Submitted by

Fahad A. Al-Zahrani

Department of Electrical and Computer Engineering

In partial fulfillment of the requirements

for the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Summer, 2005

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

COLORADO STATE UNIVERSITY

July 7, 2005

WE HEREBY RECOMMEND THAT THE DISSERTATION PREPARED UNDER OUR SUPERVISION BY FAHAD A. AL-ZAHRANI ENTITLED MODELING, PERFORMANCE ANALYSIS, AND SURVIVABILITY EVALUATION OF MULTI-FIBER WDM OPTICAL NETWORKS BE ACCEPTED AS FULFILLING IN PART REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY.

Committee on Graduate Work

M. M. Siddiqui

Adviser

Department Head

ABSTRACT OF DISSERTATION

MODELING, PERFORMANCE ANALYSIS, AND SURVIVABILITY EVALUATION OF MULTI-FIBER WDM OPTICAL NETWORKS

Performance goals of future high capacity optical networks may be achieved using a combination of features such as additional wavelengths, parallel fibers, converters and complex switch architectures. The trade-offs involving the use of multi-fiber multi-hop networks such as the number of fibers, number of wavelengths, conversion options, and different switch configurations are examined, and their impact on end-to-end blocking and throughput performance is evaluated. Models relating network parameters to end-to-end performance of circuit-switched all-optical networks are developed. The performance gain due to the use of multiple fibers with or without conversion is shown to be superior to the single fiber case, when the total number of wavelengths is the same, for traffic that traverses multiple hops between source and destination nodes.

A performance model is presented for an optical packet switch architecture in which the wavelength converters are shared per output link and each output link consists of multiple fibers. Symmetry of the switch is exploited to derive the packet loss probability for the case where traffic is destined to different output ports with equal probability. The performance is evaluated by means of an analytical model and confirmed by simulations under different switch parameter configurations. The improvement of the packet loss probability of the switch with the number of wavelength converters is characterized. The performance of the switch is also evaluated when multiple fibers are utilized at the output of the switch.

The improvement of performance due to a single wavelength conversion device in optical packet switch (OPS) as well as one extra fiber with respect to the total number of wavelengths is investigated. In this research, we show that the performance of the optical packet switch could be enhanced with less cost overhead by utilizing the converter and multi-fiber link tradeoffs. Moreover, the enhancements in performance of the switch under different packet contention resolution techniques (wavelength conversion, multiple fibers) are compared. Further, the optical packet switch performance limits are identified. The results show that synchronous switches equipped with full conversion would have the least conversion utilization rate, indicating that the use of a switch with fewer converters, i.e., partial conversion, would offer better switch resources utilization and comparable packet loss performance. The number of converters that are required to achieve the best performance possible decreases when multiple fibers are utilized.

Both availability and performance degradation of a system in the presence of failure are integral components of network survivability evaluation. Therefore, a composite model is presented for survivability of multi-fiber WDM networks that includes system availability analysis to discover the cost due to system downtime, and system failure impact analysis to discover the transient performance degradation when failure occurs. The results show that network design can exploit availability of multiple fibers to enhance both performance and survivability.

Fahad A. Al-Zahrani
Department of Electrical and Computer Engineering
Colorado State University
Fort Collins, Colorado 80523
Summer, 2005

ACKNOWLEDGEMENTS

Foremost, I would like to thank Allah for his bounties and blessing. He blessed me with loving parents, brothers, sisters, and friends. I also thank him for the guidance and ability he provided me to finish this research. Surely, without his mercy and blessing this work would not have been possible.

I also would like to take this opportunity to express my sincere and special thanks to my parents for their endless concern, love, prayers, and support. Thanks go to my bothers and sisters for continuous encouragement. Special thanks go to my brother Khalid for everything he did for me while I'm on my Ph.D. journey. I sometimes felt guilty for passing over my responsibilities to him while he never complained but provided me with encouragement. Thanks go to my wife for always being there but never asking why it was taking so long and also to my sons Ahmed and Bashaar, who make everything worthwhile.

Sincere gratitude to my advisor Professor Anura Jayasumana for providing invaluable insights, timely encouragement as well as guidance, balanced by the freedom to express myself through this work. He showed me different ways to approach research problems and the importance of being persistent to accomplish any goal. Again, I thank him from the bottom of my heart. Finally, I would like to thank all my Ph.D. committee members for their valuable time they spent reading and evaluating my research work.

DEDICATION

To my parents

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

The increasing popularity of the Internet and its audio and video applications in the past few years have contributed to increase the bandwidth requirements of telecommunications networks. This growth in Internet traffic is anticipated to increase exponentially as shown by many statistics. Advances in all-optical networks (AON) using Wavelength Division Multiplexing (WDM) lead to feasible solutions for providing a faster networking infrastructure that can meet the explosive growth of Internet traffic.

In this chapter a brief introduction to the all-optical network will be provided. Then the importance of this research work and the contributions that have been achieved in this research will be presented briefly.

1.1 All-Optical Network (AON)

The high bandwidth offered by the optical fiber and the ability to switch the data stream in the optical domain make the all-optical network (AON) an attractive solution for the backbone transport network. AON nodes are equipped with an optical cross-connect (OXC) that is able to switch the signals from input ports to output ports without optical-electrical-optical (O-E-O) conversion, which in turn eliminates the bottleneck speed of the electronic devices. Wavelength Division Multiplexing (WDM) technology adds another advantage to AON and makes it the best candidate for the next generation backbone transport network by exploiting multiple wavelengths to transmit on the same optical fiber, which introduces a means of effectively utilizing the fiber bandwidth. Thus, the capacity of the network increases with the number of available wavelengths. This makes WDM a very attractive multiplexing technology for carriers, who can rely on it to provide the necessary bandwidth in order to satisfy the ever-increasing demand for bandwidth. According to the way the OXCs can be configured to switch the optical traffic, the optical network can be classified as follows:

1. Wavelength-routed network (circuit switched)
2. Packet switched
3. Burst switched

As for this research, the wavelength-routed and optical packet switched networks will be analyzed and investigated in detail in this dissertation.

1.2 Wavelength-routed Networks

The wavelength-routed network consists of a set of optical switching nodes connected by point-to-point fiber links. A connection circuit (lightpath) has to be established between the source and destination nodes first before starting data transmission in the wavelength-routed network. The connection circuit is released when no longer required. The need to find the same free wavelength on all hops in order to establish the lightpath is called the *wavelength continuity constraint.* Due to the wavelength continuity constraint, two different demands cannot use lightpaths that share a fiber if they have the same wavelength. This constraint degrades the wavelength-routed network performance because of lightpath requests blocking. Lightpath requests are blocked either if all wavelengths are used on a hop or if there is no common wavelength on all hops. The performance degradation in the wavelength-routed network imposed by the wavelength continuity constraint will be reduced if a lightpath is allowed to be set up on different wavelengths.

Extensive studies have been made on the single-fiber wavelength-routed network where pairs of nodes are connected using a single fiber link. Different probabilistic models dealing with performance issues of single-fiber optical network have been presented in these studies [3, 7, 9]. These models studied the effect of various optical network parameters such as number of wavelengths used, size of the optical switches, the average length of the optical paths, the availability of wavelength converters, etc., on the blocking probability and utilization of the network. The single-fiber optical network limitations such as poor performance in the absence of wavelength conversion, limited network scalability, and configuration flexibility raise a need for a new implementation of the optical network that overcomes the single-fiber limitations without any extra overhead cost. The cost of wavelength converters is expected to remain high in the near future, so wavelength conversion is not a feasible solution for the single-fiber optical network limitations. Also, trying to add more wavelengths into a single fiber using WDM fine spacing techniques reduces wavelength stability by increasing wavelength cross-talk. A fiber cable consists of multiple strands. Therefore, an alternative approach for increasing the number of wavelengths is to light multiple fibers, each using the same set of wavelengths.

1.3 Multi-Fiber Optical Networks

In the multi-fiber WDM networks, each link consists of multiple fibers. Using multiple fibers on each link in the optical networks will overcome the single-fiber optical network limitations in terms of network scalability and configuration flexibility. Also, it enhances the blocking performance of the network due to the reduction of the wavelength continuity constraint effect even in the absence of the wavelength conversion resources. Recently, this optical network environment has become a hot field of research in the area of the optical network [67]. In this research, the performance enhancement that results from employing different multi-fiber optical network configurations as well as different wavelength conversion options is investigated and evaluated.

The significance of using multiple fibers can be seen when considering the number of useable channels. By using multiple fibers in the conventional C-band window (1535-1565 nm), the designer has the option of using $w = W \cdot F$ channels in each of F fibers instead of populating a fiber with W channels as a result of traffic demand scaling. This, in addition to overcoming the capacity exhaustion problem in the C-band window, provides significant performance advantages as well. Moreover, employing multiple fibers in multi-hop wavelength-routed networks will reduce the end-to-end blocking probability by reusing the same set of wavelengths in multiple fibers and by increasing the availability of the end-to-end paths.

Increasing the resources availability in the multi-fiber environment by reusing the same set of wavelengths enhances the optical network throughput, which adds more configuration flexibility and scalability to the optical network infrastructure. This configuration flexibility comes from the reduction of the wavelength continuity constraint effects due to the ability of the network to set up multiple lightpaths on the same wavelength in different fibers on the same hop. The use of multiple fibers compensates for the utilization degradation caused by the increase in the hop count that a path traverses.

However, even though having multiple fibers between nodes on one physical link enhances the all-optical network performance, there is a drawback in its survivability because in the case of link failure, all the fibers between the nodes will fail simultaneously, which results in massive traffic loss. Therefore, fiber assignment in an all-optical network to increase the survivability of the multi-fiber optical network configuration is an important research issue.

1.4 Survivability of Optical Networks

An all-optical network based on WDM technology becomes the technology of choice for the transport network due to its massive capacity, reliability, cost and scalability. This network can potentially transfer hundreds of terabits-per-second of data on each fiber link in the network. However, the interruption of service for even short periods of time may have catastrophic consequences. Thus, network survivability and service continuity have become important research and design issues for any all-optical network. In general, network survivability is defined as the ability of a network to maintain or restore an acceptable level of performance in the event of failures. A single failure in the all-optical network would cause severe service loss. Link and node failure are the basic failures that have to be considered in the all-optical network investigations. The link failure occurs because of cable cuts; node failure is due to equipment failure at network nodes.

Also, wavelength failure is possible in all-optical networks. This failure occurs due to the failure of a transmitter or a receiver device operating on that wavelength. Due to the high reliability of optical devices, the cable cut that causes link failure is the common failure in the all-optical networks. The need for an accurate and simple survivability metric measure for the optical network motivates us to investigate the survivability issues under different scenarios.

Network survivability should include system availability analysis to find out the cost due to system downtime, and system failure impact analysis to discover the transient performance degradation when failure occurs [31]. We will present a hierarchical survivability model for multi-fiber WDM networks that consists of availability and transient performance analysis.

1.5 Optical Packet Switched Networks

Running IP traffic on the top wavelength-routed networks, which are circuit-switched in nature, will result in poor utilization of the optical network bandwidth, since the data transmission time of most Internet applications is smaller than the duration of the light-path setup time. To overcome this inherently poor utilization of the WDM bandwidth in the wavelength-routed network, optical packet switching (OPS), which allows fast allocation of wavelengths in an on-demand fashion, is introduced. The optical packet network consists of optical packet switches interconnected with fibers. The Internet traffic is switched by optical packet switches on a packet-by-packet basis and this occurs entirely in the optical domain. The optical network bandwidth is utilized more efficiently in this optical network implementation since the wavelength can be utilized by different traffic initiated from different sources. However, the OPS suffers from a packets contention problem that arises when two or more of the incoming packets on the same wavelength intend to leave the switch through the same output port, which results in packet loss and poor network performance.

Using the multi-fiber connection in an optical network environment will enhance network performance in general as well as the individual OPS switches' performance due to utilizing the space dimension of the same wavelength on different fibers to resolve packets contention. So, multi-fiber implementation of the OPS will overcome the limitations of the packet contention problem. The higher bandwidth utilization of the optical link in the multi-fiber environment significantly reduces the number of converters needed inside the switching node. In this research, the worthiness of using multiple fibers on the output of the OPS in terms of the wavelengths is also investigated. The worthiness results show that adding one additional fiber to the output of the OPS is worth more than adding a wavelength conversion device, which means that the OPS performance could achieve the best performance possible with the least overhead cost.

1.6 Dissertation Outline

Different optical network configurations, architecture and a research review will be presented and discussed in detail in Chapter 2. Motivations, goals and contributions of this research work will be explained in Chapter 3. In Chapter 4, an analytical model for the single-fiber wavelength-routed network will be outlined to show the performance gain that can be achieved by using multiple fibers between nodes. A general analytical model that can be used to study the wavelength-routed network performance with or without wavelength conversion will be presented in this chapter. To overcome the poor utilization of the bandwidth of the parallel fibers in the wavelength-routed network, optical packet switching-technique will be investigated. The gain in the network performance of this optical network architecture will be evaluated in Chapter 5 through analytical and simulation models. The single and multiple fibers environments will be considered in these models for the most effective wavelength conversion sharing mechanism. In Chapter 6, optical network survivability enhancement when a multi-fiber environment is used will be considered and evaluated. A hierarchical optical network survivability will be developed and presented in this chapter. The different optical network simulators developed to verify the analytical analysis in this research are discussed in detail in Chapter 7. Documented versions of these simulators are presented in the appendices of this dissertation. Finally, the dissertation conclusion and future work are provided in Chapter 8.

