Markov chain.

Supposing that an output with k $(0 \le k < F)$ busy slots is one of the D outputs requested by a call. This call is accepted if there are idle slots in both the input and all the other $D-1$ output frames. Hence, the effective arrival rate at the output in state k is

$$\lambda_I(k) = \begin{cases} D \cdot \lambda \cdot \left(1 - p(F)\right) \cdot \left(1 - q(F)\right)^{D-1} & 0 \le k < F, \\ 0 & \text{otherwise.} \end{cases} \tag{A.2}$$

Note that the average arrival rate to an output frame is D times larger than that of an input one because a call is multicast from one source to D destinations.

The lower bound of blocking probability B_L is defined to be the probability that an arrival call finds one or more requested frames are saturated and is given as

$$B_L = 1 - \left(1 - p(F)\right) \cdot \left(1 - q(k)\right)^D . \tag{A.3}$$

Figure A.2 shows the plots of the lower bound under various values of F and D.

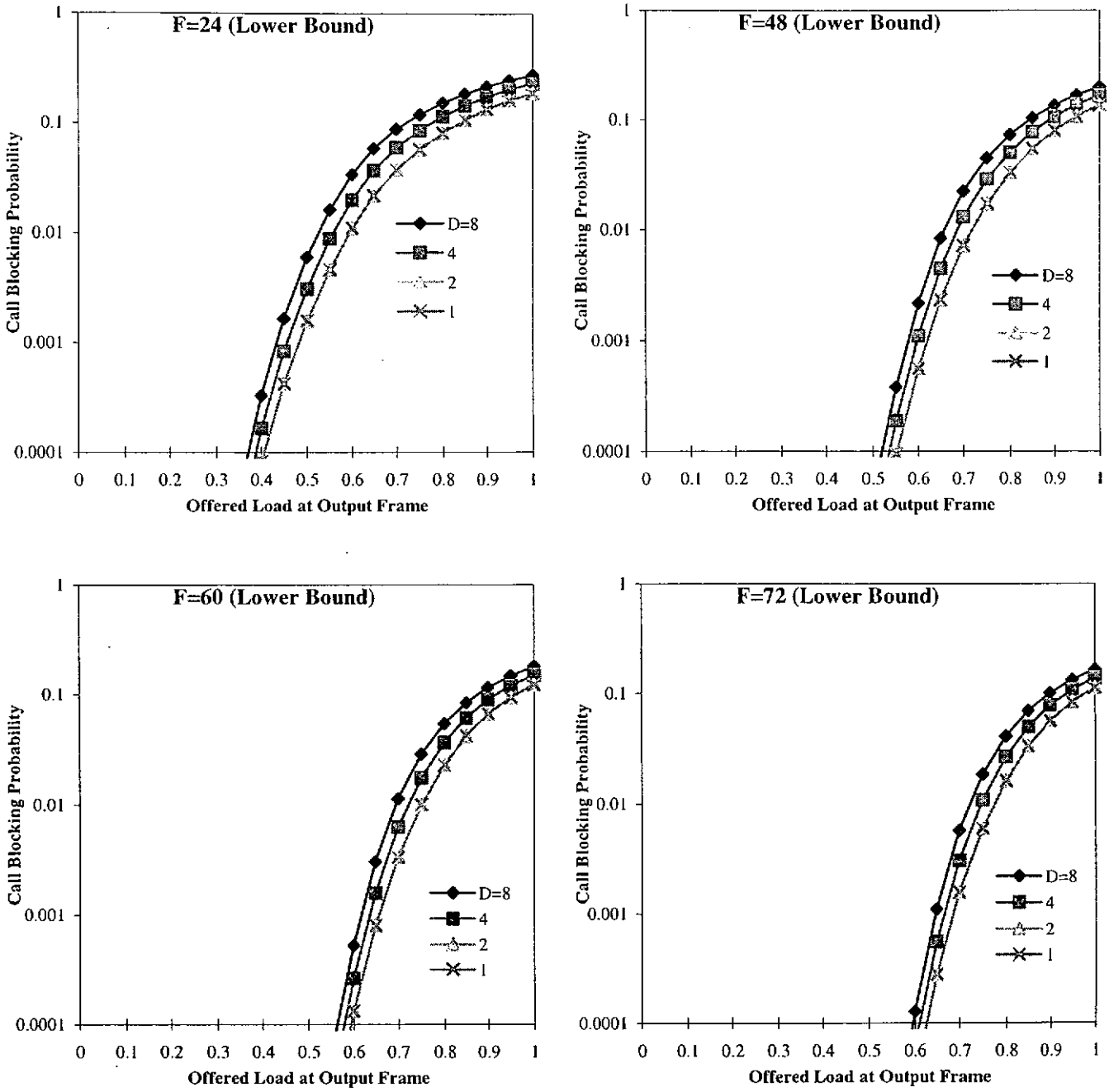


Figure A.2 The lower bound of call blocking probability for various values of F and D.

Appendix B

Figure B.1 shows a comparison of average call blocking probability predicting via purely analytical mean and computer simulations with switch size $N = 64$, respectively. It has shown that when $D = 4$, analytical results are always matched to simulation ones. However, in the case of $D = 8$, a small deviation is noted between them for various z's. It is because in this case, the number of ports in a switch (N) is comparable in value to the average call fanout D $(D/N = 8)$. According to the results in [KIM 92a], the value of D/N is not large enough to eliminate the effect of interdependence between sub-frames at different ports, which would increase the occurrence of matched idle slots among these sub-frames. Call blocking probability encountered in the computer simulation is therefore always smaller than the analytical ones.