Chapter 2

INTRODUCTION TO WDM OPTICAL NETWORK

2.1 Introduction

In the optical networks, the data streams that flow between a source-destination pair remain in the optical domain throughout their paths except at the end nodes; this provides data transparency and protocols independent at the physical layer of the optical network. The Wavelength Division Multiplexing (WDM) has been widely adopted as the technology of choice for increasing the transmission capacity of carrier networks. WDM is a technology where the optical spectrum is divided into a number of non-overlapping wavelengths. In WDM each wavelength is considered as a single communication channel operating at its peak electronic speed. The multiplexing technology provided by WDM allows these non-overlapping wavelengths to be multiplexed on a single fiber, which introduces the means of effectively utilizing the fiber bandwidth. So, the capacity of the network increases with the number of available wavelengths. This makes WDM a very attractive multiplexing technology for carriers who can rely on it to provide the necessary bandwidth in order to satisfy the ever increasing bandwidth demand.

Optical node has two main planes, the physical plane, which consists of switching elements and conversion devices if they exist, and the control plane, which is used to configure and manage the physical plane inside the node. This plane is mostly implemented in software. The switching element inside the optical node is called optical cross-connect (OXC). The optical cross-connect (OXC) has to operate at very high speed and it should be able to switch the individual wavelengths without converting the optical signal to electrical in order to be able to efficiently utilize the massive bandwidth of the fiber optic cable.

This chapter gives an overview of the previous works that have been done on performance evaluation of the all-optical network. An overview of different routing and wavelength assignments algorithms are presented in Section 2.2. Section 2.3 gives an overview of previous research on the single-fiber wavelength-routed network and its performance. The previous researches on multi-fiber wavelength-routed network environment are described in Section 2.4. An overview on the research works have been done on the performance evaluation of the optical packet switching network is presented in Section 2.5. Finally, the previous research on the optical network survivability will be presented in Section 2.8.

2.2 Routing and Wavelength Assignment Algorithms

In the wavelength-routed network employing WDM technology, a connection between a specific source-destination pair has to be established on a continuous wavelength in the absence of wavelength conversion [18]. The connection between the source-destination pair is called a lightpath. In order to establish a lightpath, a route from the source to the destination has to be determined first, and then one of the available continuous wavelengths on the determined route has to be assigned to the lightpath request. An algorithm adopted by a wavelength-routed network to select the route and assign the wavelength to the lightpath establishment request is known as a routing and wavelength assignment algorithm (RWA). These algorithms play a very important role in the connection request blocking rate [26]. In other words, adopting a very good dynamic routing and wavelength assignment algorithm on a wavelength-routed network will improve the network's performance.

Finding a route and assigning a wavelength to a connection request are the two primary jobs of the routing and wavelength assignment algorithm (RWA). Therefore, RWA can be divided into two sub-algorithms, routing algorithm and wavelength assignment algorithm, for simplicity. An overview of the routing algorithms on the optical networks will be presented first, and then the wavelength assignment algorithms.

2.2.1 Routing Algorithms

The routing algorithm is responsible for finding a route between a source-destination pair to try to initiate a lightpath between them. Routing algorithms can be classified as follows:

- Fixed routing.
- Fixed alternate routing.
- Adaptive routing.

2.2.1.1 Fixed Routing

Fixed routing is the simplest routing algorithm. In this algorithm, a fixed route for every source-destination pair is determined offline [52]. Then, any connections between the particular pair of nodes are attempted using the same predetermined route. One example of such an algorithm is a fixed shortest-path routing algorithm. The shortest-path route for each source-destination pair is calculated offline using standard shortest-path algorithms, such as Dijkstra's algorithm. In this algorithm, if a node receives a connection request, it always checks the predetermined fixed route between a specified source-destination pair for an available wavelength [67], which results a high blocking rate for the connection requests. Also, fixed routing is not capable of handling fault conditions in which one or more links in the network fail.

2.2.1.2 Fixed Alternate Routing

In a fixed alternate routing algorithm, k alternative routes, which are a subset of all possible routes between every source-destination pair, are computed offline. Each optical node maintains a routing table that has an ordered list of the k alternative routes between every node pair. For example, this list could be ordered as follows: the shortest-path route, the second shortest-path route, the third shortest-path route, etc. The first route in the ordered list of the routes is called the primary route. Normally, the hop count and the delay are used as the cost metric. If there is only one candidate route, this algorithm will behave as the fixed routing algorithm [42].

The source node attempts to establish the connection request on the primary route first. If there is no wavelength available, it attempts the other routes in sequence. If no available wavelength is found on any of the k alternative routes, then the connection request is blocked.

The fixed alternate routing algorithm can significantly improve the wavelength-routed network performance compared to fixed routing algorithm by reducing connection request blocking probability [50]. It may also be used to offer some degree of link failures protection. The study in [49] shows that having as few as two alternative routes could offer considerable performance improvement instead of having full wavelength conversion at each node with a fixed routing algorithm.

2.2.1.3 Adaptive Routing

In adaptive routing, the routes between a source-destination pair are not computed offline. Therefore, the route is computed when a connection request arrives at the node. The source node finds the best candidate route for the connection request among all possible routes between the source-destination pair by using a dynamic network information state. The network information state is determined by all lightpaths that are currently in use on the network. This network information state gets updated periodically as the traffic on the network changes.

Least-congested-path routing is an example of adaptive routing. When a connection request arrives at a node, the least-congested route among all possible routes is chosen. The number of available wavelengths on a route defines the degree of route congestion. If more than one route has the same degree of congestion, other algorithms could be used to break this tie. For example, the open shortest path (OSP) could be used to select a route among the routes having the same degree of congestion. Research work in [12] studied the employment of the shortest path in combination with the least-congested-path route and showed that this combination gives a higher performance than using least-congested-path routing alone.

2.2.2 Wavelength Assignment Algorithms

After finding the route for a source-destination pair connection request using one of the routing algorithms, a wavelength assignment algorithm runs to select one of the available wavelengths on the selected route and reserves the selected wavelength for this connection request. This section will give an overview of the most popular wavelength assignment algorithms such as:

- First-fit wavelength assignment algorithm
- Most-used wavelength assignment algorithm
- Least-used wavelength assignment algorithm
- Random wavelength assignment algorithm

2.2.2.1 First-Fit Wavelength Assignment Algorithm

The First-fit wavelength assignment algorithm is well-known and has been extensively studied in the context of wavelength-routed networks [70].

In this algorithm, all wavelengths are indexed in increasing order of index value, say, λ_1, $\lambda_2,\cdots,\lambda_i$. After the source node identifies the route to the destination using one of the routing algorithms, the wavelengths are searched in order to find a free wavelength on the identified route. The first available wavelength is assigned to this lightpath request. If all wavelengths are attempted without finding an available wavelength, the lightpath request is blocked. The computation time of this algorithm is low compared to the other algorithms because there is no need to attempt the entire wavelength space for each route since the first available wavelength will be assigned to the lightpath request. In terms of blocking probability and fairness, this algorithm performs well. In a practical wavelength-routed network, this algorithm is preferred because of its low computational overhead and low complexity.

2.2.2.2 Most-Used Wavelength Assignment Algorithm

The most-used wavelength assignment algorithm packs the wavelengths according to their utilization on the network. In other words, this algorithm assigns the current lightpath request to the most utilized and available wavelength on the network. This algorithm needs global information about the wavelengths' usage on the network. The need for global wavelength usage information on the network makes this wavelength assignment algorithm more appropriate for centralized control networks. In this type of network, there is one central control unit for the whole network [52].

When any node wants to establish a lightpath to a specific destination, it sends a request to the central control unit asking for a wavelength assignment. The central control unit searches all the wavelengths in order of their usage on the network on the predetermined route between the source-destination pair. It then assigns the most utilized available wavelength to the connection request, and updates the wavelength usage information for the assigned wavelength. The lightpath request will be blocked if all wavelengths on the route between the source-destination pair are in use or there is no continuous wavelength from source to destination in the absence of the wavelength conversion.

2.2.2.3 Least-Used Wavelength Assignment Algorithm

While the most-used algorithm tries to pack the lightpaths tightly on the wavelengths, the least-used algorithm attempts to balance the load among all the wavelengths. This means the least-utilized free wavelength on the network is assigned to the current lightpath request. Because global information about the wavelengths' usage on the network is required, the implementation of the least-used algorithm in a distributed controlled network is not an easy task. Thus, this algorithm is suitable for a centralized control network [52].

The light path request will be blocked if all wavelengths on the route between the source-destination pair are in use or if there is no continuous wavelength on the route between the source-destination pair in the absence of the wavelength conversion.

2.2.2.4 Random Wavelength Assignment Algorithm

After identifying the available wavelengths on the selected route, the random wavelength assignment algorithm with equal probability chooses one of the available wavelengths and assigns it to the lightpath connection request. The studies in [32, 15] showed through simulation that the blocking probability for the random wavelength assignment algorithm is higher than the blocking probability for the first-fit assignment algorithm.

2.2.3 Lightpath Establishment Protocols

The connection (lightpath) establishment process in the optical network has two main steps. The first step is identifying the route between the specific source-destination pair through one of the routing algorithms. The second step is the selection and reservation of one of the available wavelengths to the lightpath request. Selecting and reserving one of the available wavelengths is known as the wavelength assignment problem. The wavelength assignment problem has two main parts. The first part is the wavelength selection, which can be done through one of the wavelength assignment algorithms discussed previously. The second part is the wavelength reservation process. The wavelength reservation process could be done either by using the forward reservation protocol or the backward reservation protocol.

2.2.3.1 Forward Reservation Protocol

The lightpath establishment using forward reservation protocol will be explained using the following example. Figure 2.1 shows a physical topology for a single-fiber wavelength-routed network that is used as an example to illustrate the lightpath establishment. Assume each node in Figure 2.1 is equipped with a transceiver that can transmit or receive information on any one of the four wavelengths $(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$ that are available on the output fiber. Assume the fixed routing algorithm is adopted on this network. Table 2.1 shows the routing table for node A to all other nodes. Also assume the existing lightpaths on the network and its assigned wavelength as shown in Table 2.2. A lightpath request arrives at node A and wants to get a connection to node F. Node A will be considered as the source node for this lightpath and node F as the destination node. Node A identifies a route, which will be used by this lightpath, to destination node F. Node A will explore its routing table, shown in Table 2.1, for its predetermined route to

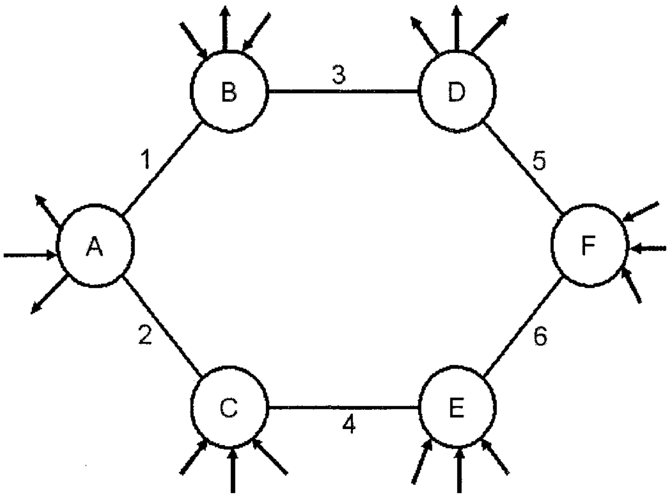

Figure 2.1: Single-fiber wavelength-routed network illustration.

Destinaion Node	Route
B	A-B
C	A-C
D	A-B-D
E	A-C-E
F	A-B-D-F

Table 2.1: Node A routing table.

destination node F. As shown in Table 2.1, this lightpath has to be established on the route A-B-D-F. Node A runs the wavelength assignment algorithm to select and temporarily reserve one or more of the available wavelengths on the fiber that connects node A to node B, say λ_1, and λ_2. After that, the source node sends a reservation message (RESV) to destination node F along the predetermined route to reserve λ_1 or λ_2.

When the reservation message gets to each node along the route, a new available wavelengths list is generated by taking the intersection between the listed wavelengths on the reservation message and the available wavelengths on the node output link. The wavelengths on the new list are temporarily reserved on the

Source-Destinaion Pair	Route	Assigned Wavlength
B-D	1	λ_1
D-F	3	λ_1

Table 2.2: Established Lightpaths

next link. For example, when the reservation message is generated at node A, it is propagated to node B, and then node B generates a new available wavelengths list, which has only one wavelength λ_2. By taking the intersection between λ_1, λ_2 wavelengths listed on the reservation message and the available wavelengths on node B output link $\lambda_2, \lambda_3, \lambda_4$ according to Table 2.2, λ_2 will be listed in the new available wavelength list generated by node B.

In order to utilize the bandwidth efficiently, a failure (FAIL) message will immediately be sent back to the source node to release wavelengths that have been reserved in the previous hops and are found to be busy in the next hop. As in our example, node B sends a failure message to node A to release λ_1 that has been reserved on link 1 and forwards the RESV message to the destination node along the selected route after updating the available wavelengths list.

If the RESV message gets to the destination node, one of the wavelengths available in the RESV message is selected and a confirmation (CONF) message in this regard is sent back from the destination node to the source node. On its way back to the source, the CONF message permanently reserves the selected wavelength and releases the other wavelengths at the intermediate nodes. When the CONF message gets to the source node, it starts transmitting the data on the reserved wavelength.

After transferring all the data, the source node sends a release message to the destination node to release the used wavelength on all hops to be made available for other lightpath requests. The lightpath request will be blocked if the intersection

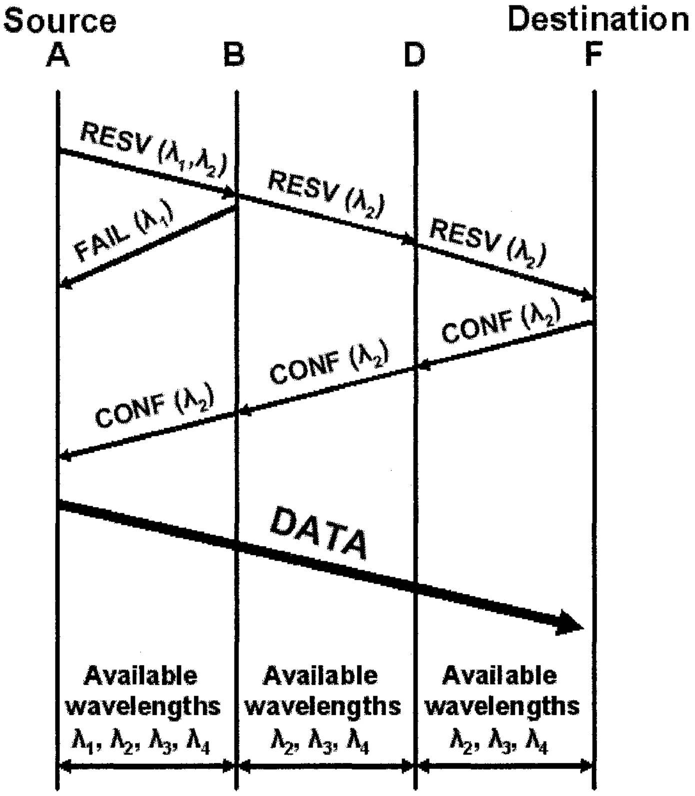

Figure 2.2: Successful lightpath establishment using forward reservation protocol.

operation at any intermediate node produces an empty available wavelengths list, which means none of the pre-selected wavelengths are available on the next hop. Figure 2.2 shows a successful lightpath establishment using the forward reservation protocol and Figure 2.3 shows an unsuccessful lightpath establishment.

Several variations of the forward reservation protocol were studied in [55, 41]. These variations of the forward reservation protocol differ in the number of wavelengths that can be temporarily reserved at any given time for a single lightpath request. For example, the concurrent lightpath requests face a high blocking rate when all available wavelengths are allowed to be reserved for a single lightpath request. On the other hand, the probability for a single lightpath to be established on any of the available wavelengths is high, but the overall wavelength-routed network performance is significantly diminished. Therefore, other variations

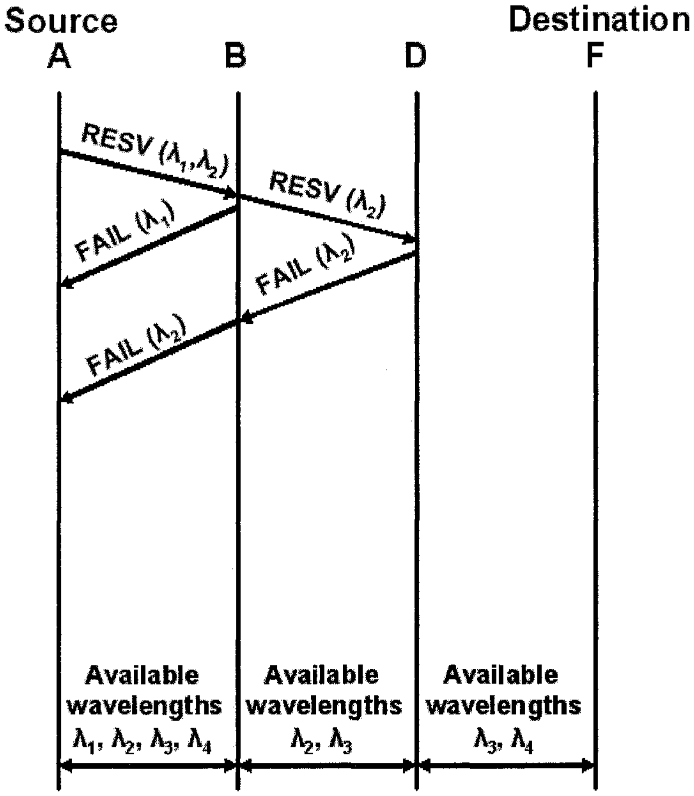

Figure 2.3: Unsuccessful lightpath establishment using forward reservation protocol.

of the forward reservation protocol were proposed. One of these variations is Selective-N. In the Selective-N version, a maximum number of the wavelengths (N) are allowed to be reserved at a time.

All different variations of the forward reservation protocol suffer the bandwidth wastage problem, which comes from reserving a set of wavelengths for a single lightpath request where only one of them will be chosen. Also, if an available wavelength could not be found on a hop, then all reserved wavelengths on all previous hops are considered as wastage bandwidth. This problem appears clearly in larger size networks.

The forward reservation protocol is more suitable to be implemented in a centralized control network [65].

2.2.3.2 Backward Reservation Protocol

In forward reservation protocol, each request reserves multiple wavelengths in order to obtain a higher success rate; this comes at the cost of lower bandwidth utilization efficiency. To overcome the disadvantage of a temporary reservation of the wavelengths in the forward reservation protocol, a backward reservation protocols was proposed in [41].

When a lightpath request arrives at a source node, the source node sends a probe (PROB) message to the destination instead of a RESV message. This PROB message collects the wavelengths availability information on all hops and does not reserve any wavelength. The available wavelengths list in the PROB message gets updated at every intermediate node while it traverses the source-destination route. The updated available wavelengths list is calculated at the intermediate nodes by taking the intersection between the current available wavelengths list and the available wavelengths on the output link. When the PROB message gets to the destination node, one of the available wavelengths is selected through running one of the wavelength assignment algorithms. The destination node sends back to the source node a RESV message, which now reserves the selected wavelength along the route while traversing toward the source node.

If the selected wavelength is not available at some intermediate node, the node sends a FAIL message to the destination node and a negative acknowledgement message (NACK) to the source node. The FAIL message releases the selected wavelength reserved so far, and the NACK message informs the source node of the occurrence of the lightpath establishment failure [55, 41, 60]. The lightpath request blocking could occur in the backward reservation protocol in one of the following two ways:

1. If the intersection operation at any intermediate node during the PROB message journey to the destination node produces an empty available wave-

lengths list, which means the output link of this intermediate node does not have any common wavelength with the previous hops

2. If the selected wavelength is found busy while the RESV message is traveling backward to the source node

The wavelengths availability information that was gathered during the PROB message journey to the destination enhances the backward reservation protocol performance. The performance improvement is achieved because the probability of selecting a busy wavelength in the forward reservation protocol is reduced by knowing the wavelengths usage information on the network at the wavelength selecting time. However, the backward reservation protocol suffers from the occurrence of the lightpath request conflict because the RESV packet reserves only one wavelength. The PROB messages for two different simultaneous lightpath requests sharing one or more common links on their route between source-destination pairs will gather the same wavelengths availability information. Thus, the probability for accidentally selecting the same wavelength by both destination nodes is high, and one of these lightpath requests will be blocked even if other wavelengths are available on the route.

The problem of the unnecessary lightpath request blocking in the backward reservation protocol could be solved by keeping the wavelengths availability information active at the destination node till the RESV message gets to the source node. Then, if the lightpath could not be established on the selected wavelength, the destination node would try another wavelength until the RESV message reaches the source node. This means the lightpath establishment process has been completed. If the destination node attempted all the available wavelengths on the source-destination pair route without any success this lightpath request will be dropped. Since the destination node needs to know when the RESV message reaches the source node, an additional message has to be sent from the source back to the destination to inform the destination node about receiving the RESV message. This version of the backward reservation protocol suffers from a high delay for lightpath setup time since it tries all available wavelengths. Therefore, a limited number of wavelengths that will be tried in this version of the backward reservation protocol will improve the protocol performance and keep the lightpath setup time fairly low.

The lightpath, as its establishment process is explained above in the forward reservation protocol or the backward reservation protocol, has to occupy the same wavelength on all hops between the source and destination nodes. This constraint increases the overall lightpaths blocking rate, which leads to a poor performance of optical network. Some of the lightpath requests would be blocked even if there are wavelengths available on all hops but none of them is a continuous wavelength. The solution for this problem is either enhancing the single-fiber wavelength-routed network by equipping nodes with wavelength conversion capability or using multiple copies of the same set of the wavelength (multi-fiber wavelength-routed network) or by using both solutions.

2.3 Single-Fiber Wavelength-Routed Network

A single-fiber optical network is an optical network using WDM as a multiplexing technology. There is only one fiber connected between every pair of nodes. Different performance models studied the effect of various optical network parameters such as number of wavelengths used, size of the optical switches, the average length of the optical paths, availability of wavelength converters, etc., on the blocking probability and utilization of the network will be reviewed in this section.

In [9], a generalized "reduced load approximation" scheme is used to analyze blocking probabilities in optical networks using a fixed routing algorithm and a least-loaded routing algorithm. If there is at least one common free wavelength on all hops of a predetermined route between the source-destination pair, the lightpath will be established on that common wavelength; otherwise, this request will be blocked. In the least-loaded routing algorithm, a lightpath request is accepted if there is an available wavelength along all hops of the path; otherwise, lightpath establishment is attempted on the alternative route with the largest number of available wavelengths. Numerical results in this research confirmed the accuracy of the researchers' approximation model for the fixed routing case. However, it is less accurate for the least-loaded algorithm. Also, this work shows the difference in computational requirements between a circuit-switched network and a wavelength-routed network. The circuit-switched network required $O(C)$ operations to compute the blocking probability, where C is the number of channels on a hop. In contrast, the wavelength-routed network required $O(C^3)$ operations for a two hops and $O(C^4)$ for a three hops lightpath.

Logical topology for a given wavelength-routed network physical topology, when the traffic that needs to be carried between all source-destination pairs is known, is considered in [3] by Banerjee and Chen. A circuit-switched network was considered in this work. Designing a logical topology that can minimize congestion for the given traffic demand was the main objective for this work. The logical connections are realized by wavelength continuous lightpaths between end users, which means no wavelength conversion is available on the physical topology. An analytical model for obtaining maximum and average connection loads for a given logical network and traffic demand matrix was developed and compared to simulation results. The lower bound of maximum congestion on logical topology for a proposed analytical model showed that the shortest-path-based routing is not efficient in reducing logical connection congestion. Two heuristic algorithms for designing logical topology were proposed. The first algorithm tries to maintain network connectivity by optimizing one-hop traffic. This heuristic algorithm performed 30% better than the random topology algorithm. The second algorithm uses minimum matching algorithm to maximize traffic that a logical connection has to carry between all source-destination pairs.

In [57] a sparse wavelength-routed network was considered. The sparse wavelength-routed network is a wavelength-routed network where a few selected nodes have a full wavelength conversion capability. An approximate blocking model for sparse networks was developed in [57]. The analysis in [57] showed that in most cases, a good performance can be achieved if only a small number of nodes have wavelength conversion ability. This study also showed that increasing the network's physical connectivity between nodes reduces the effects of the wavelength conversion on overall network performance. The lightpaths' correlation due to wavelength continuity constraint was analyzed. This correlation in the ring network is significant. Further, it showed that the wavelength conversion has more effect in mesh-tours networks than ring networks because lightpath length is longer in mesh-tours.

Another probabilistic model for a wavelength-routed network is presented in [7]. Lightpath blocking probability is calculated using the developed model for a wavelength-routed network with and without wavelength conversion. The main goal of this work was to show the importance of the wavelength conversion in the wavelength-routed network environment. Two different analytical models were developed in [7] to study the effect of lightpath length on the wavelength-routed network performance. These models utilized the analytical model for the circuit-switched network that was introduced by C. Lee in [33]. Under the steady state condition of the network, the use of a wavelength on a hop is statistically independent form all other wavelengths on the same hop. Also, the use of the same wavelength on different hops is statistically independent. The work in [7] studied the lightpath effect on the wavelength-routed network without wavelength conversion first. In the absences of the wavelength conversion as the case considered in this analytical model, the lightpath request is blocked if there is no continuous wavelength on all hops between the source-destination pair. The blocking probability model was presented in this study as follows:

$$P_{withoutconversion} = (1 - (1 - \rho)^{H})^{W} \tag{2.1}$$

where H is the number of hops between source-destination pair under consideration, ρ is the probability of a specific wavelength is busy on a hop, and W is the number of wavelengths on a hop. Also, the achievable utilization for a given blocking probability on specific route between a particular source-destination pair was presented as follows:

$$q_{withoutconversion} = 1 - (1 - P^{\frac{1}{W}}_{withoutconversion})^{\frac{1}{H}} \tag{2.2}$$

The second scenario considered in this work was a wavelength-routed network with dedicated conversion capability at each node. In this case, the lightpath blocking would happen only if all wavelengths are busy on any hop on the route between the source-destination pair under consideration. The blocking probability for a wavelength-routed network with dedicated wavelength conversion capability was given as follows:

$$P_{withconversion} = 1 - (1 - \rho^{W})^{H} \tag{2.3}$$

Also the utilization for this case was calculated as:

$$q_{withconversion} = 1 - (1 - P^{\frac{1}{W}}{}_{withconversion})^{\frac{1}{H}} \tag{2.4}$$

Also this work assumed that the access nodes (source and destination nodes) can transmit or receive data on any wavelength. The source node can establish only one lightpath at a time with one destination. This study showed the blocking probability for the wavelength-routed network increases with the number of hops H. This means the lightpath length is an important design parameter for networks without wavelength conversion. So, for a small size network the effect of the wavelength conversion on the blocking performance is negligible. But the wavelength conversion would be a very effective design parameter if the network size increases since it allows the lightpath to be established on a different wavelength if the requested wavelength is not available. In conclusion, Barry and Humblet in [7] showed that the network diameter has more effects on wavelength-routed network performance than the switch size.