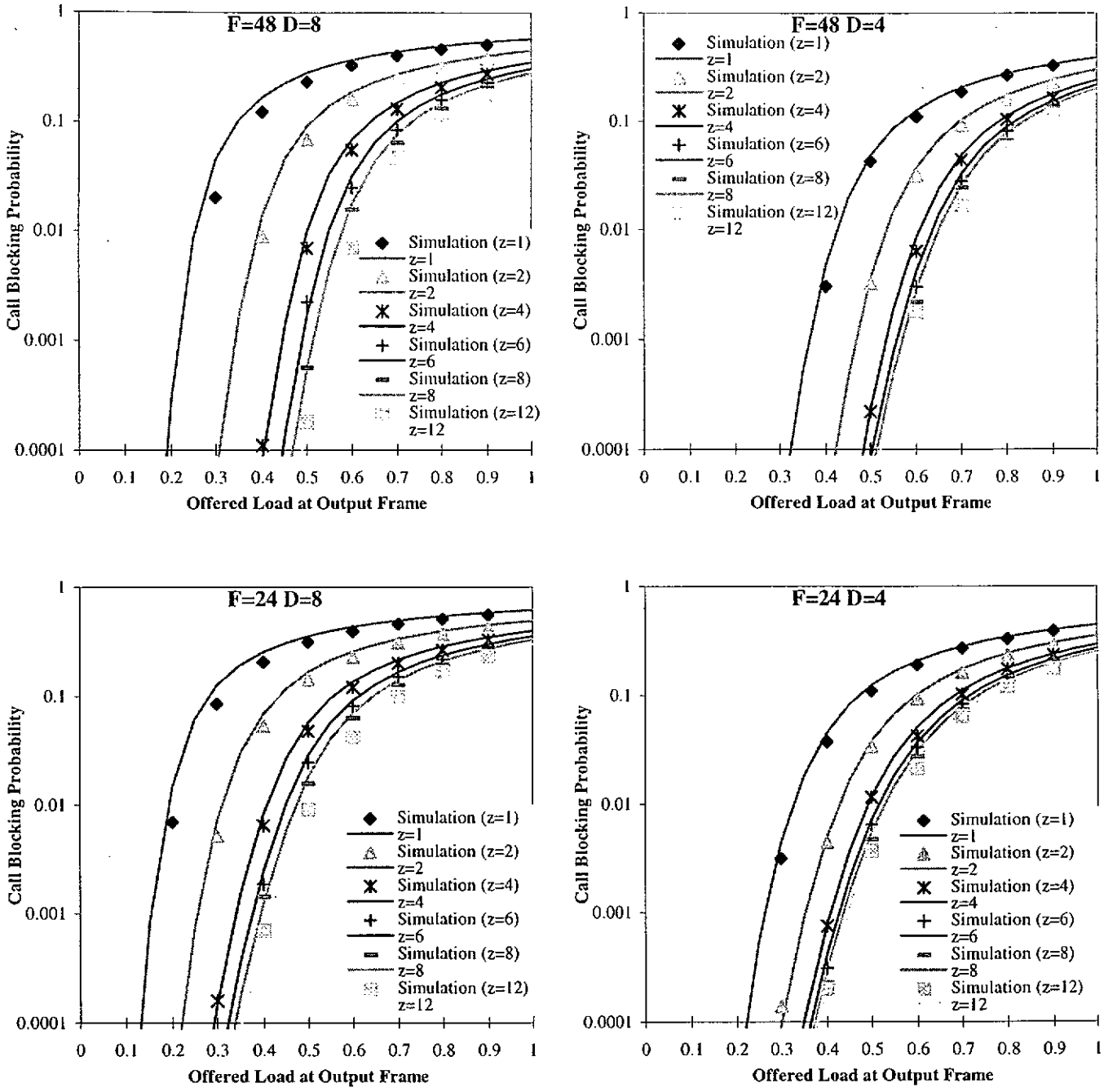


Figure B.1 Comparison of the call blocking probability between simulation results and analytical results for $N = 64$.

In figure B.1, we also observe that the gap between the analytical results and the corresponding simulation results decreases as z increases. In order to clarify this relation, we predict the performance of a demux-mux switch with smaller size $N = 32$ and compare it to the analytical ones in figure B.2. Simulating switch

performance with smaller size (N) is aimed to give us a more clear observation as according to the previous results, the deviation between these results in this case is always larger than that in figure B.1.

The resultant plots show that the deviation between different results decreases as z increases, which behaves in the same ways as that observed before. This phenomenon could be accounted by the fact that as z increases the number of sub-frames per port increases too, which provides more rooms to connect a call and then more alternatives for distributing busy slots in each sub-frame group. Thus, the interdependence amongst different sub-frame groups would be diluted and hence the deviation at these values of z may be reduced. On the other hand, the decrement in the deviation also could be explained by the convergence of switch performance to the lower bound at large values of z, which inherently reduces these deviations since the computer predicted results are also bounded by the same lower bound.