2.4 Multi-Fiber Wavelength-Routed Network

With the increased demand for network capacity, the network bandwidth requirements have increased dramatically. Optical networks based on Wavelength Division Multiplexing (WDM) are emerging to utilize the enormous bandwidth offered by the optical fibers to fulfill the ever increasing need for capacity. Due to the expectation that the cost of wavelength converters is likely to remain high in the near future, using multiple fibers on each link in WDM networks as an alternative solution to overcome the wavelength continuity constraint has become an important topic for research. In these multi-fiber WDM networks, each link consists of multiple fibers. Thus, multiple fibers in WDM networks have the same effect as the limited range wavelength conversion. In other words, using multiple fibers in the optical network enhances the blocking performance due to the reduction of the wavelength continuity constraint effect even in the absence of the wavelength conversion resources. In the remainder of this section, we will overview a number of publications that tackled this topic.

Fixed-paths least-congestion routing in multi-fiber WDM networks were studied in [35]. In this work, a multi-fiber network without wavelength conversion was considered. In the fixed-paths least-congestion routing algorithm, a set of routes to be used for each source-destination pair in a network is determined offline, and the route information is stored at each source node. If a connection request arrives, the least congested route is selected to set up the request. The request is blocked if no wavelength is free on any predetermined routes. An analytical model based on the link load correlation was developed to evaluate the blocking performance of the fixed-paths least-congestion routing algorithm. The two-hop path technique was used while developing the analytical model. The two-hop path was modeled as a three dimensional Markov chain and the blocking probability for the L-hop path was computed recursively. Due to load correlation considered in this model, and the recursive nature of calculating the blocking probability, this model takes extensive calculation operations. This extensive computation increases with the size of the network. Even this model showed that while it was accurate for fixed-path least-congestion algorithm, it is not suitable for a large network size.

The study in [36] considered the important question of how to decide how many fibers per link are required to guarantee high network performance in the multi-fiber settings of optical networks. It showed that the fiber requirement depends on many factors, such as the network topology, traffic patterns, the number of wavelengths per fiber, and the routing algorithm employed in the network. The study of fiber requirement was conducted under dynamic traffic using different topologies with alternate path routing (APR). This research work developed an analytical model to evaluate the blocking performance of such networks. The work assumed that the multi-fiber system has a similar effect to the limited wavelength conversion in WDM networks, and accordingly claimed that their analytical model is also applicable in limited-wavelength-convertible networks. They also claimed that a multi-fiber network has similar blocking performance to a full-wavelength-convertible network if they select the wavelength-fiber-pairs adequately. And accordingly they thought that multi-fiber WDM networks without wavelength conversion are not only feasible, but also a desirable choice under current technologies. These claims will be further tested in this research work and explained further. The analytical and simulation models' results showed that the number of required fibers per link to provide high network performance is slightly higher in the alternate path routing than the fixed-path routing. It also showed that a small number of fibers per link are still sufficient to guarantee high network performance in both the regular mesh-torus networks and the irregular NSFnet with alternate path routing. The study in [36] utilized the blocking probability model that was developed in [35, 37]. As mentioned before, the analytical model that was developed in [35, 37] needs very extensive computation, especially for the longer lightpaths.

The work in [10] considers wavelength band switching (WBS), which attracted attention from the optical networking industry for its practical importance in reducing the control complexity and cost of optical cross-connects (OXCs). This work addresses the theoretical aspects of wavelength band switching. The authors recognized the differences of wavelength band routing from wavelength routing, which meant that techniques developed for wavelength-routed networks could not be directly applied to effectively address wavelength band switching related problems. The paper proposed a new multi-granular OXC (MG-OXC) architecture for wavelength band switching, which they claimed to be more flexible than any existing wavelength band switching node architectures. The work claimed the adaptation of the most powerful waveband assignment strategy, and accordingly the development of an efficient heuristic algorithm called Balanced Path routing with Heavy-Traffic first (BPHT). To verify its near-optimality, they also developed an integer linear programming (ILP) model. Both the ILP and the BPHT algorithms handled the case with multiple fibers per link and hence are more general than their previous single-fiber solutions. They conducted a comprehensive evaluation of the benefits of WBS through detailed analysis and simulations. The proposed heuristic, BPHT, performed much better than a heuristic which applies the optimal routing and wavelength assignment (RWA) method. They showed that WBS using BPHT is more beneficial in multi-fiber networks than in single-fiber networks in terms of reducing the port count. They showed that the waveband granularity has a large effect on the performance of WBS networks. In particular, with appropriate waveband granularity, using MG-OXCs in multi-fiber networks can save up to 70% of ports compared to using ordinary-OXCs. They demonstrated that their ILP formulations and heuristics are useful for the efficient design of MG-OXC nodes (i.e. the dimensioning of the switching matrices) for a given set of traffic demands, and also can be used to minimize the number of used active ports in an existing network, and thus lower network operating costs, and reduce blocking probability of future requests. Their analytical and simulation results provided valuable insights into the effect of wavelength band granularity, as well as the trade-offs between the wavelength-hop and the port count required in WBS networks.

In [40], the authors proposed a computational model for calculating blocking probabilities of multi-fiber WDM optical networks. They analyzed the blocking probabilities in multi-fiber WDM optical networks from a bottom-up manner. They employed the static shortest lightpath routing as the routing policy, and wavelengths were assigned randomly. They first derived the blocking probability of a fiber based on a Markov chain, from which the blocking probability of a link is derived by means of conditional probabilities; then they extended it to the link model, which includes multiple fibers. The blocking probability lightpath model, which was based on the idle wavelength availability in all of the intermediate links, was obtained by a recursive formula. They expressed the network-wide blocking probability as the ratio of the total blocked load versus the total offered load. Their simulation results for different fiber-wavelength configurations conform closely to the numerical results based on their proposed analytical model, thus demonstrating the feasibility of their proposed model for estimating the blocking performance of multi-fiber WDM optical networks. Their simulations were conducted on the NSFNET network with different fiber-wavelength combinations of the same link capacity to verify their results.

The work in [38], analyzed the blocking probability of distributed lightpath establishment in wavelength-routed WDM networks by studying the two basic methods: destination-initiated reservation (DIR) and source-initiated reservation (SIR). It discussed three basic types of connection blocking: 1) blocking due to insufficient network capacity; 2) blocking due to outdated information; and 3) blocking due to over-reservation. In their view, the latter two types of blocking would become increasingly important when we are handling more and more bursty traffic loads. The study offered an insight into the blocking behavior of distributed lightpath establishment schemes, since the connections blocking are incurred in one combination or another in most schemes, due to the fundamental nature of the distributed process involved in setting up lightpaths. The analysis results were compared to the simulation results on both the PacNet and a 12-node optical ring, where the length of the fiber between every two adjacent nodes is 100 kilometers. They suggested that some network capacity has to be reserved for a short period of time before the data transmission begins due to the propagation delay. The resulted "capacity wastage" is more significant when connection requests arrive at a high rate with a short average duration. By taking this fact into consideration, they achieved higher accuracy in blocking analysis. The study observed that under light traffic load, the blocking mainly takes place in the backward direction, caused by outdated information; whereas under heavy traffic load, the blocking occurs mainly in the forward direction, due to insufficient network capacity. It was shown that the proposed models are accurate for both the DIR and the SIR methods, in both the regular and irregular network topologies, under the whole range of traffic loads.

The work in [44] analyzed the trade-off between link cost and node cost of conventional WDM ring networks and also included a performance comparison. They hinted that the results of their analysis would lead to a new breed of multi-fiber WDM ring networks with the simplest nodes and modest link efficiency. The paper discussed the design and performance of such multi-fiber WDM ring networks and highlighted its most promising application areas with respect to the number of nodes and ring length. The proposed WDM ring networks, which employ high-count fiber cables for growth, provides direct end-to-end optical paths between pairs of nodes, thereby removing the routing process from the intermediate nodes to improve system operation and management. From an analysis of network cost, they showed that the proposed network could be a candidate for the backbone network in metropolitan areas. Critical sub-systems were demonstrated to show the viability of the proposed WDM ring networks, which offer platforms that will accommodate growing multimedia services. The paper also reviewed the status of the critical technologies needed for implementing the proposed network with tera-bit/s capacity.

The analytical model In [36]depended on MLLC (multi-fiber link-load correlation) that required an extensive computational power, but did not consider the case where wavelength conversion is a factor. It also assumed a fixed the number of channels for supporting a fixed load and did not evaluate the behavior of the network under dynamic load conditions. In [10], the evaluation of the network performance was based on a heuristic algorithm, which meant evaluating the performance under static conditions to get the optimal solution. The implementation of waveband switching limited the performance, where conversion could not be implemented, and constrained the performance for a reduction in the hardware complexity. The works in [40, 38, 44] did not consider the introduction of wavelength conversion and accordingly the effects of such technology was not considered. In [40], the authors assumed that the loads per fiber were identical which contradicted their adopted wavelength assignment scheme, which states that a wavelength can be switched to any of the F parallel fibers.

2.5 Optical Packet Switching

In this section, the development and assessment of optical packet switching networks and its capability of providing transparency to the payload bit rate in the optical domain will be discussed. This networking concept concentrates on providing a mechanism to transport the present day IP based traffic over light. Optical packet switching offers a number of layer two and layer three functions via optical layer processing. A network built on top of an optical packet switched network will have to not only provide IP-like connectionless services, but also support higher quality connection-oriented services. This will require the network to have differentiated service qualities on the packet level. With today's technology, the received packet by the optical packet switch splits into two different optical signals (header and payload). Since the switch control and routing logic inside the switch are performed electronically, the header has to be converted to an electrical signal to be processed and to identify the packet destination. Therefore, the switching elements inside the switch can be setup properly. The processing speed of the header has to be at least equal to the data transmission rate in order to avoid delaying the payload. So, the necessity for a very high speed control plane and ultra-fast switching elements is the most challenging problem for optical packet networks[63].

OPS suffers from packets contention problem that arises when two or more of the incoming packets on the same wavelength intend to leave the switch through the same output port, which results in packet loss and poor network performance. The contention resolution schemes in optical switches could be time dimensioned [4], space dimensioned [17], or both at the same time.

Time dimension contention resolution scheme is realized by buffering the contending packets within the switch till the requested output port is cleared. Due to current technology limitations the only way to implement optical buffer is through fiber delay lines (FDLs). In this scheme, one of the contending packets is forwarded to the desired output port and the remaining packets are forwarded to FDLs where they are delayed for a fixed time, during which the contention might be resolved [27].

Optical buffer in OPS could be categorized as output buffer where the contending packets are stored in the FDL buffer attached to each output port, or as shared buffer. In the shared buffer, all the packets failing to leave the switch through the requested output port are sent to FDL to be buffered for a fixed time and re-enter the switch again to compete with the new packets on the desired output. Feasibility of using the FDL as a contention resolution scheme is limited by the non-linear effect and fiber optic cable's physical limitations such as signal attenuation and dispersion which arise due to FDL length. The signal power budget is another limitation for using long FDL since the receiver of the destination or the receiver at the switch input cannot detect the optical signal with power under a certain limit. Also the number of FDLs is an important design parameter for OPS because it has an impact on the switch size and cost [19].

So, effectiveness of the use of the FDL as a contention resolution scheme is restricted and limited by the above mentioned limitations [24].

Since the time dimension contention resolution scheme in optical domain suffers the above mentioned problems, this scheme has been avoided as much as possible. Therefore, utilizing the space dimension of the switch is the alternative contention resolution scheme. In space dimension contention resolution scheme the contending packets utilize the available resource of the OPS instead of getting buffered for a fixed time as in the time dimension contention resolution. The OPS resources that can be used in this scheme are multiple output ports, wavelength conversion and wavelengths.

Deflection routing is an example of the space dimension contention resolving technique. In deflection routing scheme, one of the contending packets utilizes the intended output port while the others are deflected to other available output ports if there are,otherwise they are dropped. Using deflection routing in OPS reduces the need for FDL and enhance over all performance of the OPS since the probability of the packet loss reduces. This contention resolution scheme performs very well in small networks with high connectivity. So, the effectiveness of this scheme significantly depends on the network topology and the nodes connectivity. The drawback of this approach is that the deflected packet may have to traverse a longer path to reach the destination, which may result in a significant crosstalk noise in the optical signal and unacceptable delay which means packets have to be re-ordered at destination. Also, the network becomes more congested as more deflections take place. This scheme could lead to poor wavelength utilization and low network performance if the deflected packets are dropped after a number of deflections.

The research work in [53] gives an overview of the characteristics and challenges of optical packet switching. It shows that the continuous progress in photonic technology will permit the development of new systems to meet the needs of processing and switching IP-based traffic in the terabit range. These developments will be induced by the removal of critical electronic system limits, high electromagnetic interference, and power consumption that exist in present day electronic switching elements. This paper presented some potential architecture of photonic packet switching and their key functionalities and also the feasibility of photonic packet-switching fabrics with state-of-the-art photonic technology. The feasibility of such a technology is limited to the advancement of the optical switching elements and the architectural solutions adopted for handling the IP-based traffic in the optical domain as far as routing and contention resolution go. The key components and subsystems to realize large high-speed fabrics capable of satisfying optical switching of IP-based traffic require an optical technology to process the packet header and extract the routing information without delaying the packet and within the optical domain to maintain transparency. An assessment of the transparent optical packet concept (i.e. packets based on a fixed duration payload transparent to the bit rate) as a possible transport and switching technique under study in the framework of the KEOPS project has been presented. The implementation of such a system, as the paper claims, could improve link utilization efficiency through a refinement of the switching granularity by exploiting the time domain while simultaneously keeping the advantages of photonic technologies. The paper shows that the main challenges for photonic packet-switching systems and networks relate to providing complex functionalities such as buffering, synchronization, and processing. Significant progress has been made by conceiving new optical packet formats and interfaces, including processing functions, and also by modifying or complementing the existing standards when applied in all-optical network.