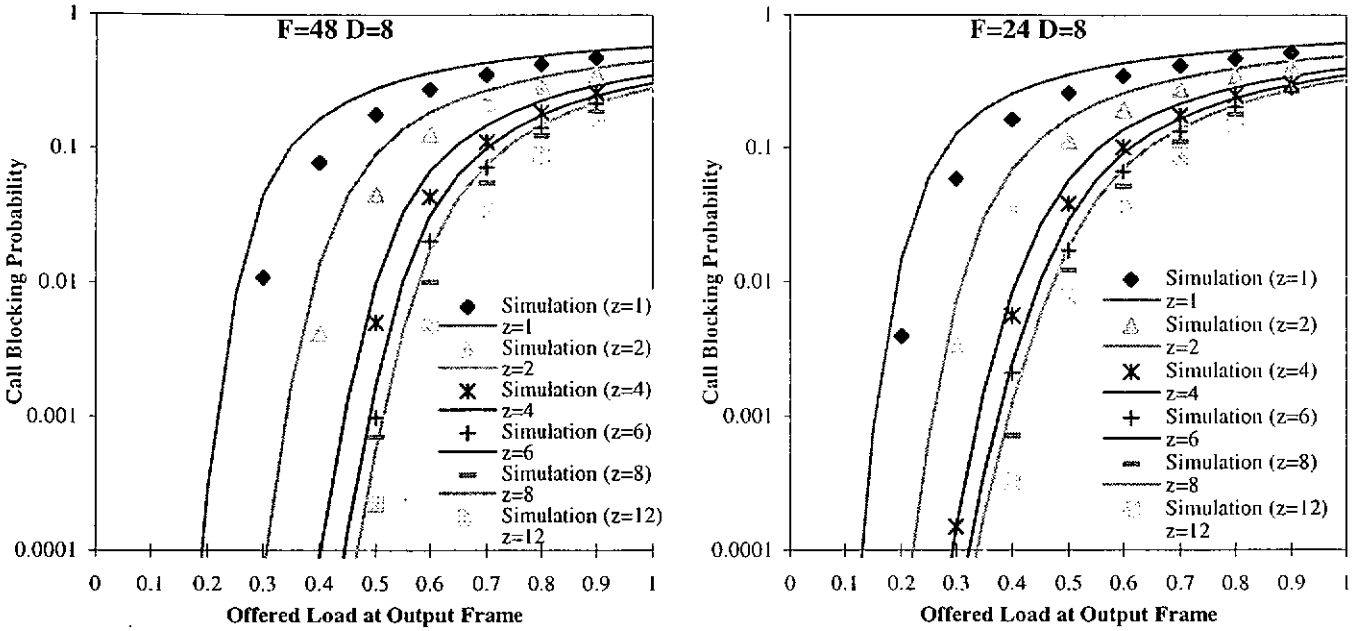


Figure B.2 Comparison of the call blocking probability between simulation results and analytical results for $N = 32$.

Appendix C

To analyze the call blocking probability of multistage switch proposed in Chapter 4, we have to derive the probability distribution of the overlap in close windows between two given frames. The probability function is given by the following Lemma. Note that the meaning of the denotations and terms used here are referred to that in Chapter 4.

Lemma 4.1: Considering a frame with distribution vector $s=(n_1,n_2,\ldots n_F)$ has $l'\ (0\le l'\le l)$ close windows overlap the close window region of size $l\ (l<w)$ in the other frame. The probability distribution of overlapping close windows $\delta(l')$ is given by

$$\delta(l'\backslash s)=\begin{cases}\dfrac{\sum_{i=1}^{w-1}n_i\cdot i+\sum_{i=w}^{F}n_i\cdot(w-l+1)}{F} & l'=0,\\ \dfrac{n_{w+l'}\cdot(l-l'-1)+\sum_{i\ge w+l'}^{F}n_i\cdot 2}{F} & 0<l'<l,\\ \dfrac{\sum_{i\ge l+w}^{F}n_i\cdot(i-w-l+1)}{F} & l'=l.\end{cases} \tag{4.9}$$

Proof: The probability distribution is derived by counting the number of ways to distribute a close window region of size l $(l < w)$ in A to have such an overlap. For clarification, the proof is divided into three cases according to the value of l'. Figure C.1 shows the conceptual representation of the cases.

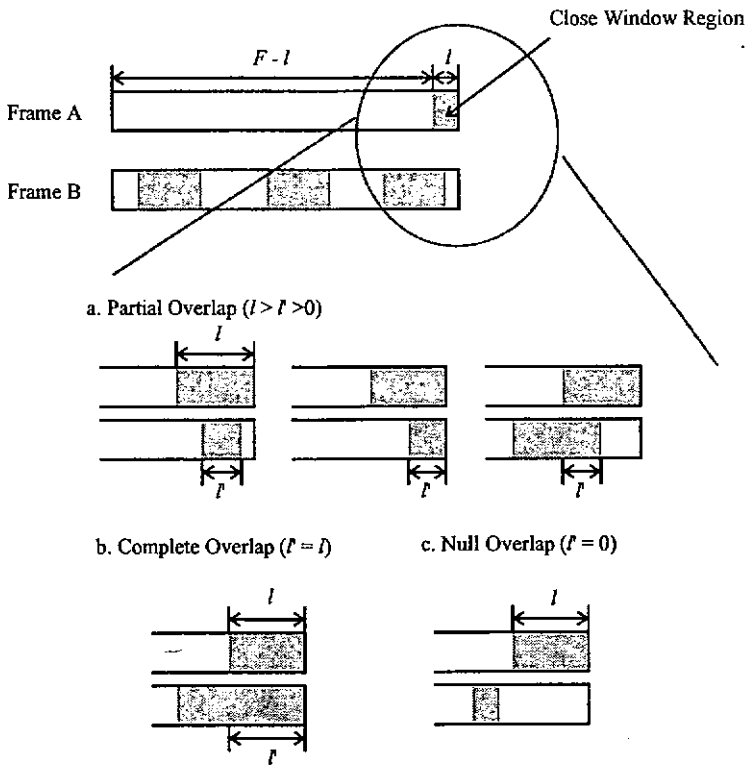


Figure C.1 The ways of the close windows overlap between frame A and B.

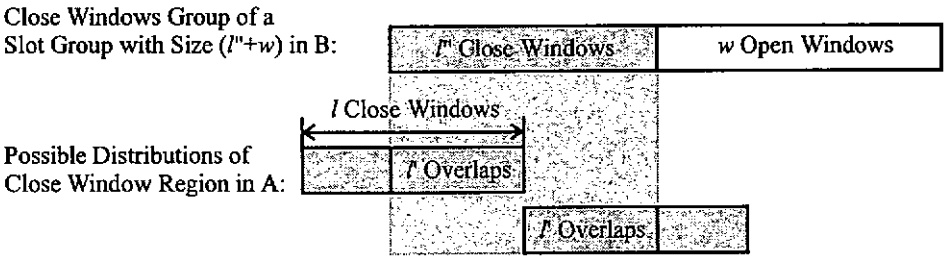
In the case of $0 < l' < l$, there are l' close windows in A overlap a close window group (of size $l'' \geq l'$) in B. The probability distribution of l' overlaps is calculated by counting the number of ways that the close window region in A can distribute with respect to each of these close window groups to achieve this overlap. Considering a slot group of size $i > w + l'$ has $i - w$ close windows packed together. To have l' overlaps, the number of possible distributions of the close window region with respect

to this slot group is two (see figure C.2(a)). Particularly, for a slot group of size $w+l'$, there are totally $(l-l'+1)$ ways to have such extent of overlap (see figure C.2(b)). Thus, the probability distribution of l' $(0<l'<l)$ overlapping close windows is

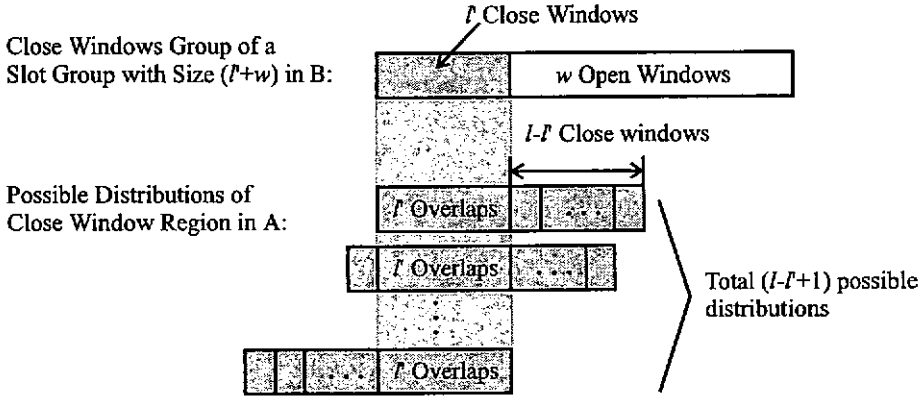
$$\delta(l'\backslash s) = \frac{\begin{pmatrix}\text{No. of ways to distribute the close window region to}\\ \text{have } l' \text{ overlapping close windows in each slot group}\end{pmatrix}}{\begin{pmatrix}\text{Total no. of possible distributions of the close}\\ \text{window region in a frame}\end{pmatrix}} \qquad 0<l'<l$$

$$= \frac{n_{w+l'}\cdot(l-l'+1)+\sum_{i>w+l'}^{F} n_i \cdot 2}{F}$$

$$= \frac{n_{w+l'}\cdot(l-l'-1)+\sum_{i\geq w+l'}^{F} n_i \cdot 2}{F} \qquad \text{(C.1)}$$



(a)



(b)

Figure C.2 Distributions of close window region in A with respect to different slot group in B.

Similarly, the probability of having a null overlap $(l'=0)$ is calculated by finding how many ways that the close window region in A can distribute within the open window regions in B. Considering a close window group (if any) in each slot group has w open windows in front of it. There are $(w-l+1)$ possible distributions of a close window region with size l in the positions of these w open windows without overlap the corresponding close windows. Slot groups with size $i < w$ contain no close window. Their occurrence always enlarges the size of the following open window regions and increases the number of ways to distribute the close window region within these open window groups. The additional number of distributions is equal to their size (i slots). Figure C.3 show the distribution of close window region in different slot groups. The probability of a null overlap is given by the following equation.

$$\delta(0 \setminus s) = \frac{\sum_{i=1}^{w-1} n_i \cdot i + \sum_{i=w}^{F} n_i \cdot (w-l+1)}{F} \qquad \text{(C.2)}$$