The study in [68] considered the fundamental issue of contention resolution in optical packet switch due to its importance in increasing the utilization of the optical packet switching when packets from different input ports are destined to the same output port at the same time. The main aim of the paper was a trial to emphasize the need for optical RAM that could facilitate the implementation of optical packet switching. The fiber delay lines (FDLs), which rely on the propagation delay of the optical signal in silica to buffer packets in time, were considered to complement the existence of the optical packing switching technology. The study mentioned that FDLs are bulky and not scalable and the increased use of FDLs will significantly increase the cost and size of the switching fabric, which means that the number of FDLs to be used for the buffering purpose is very limited. The work done in this paper performed an analytical throughput analysis on large optical packet switches under heavy traffic load and used simulations to validate their analysis to answer the question of whether optical packet switching is practical in core networks.. Their results from performance evaluation indicated that throughput of large optical packet switches is not satisfactory under heavy traffic load, which is expected of the optical core. Both analysis and simulations revealed that optical packet switching in core networks would not be practical without a breakthrough in the research area of optical buffering technology.

The study in [20] investigated the performances of packet-switching architectures working in synchronous and asynchronous settings. It also considered the use of wavelength converters, in a shared manner, to resolve packet contention and their effects on these systems. The paper investigated how many converters could be saved by utilizing a sharing technique for the conversion resources in the synchronous and asynchrony architectures and then it compared the obtained results for such systems. Two different sharing strategies were considered: shared per output fiber (SPL architecture) and share-per-node architecture. The simulation and analytical models showed that the switching architectures operating in a synchronous way allow a greater saving of wavelength converters. These results were considered under both unicast and multicast traffic. In the multicast traffic case, it was demonstrated that the implementation of SPN architecture is not convenient because it allows less saving of wavelength converters and further increases the complexity of the switching matrix. This performance worsening is due to a positive correlation effect of the conversions number required by the packets directed to the various output fibers of the switch and the need for wavelength conversion.

The study in [21] analyzes an architecture for optical packet switches in which the wavelength converters are shared per input line (SPIL). The architecture performance is evaluated by means of an analytical model and is compared with that of an optical packet switch architecture in which the wavelength converters are shared per output line (SPOL). It has been demonstrated that the SPIL architecture, even if a very simple control logic is adopted, allows for the saving of a remarkable number of wavelength converters. They assumed that the optical packet switching operation is synchronous on a time-slot basis. The analytical model presented aimed at evaluating the packet loss probability of a SPIL switch versus the number of converters required. The packet loss probability in both architectures considered for comparison depended on the adopted Packet Scheduling Algorithm (PSA) and was minimized by optimizing the assignment of wavelength converters to the packets that needed wavelength conversion. The study used a Packet Scheduling Algorithm proposed in an earlier publication.

In study [64], the authors presented a comparison between a slotted and unslotted all-optical packet-switched network with priority-based routing. It discussed how migrating the switching function from electronic domain to optical domain can help resolve the optical-electrical-optical conversion bottleneck. By means of simulation, the authors implemented this comparison study of the packet loss rate between the two networks. By having four fiber delay lines on each switching node of the unslotted network while incorporating both limited wavelength conversion and limited deflection routing, the authors were able to achieve a packet loss rate of less than 0.01 for transmitter load under 0.3 and to match it with the performance of slotted network with one fiber delay line. This means that they simulated slotted and unslotted networks and by increasing the number of fiber delay line in the unslotted network node, they tried to reach the performance of a slotted network with one fiber delay line. Accordingly, they investigated the possibility of having classified packet priorities. It was shown that a four fiber delay-line per node topology would a provide packet loss rate of less than 0.01 for all three priority classes with transmitter load less than 0.3. These results indicated that it is possible to avoid the complicated packet fragmentation/reassembly and synchronization stages required by the slotted network and accommodate variable packet size without sacrificing network performance.

The papers reviewed in this section did not consider the introduction and the implementation of multiple fibers in optical packet switching networks. Therefore, the performance of the optical packet networks lacked a new dimension of the packets contention resolution. Also, the analysis in the pervious studies considered the symmetrical case per the input traffic, which means all the analysis were carried out under the assumption the load on all inputs are equal. In reality this assumption is not failed since it is very rarely or could be impractical to find a switch satisfy this condition.

2.6 Optical Burst Switched Ntework

Optical Burst Switching (OBS) is a switching paradigm that combines packet switching and circuit switching nature to reduce the overhead management. Using today's optical network technology the Optical burst switching seems the most practical implementation for all-optical network. In the OBS network the buffering of packets can only be at the edge nodes. Therefore, the optical traffic received by the core nodes inside the optical cloud is either forwarded or dropped. The edge nodes aggregate multiple packets destined to the same destination to resample a larger packet (Burst) until the node has enough data or a burst formation timeout occurs. At this point, the burst is sent through the all-optical core under one header which means the overhead control and management that appear in circuit switching and packet switching will be reduced since a single header will be processed for multiple packets.

A control packet that contains the burst header is sent before the burst to reserve the needed resources and configure the switching elements in the core nodes along the route of the burst. Then the burst is sent after an offset time that is necessary for the header to be processed and nodes to be setup. If all needed resources are successfully reserved along the burst's route, the burst traverses the route, and then the reserved resources are released once the burst has finished. The burst can be either deflected or converted to another wavelength if the requested wavelength is not available on one of the intermediate hops in case any of these contention schemes are deployed by the OBS network; otherwise the burst will be dropped.

Just-In-Time (JIT) and Just-Enough-Time (JET) are the main signaling protocols proposed in literature for BOS network. In the JIT signaling protocol, control messages are sent immediately before the burst on an out-band wavelength along the route to reserve the resources in the intermediate switches. The reserved resources will not be released till another release control message is sent along the route to release all reserved resources. Reserving the optical resources immediately before the burst arrives minimizes the waiting time before the burst is transmitted. So, using the JIT for BOS network signaling makes the network infrastructure independent of the data format and therefore, bursts travel transparently through the configured path. The JIT requires an optical buffer at the intermediate nodes to buffer the burst while the header gets processed and switching elements configured. In contrast, in Just-Enough-Time (JET) the ingress node where the burst is sampled sends only the burst header on the control wavelength to configure the intermediate switches and after an estimated offset time the burst will be sent. In this signaling protocol the burst length and the arrival time for the burst have to be known in order to reserve the resources at the intermediate nodes for a fixed period of time. This reservation technique allows a better utilization of the resources at the intermediate nodes and eliminates the optical buffer inside the optical cloud. The offset time (time between sending the header and start sending the burst) has to be carefully chosen since a larger time means a larger delay for the burst.

2.7 Wavelength Conversion

The wavelength continuity constraint, which restricts the lightpath to be established on the same wavelength on all links, imposes a performance limitation on the wavelength-routed network. Allowing a lightpath to be set up on different wavelengths throughout the source-destination pair improves the overall wavelength-routed network performance by reducing the lightpath setup requests blocking rate. Permitting the lightpath to occupy different wavelengths throughout the source-destination pair route can be realized by using a wavelength converter at the intermediate nodes where the requested wavelength from a pervious hop is not available on the next hop [34]. The switching node that is equipped by the wavelength conversion capability is known as a wavelength-convertible switch.

According to wavelength conversion scheme that deployed by wavelength-convertible switches they can be classified as follows:

1. Dedicated wavelength-convertible switch
2. Share-per-node wavelength-convertible switch
3. Share-per-link wavelength-convertible switch

The converters in each of the above wavelength-convertible switch architecture are either full-range converters or limited-range converters. Thus, the wavelength-routed network performance differs according to the wavelength-convertible switch architecture that is adopted at the switching nodes. The following sections describe the different wavelength-convertible nodes architectures and the lightpath establishment process using these architectures.

2.7.1 Dedicated Wavelength-Convertible Switch

Figure 2.4 shows the dedicated wavelength-convertible node architecture. In this architecture, each outgoing wavelength has a dedicated wavelength converter. This switch architecture provides the optical network the highest flexibility configuration, since each outgoing wavelength has its own full-range wavelength converter [47, 34]. If a lightpath request arrives at a switching node that has adopted this convertible switch architecture, the switching node control plane checks the outgoing requested wavelength and forwards the request on this wavelength if it is free. Otherwise, the control plane searches all other outgoing wavelengths. If a free wavelength is found, then the switching node control plane forwards the request

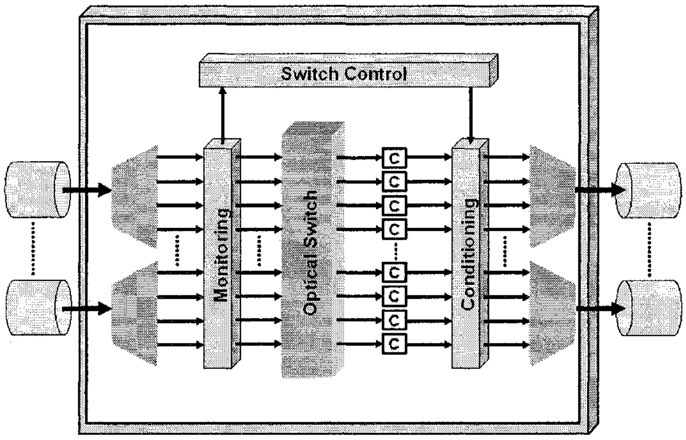

Figure 2.4: Dedicated wavelength-convertible switch architecture.

to the free wavelength converter to be converted to the new desired free wavelength. Therefore, using the dedicated wavelength-convertible switch architecture at all switching nodes in the optical network overcomes the single-fiber optical network performance limitation due to the wavelength continuity constraint. The routing and wavelength assignment problem on this type of network behaves as a normal circuit-switched network where the only limiting factor is the number of channels available on each output link. In other words, the outgoing wavelength exhaustion is the only reason for the lightpath blocking in the dedicated wavelength-convertible switch. Due to high network cost associated with using the dedicated wavelength-convertible switch at all switching nodes, the dedicated wavelength-convertible switch setting is not a feasible solution for the optical network performance limitation imposed by the wavelength continuity constraint, and so it can not be considered as an optimal solution for the single-fiber network environment.

2.7.2 Share-Per-Node Wavelength-Convertible Switch

In the share-per-node wavelength-convertible switch architecture, a limited number of full-range converters are packed in a single-wavelength converter bank [34]. The wavelength converter bank is provided for the entire switch, as shown in Figure 2.5. This architecture introduces new hardware to the switching node [47]. The new hardware is an optical switch that will be used to switch the converted signal (the output of the conversion bank) to the desired output fiber. If a reservation message for a lightpath request arrives at a share-per-node wavelength convertible switch, the control plane checks the requested wavelength on the desired output link first. If it is available, it will be reserved for this lightpath request and will forward the reservation message to the next node. Otherwise, the control plane checks the conversion bank for a free converter and checks the output wavelengths for a free wavelength. If a free converter and a free wavelength are found,

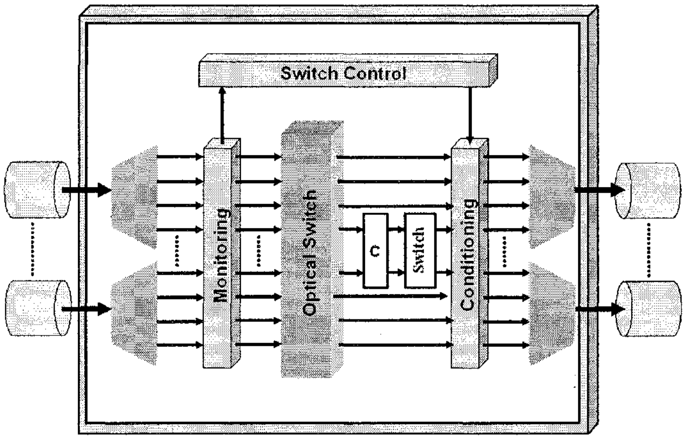

Figure 2.5: Share-per node wavelength-convertible switch architecture.

the lightpath request will be assigned to this free wavelength and converter. The converter will be used to convert the original wavelength to the free wavelength. Figure 2.6 shows a flow chart for the lightpath establishment procedure explained above.

The lightpath blocking in this switch architecture occurs if the requested wavelength is busy on the desired output link and all the converters in the conversion are busy or there is a free wavelength converter and all the other wavelengths are busy [47, 34]. The switch cost reduces due to using a limited number of the wavelengths conversion resources. However, its complexity increases by introducing the second optical switch.

2.7.3 Share-Per-Link Wavelength-Convertible Switch

The share-per-link wavelength-convertible switch architectures is shown in Figure 2.7. In Figure 2.7, the converters at the switching node are divided into a number of conversion banks. Each conversion bank is dedicated to one of the outgoing fiber links. The converters in a specific conversion bank can be accessed only by the incoming wavelengths that are destined for any wavelengths in the outgoing link that this conversion bank is associated with. In this architecture, the optical

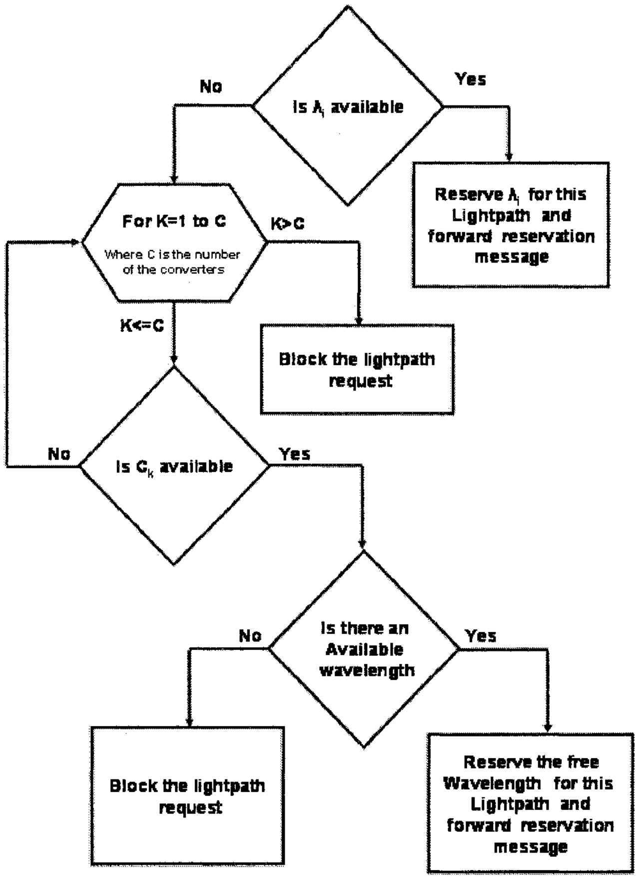

Figure 2.6: Lightpath establishment flow chart in share-per-node wavelength-convertible switch.

switch routes the wavelengths that need conversion to the conversion bank that is associated with the desired output link. This switch architecture reduces the hardware complexity that is introduced by the share-per-node wavelength convertible switch.