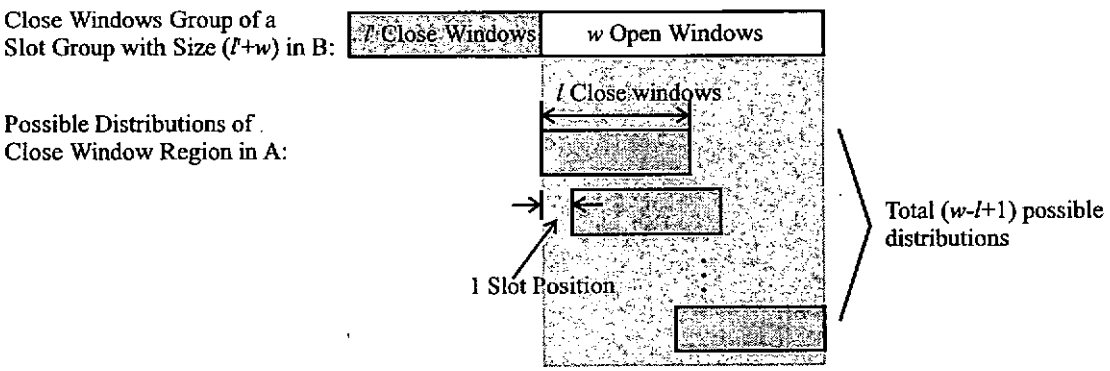


Figure C.3(a) Distribution of close window region within w open windows

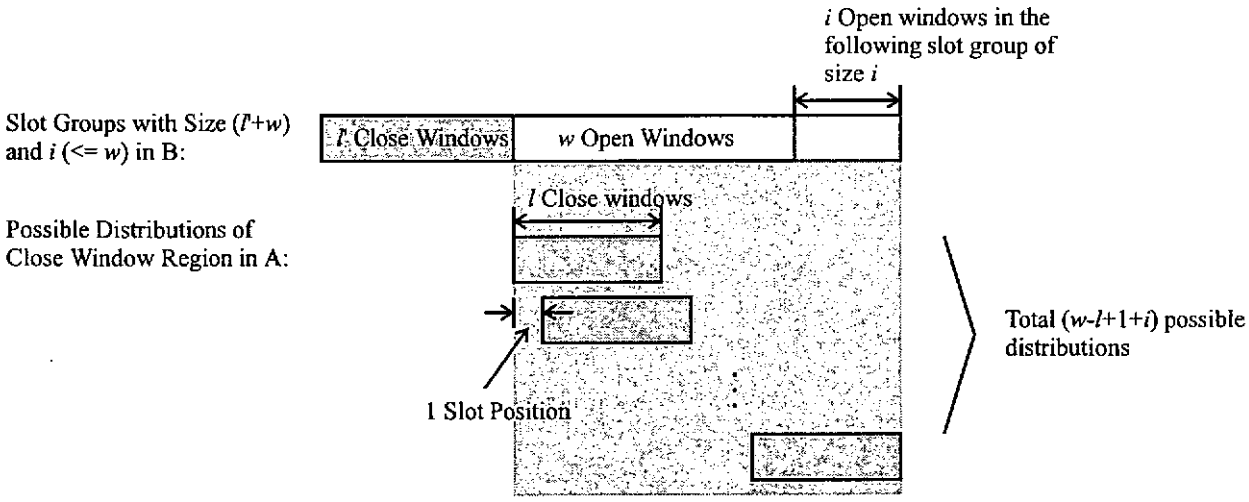


Figure C.3(b) Distribution of close window region in an open window group which is enlarged by the following slot group of size $i \leq w$.

Finally, the probability of a full overlap $(l' = l)$ is calculated by counting the number of ways to distribute the close window region in each close window group with size $l'' \geq l$ in B. The probability function is derived in a way similar to that demonstrated in the previous cases and is given as follows

$$\delta(l \setminus s) = \frac{\sum_{i \geq w+l}^{F} n_i \cdot (i - w - l + 1)}{F} \qquad \text{(C.3)}$$

■

Appendix D

In this appendix, a comparison of average call blocking probabilities predicted by numerical analysis and computer simulations for switch size $N = 64$ and 32 is made and shown in figure D.1 and D.2.

In the plots, we observed that the larger the ratio D/N the greater the difference between the simulation results and the analytical results. It is due to the interdependence of the busy slots amongst the input frames and the output frames, which has been briefly discussed in [ROSE 87] and [KIM 92a]. In general, this inter-frame dependence increases the occurrence of common idle slots so that the average call blocking probability predicted by computer simulations are always smaller than that obtained in the analysis.

Also, it has shown that the result difference increases as the window size w increases regardless of the values of F, D and N. It could be accounted by the reduction of the inter-frame dependence, which is due to the additional freedom granted by the window scheduling on the assignment of conflict-free time slots for each call. This eliminates the constraint imposed by the traditional one-shot scheduling which is the main cause for this interdependence. Therefore, the larger the window size w the greater the freedom on call scheduling and hence the smaller the inter-frame dependence resulted. Besides, according to the discussion in Appendix B, the reduction in difference could also be accounted by the same lower bound (theoretical limit) faced by these sets of results, where both of them converge to it as the window size increases.

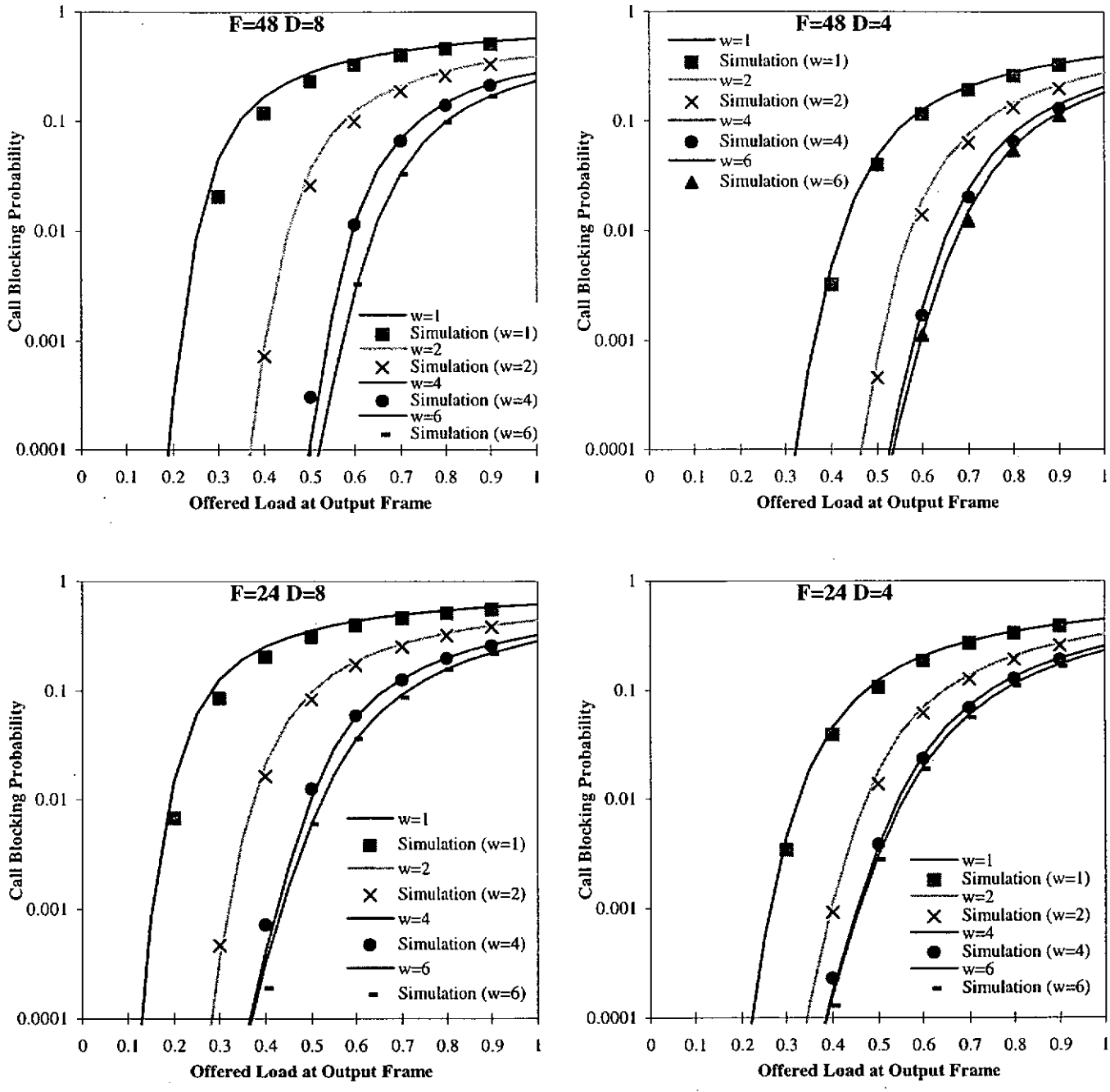


Figure D.1 Comparison of the call blocking probability between simulation results and analytical results for $N = 64$.