When an optical signal arrives at a switching node that adopts this switching architecture it is de-multiplexed to different wavelengths, and then the optical switch routes each of the wavelengths to the request output link according to its incoming port and wavelength. If the control plane figures out that the same wavelength on the output link is busy, it sets up the optical switch to direct this

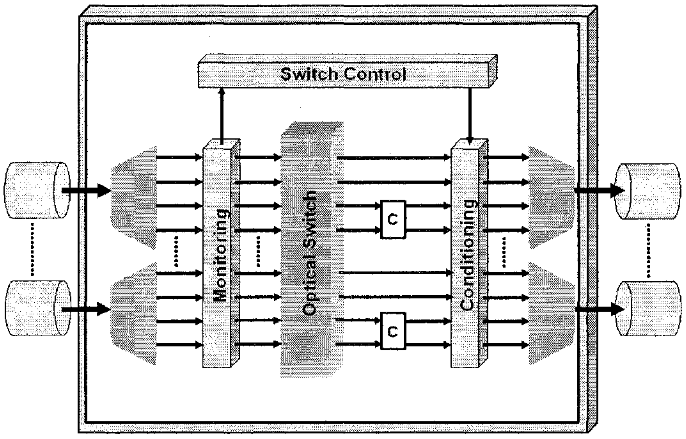

Figure 2.7: Share-per-link wavelength-convertible switch architecture.

wavelength to a special port that leads to the conversion bank associated with the desired output link. The lightpath blocking in this switch architecture will take place if the requested wavelength is busy on the requested output link and either there is no converter available in the conversion bank that associated with the desired output link or all the wavelengths on this link are busy.

Adopting any of the above wavelength conversion mechanisms in all-optical network will enhance its performance. This performance improvement differs from one conversion mechanism to another. While the dedicated wavelength conversion gives the best improvement, the low limited range conversion gives the least performance improvement. Using the dedicated wavelength conversion scheme to achieve the best performance gain is not a feasible solution, since the cost of the conversion is high due today's wavelength conversion technology limitations. Therefore, many research works have been investigating the optical network performance when a limited number of the costly optical resources are used. On the other hand, other research studies explored different alternative solutions such as multi-fiber environment to enhance the optical network performance and keep its implantation cost as minimum as possible. Multi-fiber environment solution is considered as the main motivation needs to be explored in detail for this research.

2.8 Optical Network Survivability

The integration of network technologies with the fast changes of fiber optic networking techniques, the prevention of service interruption, and the reduction of loss of service to a minimum have become a critical research issues. The survivability of an optical network and its ability to provide continuous service in the presence of failures where the optical fiber is the transport medium determines the capacity, reliability, cost, and scalability of the networking system. Survivability mechanisms used in today's optical networks and recent research efforts in providing survivability for next-generation WDM networks is the main concentration of this section.

This publication in [69] took a look at the techniques used to achieve survivability in optical networks, and how those techniques are evolving to make next-generation WDM networks survivable. It examined various survivability techniques for optical networks. These techniques were classified as pre-designed protection and dynamic restoration techniques. They explored the automatic protection switching and self-healing rings in the electro-optic SONET network. The paper argued that these techniques may be adapted to WDM networks with some modifications. The authors also argued that the availability of multiple channels in a WDM link, which provides flexibility in capacity assignment, makes the survivability design more complicated due to the constraints on wavelength assignment. In other words, the multiplicity in wavelengths makes the protection schemes more complicated. The paper showed that most survivability schemes proposed for WDM optical networks have been pre-designed protection against single-link failures. They expected that survivability schemes for channel failures will become an important topic of research in the future. Also, they expected dynamic restoration to provide a practical alternative to pre-designed protection schemes. Path and link restoration schemes are compared using two metrics: average restoration time and restoration efficiency, which is defined as the proportion of the failed connections that are restored.

The work in [13] discussed several survivability techniques, which were developed for use in SONET and SDH networks (non-WDM networks), used in optical networks. It explored these techniques in detail and their applicability to WDM networks. Different network architectures based on point-to-point, ring and mesh topologies and their restoration techniques were discussed. Different automatic protection techniques like Linear, Ring based for SONET were explored. In WDM networks, there were two main categories of restoration techniques path protection and link protection, that were paid much attention to in the paper. The paper argued that for effective fault recovery of networks, protection routes must be preplanned considering various factors like available resources, the extent to which the network must be protected and allowed time within which the network must be restored. As in the previous paper, this paper also argued that WDM network survivability in mesh networks is much more difficult to plan and control than in rings and it is largely an open and pre-standard issue. Some of the stressed points in the paper are the following. In path protection, the source and destination nodes of each connection statically reserve backup paths on an end-to-end basis during call setup. In path restoration the source and destination nodes of each connection that traverses a failed link dynamically discover a backup route on an end-to-end basis after the link failure and the whole path between the two endpoints of the traffic is replaced. It showed that dedicated path protection is resource consuming in mesh networks because of physical route diversity constraint whereas in the shared protection path scheme, a single protection path is used to protect a set of N paths. The paper argued that it is difficult to provide a dedicated backup path around each link of the primary call and on the same wavelength as the primary path. It explained that in shared-link protection, protection fibers can be used for protection of more than one link. Hence the capacity reserved for protection is greatly reduced. This paper was more of an overview of the survivability mechanisms that could be adopted and used in DWM optical networks.

The work in [62] considered the problem of designing a network of optical cross-connects (OXCs) which provides end-to-end lightpath services to large numbers of client nodes, under the requirement that the network will survive any single-link failure. The main objective of the study was to quantify the additional resource requirements of implementing path protection schemes over a network with no survivability properties. The authors presented a heuristic routing and wavelength assignment algorithms for dedicated path protection and two variants of shared path protection, and integrated them into the physical and logical topology design framework they developed in an earlier study. They applied their heuristics to networks with up to 1000 client nodes, with a number of lightpaths that was an order of magnitude greater than the number of clients, and for a wide range of values for system parameters such as the number of wavelengths per fiber, the number of optical transceivers per client node, and the number of ports per OXC. The results provided insights into the relative resource requirements of dedicated and shared path protection schemes. The paper also concluded that with using shared path protection schemes, it is possible to build cost-effective survivable networks that provide rich connectivity among client nodes with only a modest additional amount of resources over a network with no survivability properties.

The work in [48] examines different approaches to protect a mesh-based WDM optical network from the failure of network elements (e.g., fiber links and cross connects) that may cause the failure of several optical channels, leading to large data losses. The approaches considered in the paper were based on two survivability paradigms: 1) path protection/restoration and 2) link protection/restoration. The study examined the wavelength capacity requirements, and routing and wavelength assignment of primary and backup paths for path and link protection and proposed distributed protocols for path and link restoration. The study also examined the protection-switching time and the restoration time for each of these schemes, and the susceptibility of these schemes to multiple link failures. They formulated ILPs to determine the capacity utilization for the protection schemes for a given traffic demand. The numerical results obtained for the considered network topology and for random demands (between 10 and 35 connections) indicated that shared-path protection provides significant savings in capacity utilization over dedicated-path and shared-link protection schemes, and dedicated-path protection provides marginal savings in capacity utilization over shared-link protection. On the other hand, if two fiber links fail in the network at the same time, then the number of connections that are dropped under shared-path or dedicated-path protection schemes is more than those that are dropped under shared-link protection. Thus, they observed that, in each protection scheme, there is a tradeoff between the capacity utilization and the susceptibility to multiple fiber failures. They formulated a model of protection-switching times for the different protection schemes, based on a fully distributed control network. Based on their assumptions, they found that when the OXC configuration time is low (10 ns), the protection schemes in increasing order of average protection-switching times (for a random demand of 30 connections) are as follows: 1) shared-link-2.79 ms, 2) dedicated-path-3.49 ms, and 3) shared-path-5.02 ms. When the OXC configuration time is high (10 ms), the protection schemes in increasing order of average protection-switching times (for a random demand of 30 connections) are as follows: 1) dedicated-path-3.49 ms, 3) shared-link-46.45 ms, and 4) shared-path-56.43 ms. They proposed distributed control protocols for path and link restoration. Numerical results obtained from simulation experiments on these protocols indicate that path restoration has better restoration efficiency than link restoration, and link restoration has a faster restoration time than path restoration.

The work in [6] studied resilient multi-fiber wavelength-routed optical networks and investigated the influence on the network performance of the maximum number of wavelengths per fiber, restoration strategies, node functionality, and physical topology. Fiber requirements were analyzed for numerous network topologies both without and with link failure restoration, considering different optical cross-connect (OXC) configurations and terminal functionalities. An integer linear program (ILP) formulation was presented as an exact solution of the routing and wavelength allocation (RWA) problem, with minimal total number of fibers. Lower bounds on minimal total number of fibers were discussed, and heuristic algorithms were proposed. Three restoration strategies were considered and compared in terms of capacity requirement. The study of numerous topologies showed that little benefit was achieved with the introduction of wavelength conversion within the optical cross-connects (OXC's), in the case without restoration and also with restoration if wavelength-agility was provided within the terminals. It was shown that the combination of physical connectivity and network size has great importance in determining the extra capacity required for restoration. It was also demonstrated that mesh wavelength-routed optical networks could achieve considerable capacity saving with respect to restoration approaches currently utilized at higher network layers, such as SONET/SDH rings. In this study, the traffic growth was studied for the optimal design of "future proof" networks, in which case wavelength conversion could improve resource utilization. However, the results demonstrated that the benefit strongly depended on network size, connectivity, and wavelength multiplicity. Also, the results showed that decreasing physical connectivity increases the value of required wavelength conversion resources to obtain significant gain with wavelength-routed optical networks. Network evolution in terms of growth in traffic demand was investigated to study the importance of wavelength conversion within the OXC's as a function of network size and connectivity, traffic demand, and wavelength multiplicity.

The papers reviewed in this section lacked the consideration of multi-fiber environment as a protection scheme, which meant that the fiber assignment for designing a survivable WDM network was not studied. They studied the optical network survivability issues by quantifying the performance degradation of the network when different failures occur or by evaluating the resources availability for restoration after failure, which is not reflecting the correct measure of the optical network survivability.

These deficiencies and limitations in the reviewed research works will be considered as the main motivations for this research. A detail clarifications and definitions for these limitations and deficiencies will be presented and discussed in the following chapter. In this research the multi-fiber network will be proposed as a promising solution for these limitations. So studying multi-fiber optical network performance under the different wavelength conversion scenario is considered one of the main goals of this research. The goals and contributes for this research will also be identified and explained in some detail at the end of the next chapter.

Chapter 3

MOTIVATION AND RESEARCH OBJECTIVES

3.1 Introduction

The flexibility in bandwidth allocation and the bandwidth granularity for heterogeneous services in Internet traffic is determined by the number of wavelengths in the fiber that can be utilized. Trying to add more wavelengths into a single fiber using fine spacing techniques quiver increased the wavelength stability, and increases wavelength crosstalk in a DWDM network. From the point of view of cost, adding more wavelengths to the single-fiber all-optical network environment requires addition of transmitters and receivers operating at unused wavelengths. Increased cost comes from the necessity of redesigning the transceivers (transmitters/receivers) and the switch so they are capable of transmitting, receiving and switching the new added wavelengths. In addition, the new added resources to the switch add more overhead on the control plan.

In this chapter, the limitations associated with the single-fiber optical network environment which are considered as motivations for this research work are identified. Then, the goals and objectives of this research based on deficiencies and limitations in the previous research done in this area, as described in Chapter 2, are described.

3.2 Motivation for th Research

The need for fast and efficient transport techniques has increased as a result of the rapid growth of Internet traffic. Internet traffic growth has been studied by research such as [16, 45, 54, 46]. These studies have found that since 1997, Internet traffic has doubled every year. The capacity of backbone transport networks should match this growth rate. The single-fiber optical network was introduced as a promising solution for accommodating the growth of Internet traffic. However, the single-fiber optical network suffers from many limitations such as poor performance in the absence of the wavelength conversion, limited network scalability and configuration flexibility. The single-fiber wavelength-routed network performance studies that were discussed in Chapter 2 lack the performance study of the network under dynamic traffic nature as in [3] or they are considered as a very limited case as in [9, 57]. The research work in [7] studied the single-fiber wavelength-routed network with and without wavelength conversion. However, it only considered the least cost-effective conversion option; that, is the dedicated conversion scheme. The other wavelength conversion options could give a comparable performance to the dedicated conversion option and they are more cost effective; this was not considered in [7]. The cost of the conversion resources is expected in the near future to remain the dominant overhead optical network implementation cost due to current technology limitations. Therefore, many studies have been investigating the optical network performance when a limited number of costly optical resources are used.

The goal of this research is to develop architectural solutions for the single-fiber optical networks limitations. In this research, solutions for overcoming these limitations are proposed by exploring the benefits of different multi-fiber optical network configurations as well as different wavelength conversion options. Such architectures will overcome the limitations of the single-fiber optical networks in terms of network scalability, configuration flexibility and performance. The use of multi-fiber optical networks combined with a diverse range of wavelength conversion options, in comparison to the single-fiber networks, is proposed as a potential solution for improving optical network utilization performance resulting from the reduced optical path setup conflicts. Different generic optical network models will be developed in order to assess the network scalability and configuration flexibility with the use of multiple fibers complemented with different wavelength conversion options. The use of multiple fibers and wavelength conversion resources should not come at the cost of node complexity, which results in greater overhead in control and management of the optical node. Thus, the control and management of the optical nodes needs to be evaluated in order to estimate the overhead that will be added to the network control plane.

In this research, the deficiencies and limitations of the work that were presented in Chapter 2 will be identified. The blocking rate of connection requests is one of the main performance measures that can be used to evaluate the overall optical network efficiency. The connections blocking rate will be investigated considering different network design parameters such as the number of wavelengths, different conversion schemes and different network environments (single-fiber and multi-fiber networks). The architectural aspects of the network and their effects on connection blocking will be considered and evaluated. Considering different optical cross-connect architectures broadens the scope of this research in terms of the complexity and efficiency of network performance. These performance aspects will be investigated having in mind the mechanism of distributing network resources, such as the different conversion capabilities and their associated control complexity.

In a wavelength-routed network, a connection (lightpath) between the source-destination pair has to be established on top of the WDM multiplexing technology before data transmission starts, and this results in ineffective use of the enormous bandwidth provided by such technology [67]. To overcome this inherent poor utilization of the WDM bandwidth in the wavelength-routed network, optical packet switching (OPS), which allows fast allocation of wavelengths in an on-demand fashion, is introduced. OPS suffers from the packets contention problem that arises when two or more of the incoming packets on the same wavelength intend to leave the switch through the same output port, which results in packet loss and poor network performance. The performance of OPS with a share-per-link conversion scheme will be explored in this study. The performance assessment of the OPS will be conducted in both single-fiber and multi-fiber environments considering different node design parameters such as the number of wavelengths, number of converters, and number of fibers.

All-optical network survivability is an important research and design issue. Deploying multiple fibers between nodes at the physical topology level provides the network with better performance. However, this optical network design suffers from the severity of information loss at the time of failure occurrence. The pervious work on optical network survivability lacked the assessment of the network survivability when a multi-fibers optical network is deployed. Also, the designing issues for the multi-fiber optical networks from the survivability point of view were not considered in the reviewed research works. Moreover, the developed survivability measures for the optical network do not reflect the true value of the network survivability since these assessments are either based on the performance degradation of the network when failures occur or the resources availability for the restoration after the failure.

In summary, the research discussed in Chapter 2 focused on the single-fiber optical network as a promising solution for the backbone transport network bandwidth limitation. However, several aspects need further investigation:

- Dedicated wavelength conversion was adopted as a performance enhancement technique for the single-fiber environment. The dedicated wavelength convertible switch is the most flexible architecture, but it is the least cost efficient wavelength conversion option.
- The hardware complexity associated with share-per-node conversion options has not been investigated and evaluated.
- The limitation imposed by the circuit-switching nature of the wavelength-routed network in the single-fiber environment is a critical performance issue that has not been investigated in detail.
- The effect of employing multiple fibers within the optical packet switch element has not been explored.
- The use of wavelength conversion and multi-fiber in the optical packet switch as contention resolution techniques has not been investigated.
- The huge bandwidth of the single fiber makes the survivability of the all-optical network a crucial design issue that should be investigated further.

3.3 Research Goals

The main goal of this research is to explore the use of multi-fiber links in optical network architectures. The significance of the multi-fiber optical network is that it has the potential to significantly increase the performance of networks due to the availability of multiple copies of the same wavelength per hop. It also provides configuration flexibility to optical networks by allowing the lightpaths to be established on the requested wavelength on any of the parallel fibers. Performance parameters such as blocking rate, throughput, and survivability are investigated by developing accurate and informative performance models based on analytical and simulation techniques.

This work will explore the tradeoffs that a multi-fiber environment offers in order to reduce the need for costly resources such as wavelength converters. In addition, the performance of the optical packet switching network when a multi-fiber environment is utilized as a mechanism for contention resolution will be investigated and evaluated.

The survivability of the multi-fiber environment network will be investigated as well. A hierarchical survivability model that combines both performance degradation of the network at the time of failure and resources availability for the restoration will be developed in this research. The goal of this part of the research is to propose a new design paradigm for multi-fiber optical networks that can enhance network performance as well as network survivability.

3.4 Research Objectives

A main objective of this study is to develop a model that can be used to analyze the multi-fiber optical network performance under different parameters. These models will show how the optical network performance can be improved through introducing multi-fiber links to optical networks.

The performance evaluation of the multi-fiber wavelength routed networks with and without wavelengths conversion is one of the main objectives of this research. In this performance analysis, different switching architectures will be considered. The objectives of this research include performance evaluation for the single-fiber wavelength-routed network when different conversion-sharing mechanisms are employed.

we will also consider the multi-fiber setting for the packet-switching network as a mechanism for contention resolution. The study will involve the investigation of the effects of such a setting on the performance of the optical packet switching element.

We will use a combined approach to quantifying the performance degradation of the network when failures occur and quantifying resources availability for restoration after failure to evaluate the degree of survivability of the network by considering both measures concurrently. Moreover, an analytical model will be developed to quantify the survivability measure.

The objectives of this research are summarized as follows:

1. Evaluate the single-fiber wavelength-routed network performance under different wavelength conversion options.
2. Compare the enhanced performance of the single-fiber wavelength-routed network due to using conversion to multi-fiber wavelength-routed network without deploying any conversion scheme.
3. Explore the multi-fiber wavelength-routed network performance when different conversion scenarios are employed.
4. Develop a general analytical model for the linear network topology that can capture the wavelength-routed network behavior and show the effect of different network parameters such as number of wavelengths, number of hops, number of fibers, offered load and conversion option on the network blocking rate.
5. Investigate the performance gain due to the use of multiple fibers with or without wavelength conversion compared to the single fiber case, when the total number of wavelengths is the same, for traffic that traverses multiple hops between source and destination nodes.

6. Conduct a tradeoff study between the advantages of having of multiple fibers per link and wavelength conversion.
7. Analyze the optical packet switch architecture in which the wavelength converters are shared per output link.
8. Evaluate the OPS performance in both single-fiber and multi-fiber environments.
9. Evaluate wavelength converters utilization in order to identify the efficient use of this costly resources.
10. Develop a composite model that includes (a) system availability analysis to discover the cost due to system downtime and (b) system failure impact analysis to discover the transient performance degradation when failure occurs to evaluate the network survivability.
11. Propose an algorithm to carry out the steady state availability analysis of the network even when the available paths between a pair of nodes are non-disjoint.

The rest of this dissertation will address these objectives in more detail. First, the wavelength-routed network performance objectives (objectives 1 to 6) will be focused on and explored in Chapter 4. Next, a comprehensive investigation for objectives 6 to 10, which assess the optical packet switch performance, will be conducted in Chapter 5. In Chapter 6, the survivability of the optical network (objectives 11 to 12) will be investigated. A general overview of the different simulators that were built in this research will be discussed and explained in Chapter 7.

Chapter 4

PERFORMANCE EVALUATION OF MULTI-FIBER WAVELENGTH-ROUTED NETWORK WITH AND WITHOUT WAVELENGTH CONVERSION

4.1 Introduction

In Chapter 2, we showed the performance limitation of the single-fiber network when the conversion is not employed. The high cost associated with using the dedicated conversion capability is also explained in Chapter 2. The research in this work considers the performance issues for the single-fiber wavelength-routed network using sharing wavelength conversion resources, and the multi-fiber case with and without wavelength conversion capabilities.

We compare single-fiber network performance to multi-fiber network performance under the same network parameters in both cases. This performance comparison will be presented and discussed in Sections 4.6 and Section 4.7 below. Closed-form expressions are derived to show the improvement in throughput under various operational scenarios.

4.2 Analytical Model

The analytical model presented in this section characterizes the end-to-end blocking performance of a wavelength-routed network. The analytical model for a wavelength-routed network without conversion for both single-fiber and multi-fiber

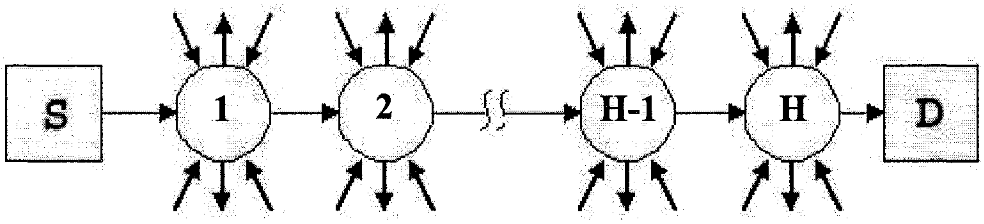

Figure 4.1: A single-fiber multi-hop environment.

cases will be developed first. The performance gain and end-to-end throughput analytical model will also be presented for both cases. Secondly, the analytical model for a wavelength-routed network with wavelength conversion will be developed. Different wavelength conversion scenarios throughout the developing process will be considered.

4.3 Blocking Probability for Paths without Wavelength Conversion

The single-fiber optical network topology shown in Figure 4.1 will be considered in the development process of the analytical model. In this topology, the access node S requests a session to destination node D over a DWDM optical network. We assume that every call requests a capacity of a full wavelength bandwidth. Any particular fiber along the path has the same set of wavelengths and no two sessions on the same fiber use the same wavelength, as determined by the wavelength continuity constraint. An optical path is set from source (access node S) to destination (access node D) by reserving a particular wavelength in all intermediate hops. A connection request will be blocked if a single hop in the route between source-destination pair is supporting W sessions simultaneously, where W is the total number of wavelengths per hop, or when there is no continuous wavelength free on all hops in this route.

The model for a single-fiber multi-hop environment without wavelength conversion in [7] approximates the blocking probability along a path for a multi-hop

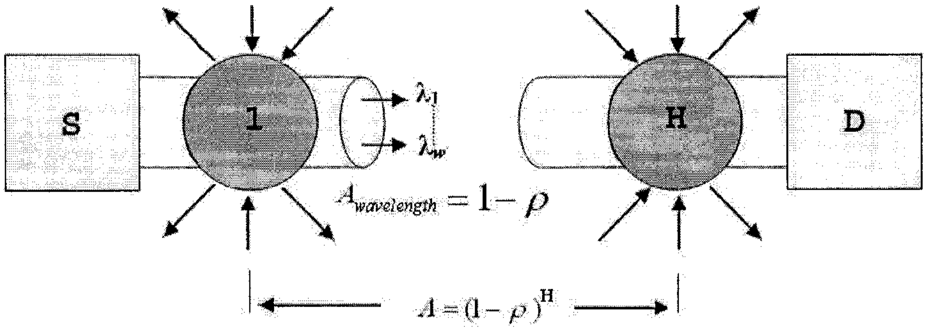

Figure 4.2: End-to-end availability in a single-fiber network.

single fiber between nodes with a uniformly distributed load on all links entering the intermediate nodes by

$$P = (1 - (1 - \rho)^H)^W \tag{4.1}$$

where H denotes the number of hops along a path, $\rho = \frac{\lambda}{\mu}$ is the probability that a wavelength is used in a hop, λ is the arrival rate per wavelength, and μ is the service rate per wavelength on a hop. ρ is also a measure of the load per wavelength. The blocking probability P is the probability that no continuous wavelength is available to set up a lightpath between the source and the destination (i.e., each wavelength is used on at least one of the intermediate hops H).

The availability, A, along the path between the pair of nodes under consideration in the case of a single-fiber multi-hop environment without wavelength conversion is shown in Figure 4.2.

Previous studies indicated that utilization decreases when the number of hops, H, increases [9, 7]. This means that the fiber should have a larger number of wavelengths and network diameter should be kept small to achieve better link utilization in wavelength-routed WDM networks. This setting is not realistic due to the ever-growing network reach and traffic demand, which consumes the usable wavelengths range rapidly. The number of hops that a path traverses is extremely important due to the dramatic change in utilization of the optical link. It was also

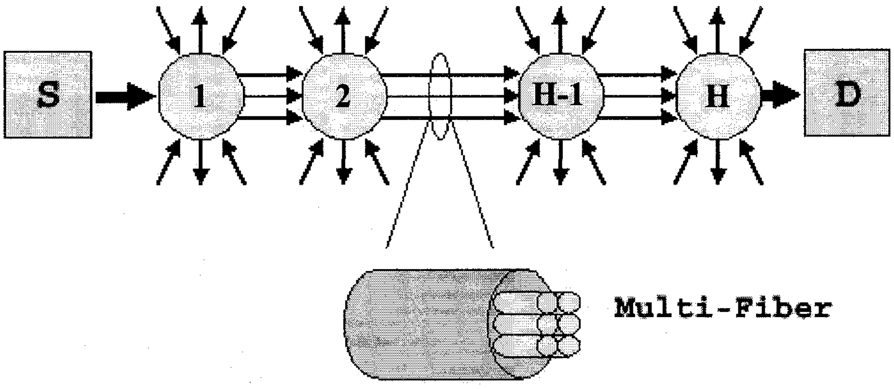

Figure 4.3: Multi-fiber multi-hop environment.

demonstrated in [9, 7] that the blocking probability increases with the offered load as well as the network diameter.