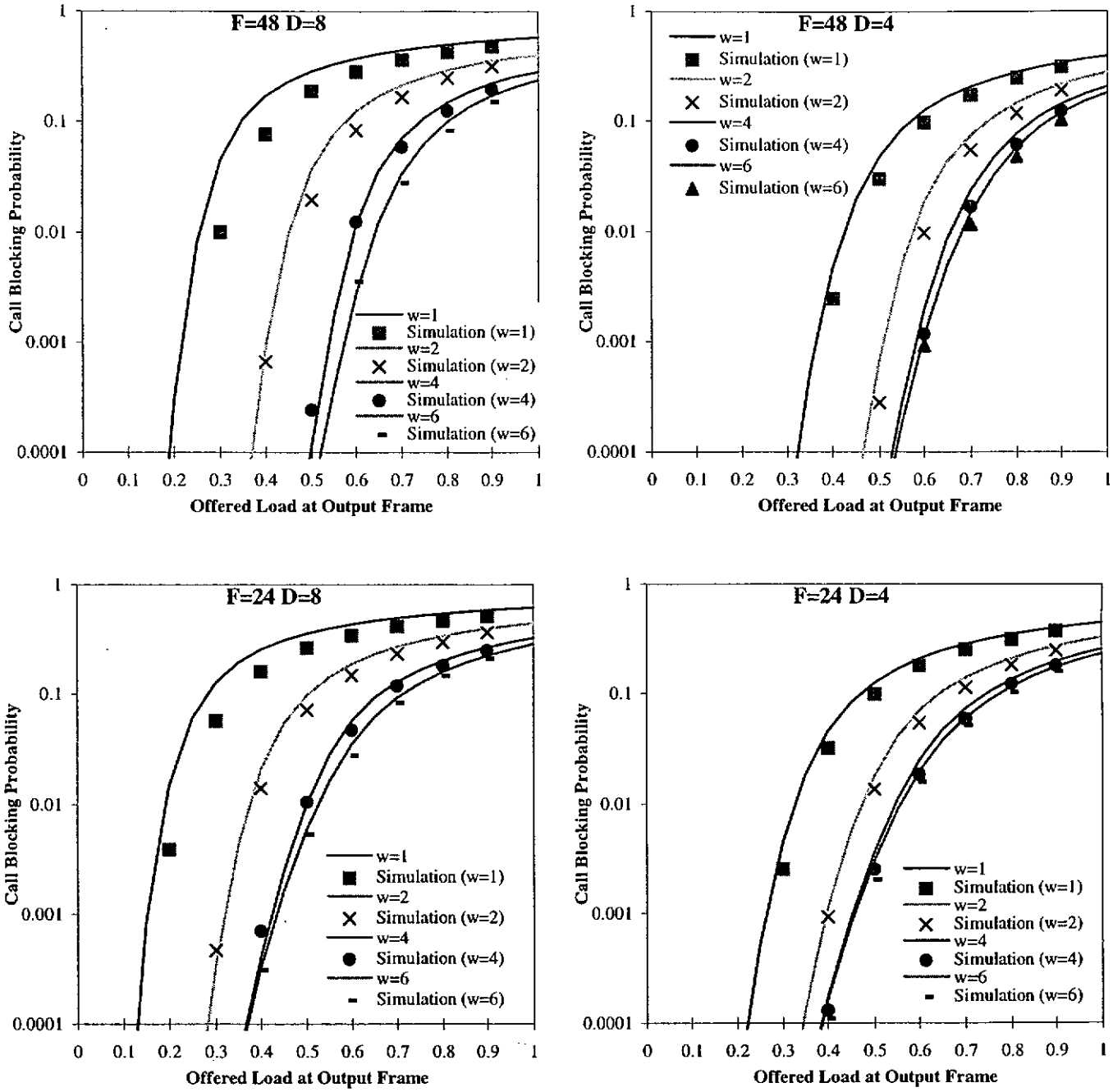


Figure D.2 Comparison of the call blocking probability between simulation results and analytical results for $N = 32$.

Bibliography

[ACAM 78] A. S. Acampora and B. R. Davis, "Efficient utilization of satellite transponders via time-division multibeam scanning," *Bell Syst. Tech. J.*, vol. 57, no. 8, pp. 2901-2914, 1978.

[BART 84] S. M. Barta and M. I. Honig, "Analysis of a demand assignment TDMA blocking system," *AT&T Bell Lab. Tech. J.*, vol. 63, no. 1, pp. 89-114, 1984.

[BENE 65] V. E. Benes, *Mathematical Theory of Connecting Networks and Telephone Traffic*. New York: Academic, 1965.

[BONG 81a] G. Bongiovanni, D. Coppersmith, and C. K. Wong, "An optimum time slot assignment algorithm for an SS/TDMA system with variable number of transponders," *IEEE Trans. Communications*, vol. COM-29, pp. 721-726, May 1981.

[BONG 81b] G. Bongiovanni, D. T. Tang, and C. K. Wong, "A general multibeam satellite switching algorithm," *IEEE Trans. Communications*, vol. COM-29, pp. 1025-1036, July 1981.

[BONU 89] M. A. Bonuccelli, "A fast time slot assignment algorithm for TDM hierarchical switching systems," *IEEE Trans. Communications*, vol. COM-37, pp. 870-874, Aug. 1989.

[BUTN 96] S. E. Butner and R. Chivukula, "On the limits of electronic ATM switching," *IEEE Network Magazine*, pp. 26-31, Nov. 1996.

[CALI 82] P. Calingaert, *Operating System Elements*. Englewood Cliffs: Prentice-Hall, 1982

[CHAL 90] S. Chalasani and A. Varma, "An improved time-slot assignment algorithm for TDM hierarchical switching systems," in *Proc. 4th Int. Conf. Data Commun. Syst. Perform.*, Barcelona, Spain, June 1990.

[CHAL 93] S. Chalasani and A. Varma, "Fast parallel time-slot assignment algorithm for TDM switching systems," *IEEE Trans. Communications*, vol. 41, pp. 1736-1747, Nov. 1993.

[CHAL 94] S. Chalasani and A. Varma, "Efficient time-slot assignment algorithms for SS/TDMA systems with variable-bandwidth beams," *IEEE Trans. Communications*, vol. 42, no. 2/3/4, pp. 1359-1370, Feb./Mar./Apr. 1994.

[CHEN 94] W. T. Chen, P. R. Sheu, and J. H. Yu, "Time slot assignment in TDM multicast switching systems," *IEEE Trans. Commun.*, vol. 42, no. 1, Jan. 1994.

[CHEN 95] W. T. Chen and H. J. Liu, "An adaptive scheduling algorithm for TDM switching systems," *IEEE Trans. Communications*, vol. 43, no. 2/3/4, Feb./Mar./Apr. 1995.

[CLOS 53] C. Clos, "A study of non-blocking switching networks," *Bell Syst. Tech. J.*, March 1953.

[DELO 94] D. Deloddere, W. Verbiest, and H. Verhille, "Interactive video on demand," *IEEE Communications Magazine*, vol. 32, no. 5, pp. 82-88, May 1994.

[ERRA 94] A. Erramilli, E. H. Lipper, and J. L. Wang, "Some performance considerations for mass market broadband services," *IEEE Proc. of the 1st International Workshop on Community Networking Integrated Multimedia Service to the Home*, NY, USA, pp. 109-116, 1994.

[FUNA 94] N. Funabiki and Y. Takifuji, "A parallel algorithm for time-slot assignment problems in TDM hierarchical switching systems," *IEEE Trans. Communications*, vol. 42, no. 10, pp. 2890-2898, Oct. 1994.

[GERS 96] O. Gerstel, "On the future of wavelength routing networks," *IEEE Network Magazine*, pp.14-20, Nov. 1996.

[GIPS 96] T. Gipser and M. S. Kao, "An all-optical network architecture," *J. Lightwave Technol.*, vol. 14, no. 5, May 1996.

[GOPA 82a] I. S. Gopal, G. Bongiovanni, M. A. Bonuccelli, D. T. Tang, and C. K. Wong, "An optimal switching algorithm for multibeam satellite systems with variable bandwidth beams," *IEEE Trans. Communications*, vol. COM-30, pp. 2475-2481, Nov. 1982.

[GOPA 82b] I. Gopal, D. Coppersmith, and C. K. Wong, "Minimizing packet waiting time in a multibeam satellite system," *IEEE Trans. Commun.*, vol. 30, pp. 305-316, 1982.

[HOPC 73] J. E. Hopcroft and R. M. Karp, "An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs," *SIAM J. Comput.*, vol. 2, pp. 225-231, Dec. 1973

[HOU 96] T. C. Hou, "Hierarchical Path Hunt in a Broadband Multirate Circuit Switching Fabric," *IEEE J. Select. Areas Commun.*, vol. 14, no. 2, pp.386-398, Feb. 1996.

[HUAN 84] A. Huang and S. Knauer, "Starlite: A wideband digital switch," in *Proc. IEEE GLOBECOM '84*, pp. 121-125.

[INUK 79] T. Inukai, "An efficient SS/TDMA time slot assignment algorithm," *IEEE Trans. Communications*, vol. COM-27, pp. 1449-1455, Oct. 1979.

[JOEL 79] A. E. Joel, Jr., "Digital switching-how it has developed," *IEEE Trans. Communications*, Vol. COM-27, pp. 948-959, July 1979.

[KIM 92a] C. K. Kim and T. T. Lee, "Call scheduling algorithms in a multicast switch," *IEEE Trans. Commun.*, vol. 40, no. 3, pp.625-635, Mar. 1992.

[KIM 92b] C. K. Kim, "Performance analysis of a duplex multicast switch," *IEEE Trans. Commun.*, vol. 40, no. 10, pp.1615-1624, Oct. 1992.

[KIM 96] C. K. Kim, "Blocking performance of heterogeneous traffic in multirate multicast switch," *IEEE J. Select. Areas Commun.*, vol. 14, no. 2, pp.374-385, Feb. 1996.

[KLEI 75] L. Kleinrock, "Queueing systems," *A Wiley-Interscience publication*, vol.1, pp. 105-106, 1975.

[LAWL 76] E. L. Lawler, *Combinatorial Optimization: Networks and Matroids*. New York: Holt, Rinehart and Winston, 1976.

[LEE 88] T. T. Lee, "Nonblocking copy networks for multicast packet switching," *IEEE J. Select. Areas Commun.*, vol. SAC-6, pp. 1455-1467, Dec. 1988.

[LEWA 83] J. L. Lewandowski, J. W. S. Liu, and C. L. Liu, "SS/TDMA time slot assignment with restricted switching modes," *IEEE Trans. Communications*, vol. COM-31, pp. 149-154, Jan. 1983.

[LI 91] S. Q. Li, "Performance of trunk grouping in packet switch design," *Proc. IEEE INFOCOM'91*, Miami, FL, pp. 688-693, Apr. 1991.

[LIEW 89] S. C. Liew, "Comments on 'fundamental conditions governing TDM switching assignments in terrestrial and satellite networks'," *IEEE Trans. Communications*, vol. 37, pp. 187-189, Feb. 1989.

[LUND 48] K. Lundkvist, "Method of computing the grade of service in a selection stage composed of primary and secondary switches," *Ericsson Rev.*, no. 1, pp. 11-17, 1948.

[NARA 94] N. Naraghi-Pour, M. Hegde, and S. Suresh, "Scheduling multi-rate traffic in a time-multiplexed switch," in *Proc. IEEE International Symposium on Information Theorey*, pp. 406, 1994.

[PRYC 91] M. De Prycker and M. De Somer, "Performance of a service independent switching network with distributed control," *IEEE J. Select. Areas Commun.*, vol. 5, no. 8, May 1991.

[RAMA 84] R. Ramaswamy and P. Dhar, "Comments on "an efficient SS/TDMA time slot assignment algorithm"," *IEEE Trans. Communications*, vol COM-32, pp. 1061-1065, Sept. 1984.

[ROSE 87] C. Rose and M. G. Hluchyj, "The performance of random and optimal scheduling in a time-multiplex switch," *IEEE Trans. Commun.*, vol. COM-35, pp. 813-817, Aug. 1987.

[ROSE 89] C. Rose, "Rapid optimal scheduling for time-multiplex switches using a cellular automaton," *IEEE Trans. Communications*, vol. COM-37, pp. 500-509, May 1989.

[SCAR 83] T. Scarcella and R. V. Abbott, "Orbital efficiency through satellite digital switching," *Communications*, pp. 38-46, May 1983.

[SYSK 60] R. Syski, *Introduction to Congestion Theory in Telephone Systems.* London: Oliver and Boyd, 1960, ch. 8.

[TURN 86] J. S. Turner, "New directions in communications," in *Proc. Int. Zurich Seminar on Digital Comm.*, Zurich, Switzerland, Mar. 1986.

[TURN 88] J. S. Turner, "Design of a broadcast packet switching network," *IEEE Trans. Commun.*, vol. 36, pp. 734-743, June 1988.

[VARM 92] A. Varma and S. Chalasani, "An incremental algorithm for TDM switching assignments in satellite and terrestrial networks," *IEEE J. Select. Areas Commun.*, vol. 10, no. 2, pp. 364-377, Feb. 1992.