In the multi-fibers multi-hop wavelength-routed network configuration, which is shown in Figure 4.3, every fiber optic link between adjacent nodes along the path consist of F parallel fibers. All the hops have the same number of fibers. The same set of wavelengths is assumed in each fiber.

All requests will be accommodated if two or more sessions, depending on F, request the same wavelength on the same hop due to the availability of the F copies of the same wavelength (one copy per fiber). This overcomes the wavelength exhaustion problem in the C-band window by using multiple versions of the same wavelength set in each fiber. Hence, the effect of the wavelength continuity constraint on the wavelength-routed network performance will be diminished by employing this optical network configuration.

The blocking probability P is the probability that each wavelength in every fiber is used on at least one of the intermediate hops, H, where w is the number of wavelengths per fiber in the multi-fiber case. The availability, A, along the path in the case of the multi-fiber multi-hop configuration without wavelength conversion is shown in Figure 4.4. P, in this case, is given by:

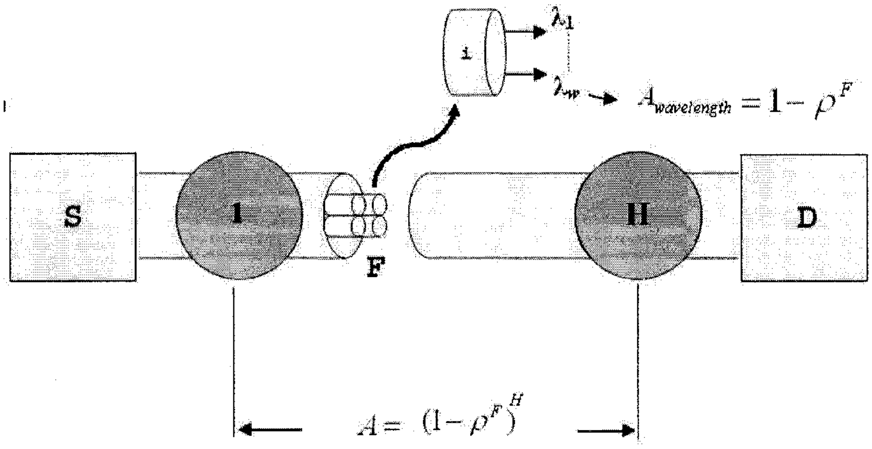

Figure 4.4: End-to-end availability in a multi-fiber network

$$P = (1 - (1 - \rho^F)^H)^w \tag{4.2}$$

The total number of channels in a hop is W, where $W = w \cdot F$.

The blocking probability of any given optical path diminishes due to the reduction in the probability that a certain wavelength is entirely exhausted in all fibers, F. Expression 4.3 is the achievable end-to-end throughput for a given blocking probability with respect to a certain source-destination pair load G in a network that employs multi-fiber multi-hops and has no wavelength conversion option.

$$S = (1 - P) \cdot G \tag{4.3}$$

The change in the blocking probability P due to the load of the connection under consideration, G, is negligible. This is due to the fact that the total capacity of an optical fiber link is typically significantly higher than the load generated by a source-destination pair.

The throughput gain in the case of the multi-fiber multi-hop without conversion option with respect to the single fiber situation increases appreciably due to less blocking probability and higher utilization. This gain is considered a significant improvement that reflects on the end-to-end path blocking probability and overall network performance when the total number of wavelengths is equal in both cases:

$$g = \frac{1-(1-(1-\rho^F)^H)^w}{1-(1-(1-\rho)^H)^w} \tag{4.4}$$

The gain in the achievable throughput comes at the cost of adding multiple fibers between nodes with a similar set of wavelengths in each fiber. It increases as the blocking probability of a given path decreases. In the next section, the gain and the end-to-end achievable utilization due to the use of different conversion scenarios in both single-fiber and multi-fiber environments is investigated.

4.4 Blocking Probability for Paths with Wavelength Conversion

Using wavelength conversion in a wavelength-routed network will improve the overall network performance since the lightpath could utilize different wavelengths on different hops. The network performance improvement in the single-fiber network is higher than the network performance improvement in the multi-fiber network since the connection request blocking rate in the multi-fiber environment is diminished as a result of employing redundant resources. In this section we model different wavelength conversion schemes for both single-fiber and multi-fiber networks. Using wavelength conversion, the blocking of lightpath request occurs if there is no continuous wavelength between the source-destination pair and either there are no wavelength conversion resources available or no other wavelength is available on the output link at the intermediate nodes.

4.4.1 Blocking Probability for Single-Fiber Paths with Partial Wavelength Conversion

The blocking probability P with sharing wavelength conversion resources in the single-fiber case is shown in expression 4.7. In this case, the blocking is considered where the requested wavelength is available and when it is not available but

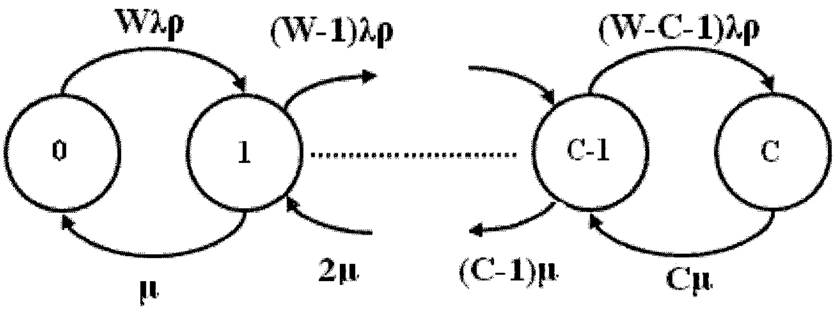

Figure 4.5: M/M/C/C model for the single-fiber wavelength conversion bank configuration.

at least one other wavelength is free and at least one wavelength converter is available. The wavelength conversion bank used in all subsequent sharing conversion scenarios are modeled as a M/M/C/C model shown in Figure 4.5. The probability of zero converters being busy, P_0, and the probability of all converters being busy, P_C, are derived to predict the behavior of the conversion bank, where C denotes to the number of wavelength converters in the wavelength conversion bank with respect to the total number of wavelengths available on the hop W

$$P_0 = \frac{1}{1 + \sum_{k=1}^{C}(\frac{1}{k!}) \cdot (\frac{W!}{(W-k)!}) \cdot \rho^{2 \cdot k}} \tag{4.5}$$

$$P_C = \frac{1}{C!} \cdot \frac{W!}{(W-C)!} \cdot \rho^{2 \cdot C} \cdot P_0 \tag{4.6}$$

$$P = (1 - ((1-\rho) + \rho \cdot (1 - P_c) \cdot (1 - \rho^{W-1}))^{H})^{w} \tag{4.7}$$

and where $C = W \cdot DC$ and DC is a fraction that represents the number of converters with respect to W.

4.4.2 Blocking Probability for Share-per-Link Multi-Fiber Paths

The blocking performance model for a multi-fiber wavelength-routed network employing the share-per-link wavelength conversion scheme is developed in this section. Figure 4.6 shows single node hardware architecture in this network. The

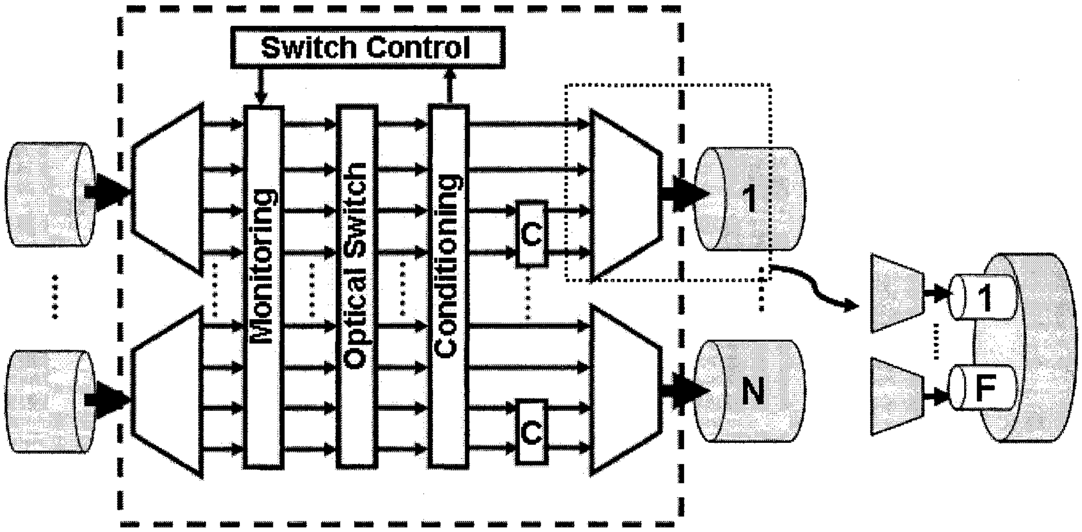

Figure 4.6: Share-per-link wavelength convertible switch.

different components and their functionality are explained in detail in Chapter 2. In this node architecture, a dedicated collection of converters is offered to each outgoing link. The lightpath requests destined to a specific outgoing link can only utilize the conversion resources that are assigned to this outgoing link. One source of the connections blocking is insufficient network resources. A poor conversion resources allocation is a factor that could lead to poor network performance due to a higher rate of blocked traffic. Therefore, the conversion resources allocation algorithm (CRAA) adopted by the switch control plane is an important performance factor. The CRAA that will be considered in this analysis for the share-per-link switch is a narrow search algorithm (NSA). The narrow search algorithm is developed with the hardware complexity of the wavelength convertible switch in mind. It searches for a free converter in the conversion bank that belongs to the corresponding outgoing fiber in the share-per-link wavelength convertible switch. The following is the general description of NSA:

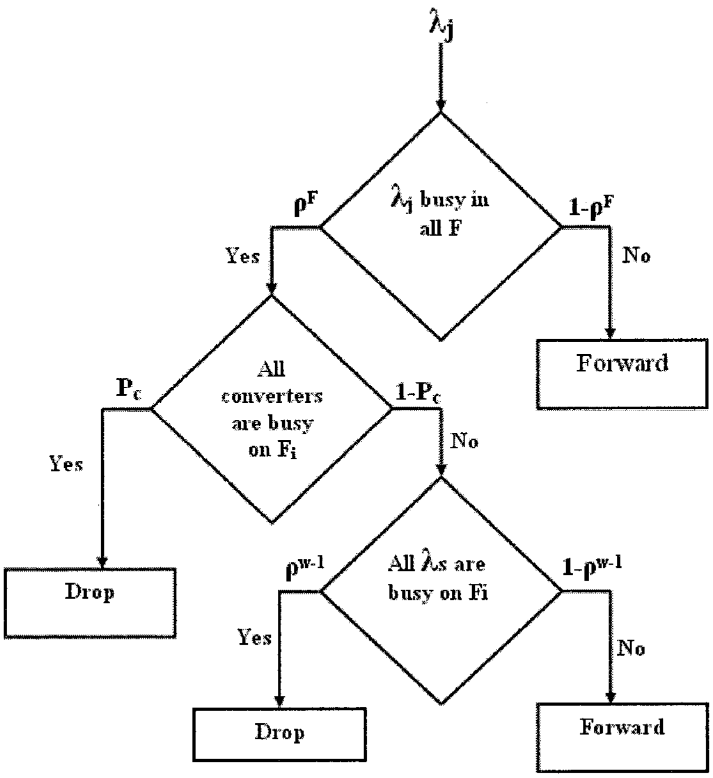

Figure 4.7: Narrow search algorithm flow chart.

1. Detect the connection request and check for the desired output wavelength in all outgoing fibers considering the final destination. In case the desired wavelength is available in any fiber, go to Step 4. The probabilities associated with these transitions are shown in Figure 4.7. For example, the probability the desired channel is not available in any fiber is ρ^F.

2. Check for a free converter in the conversion bank that belongs to the corresponding outgoing fiber. In case a converter is not available, go to Step 5.

3. Check for any other wavelength in the outgoing fiber to use the available converter with. In case a wavelength is not available, go to Step 5.

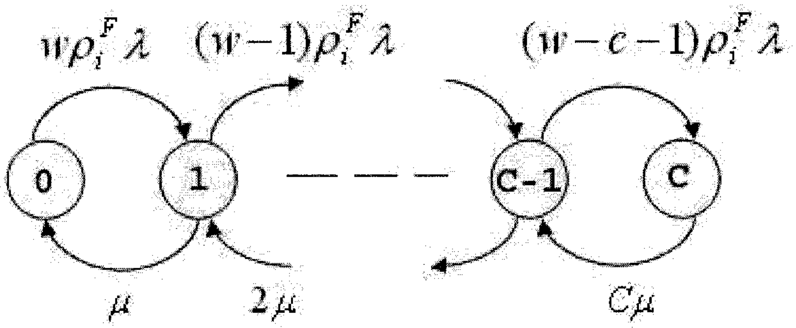

Figure 4.8: M/M/C/C model for share-per-link using NSA.

4. Forward the connection request; go back to step 1.
5. Drop the connection request; go back to step 1.

In the model for the share-per-link conversion bank employing the *NSA*, the request for a converter is the arrival rate per wavelength λ given that the same wavelength on hop i is busy in all output fibers, an event with a probability of ρ^F. So, a conversion request rate due to a single wavelength is obtained by $\rho^F \cdot \lambda$ as shown in Figure 4.7, where ρ is the probability of the requested wavelength being busy, and P_c is the probability of a converter bank being busy. Consequently, the transient rate from state j (j converters are busy) to state j+1 is $(w-1) \cdot \rho^F \cdot \lambda$. The probability of zero converters being busy, P_0, and the probability of all converters being busy, P_c, are derived from the M/M/C/C model shown in Figure 4.8 to predict the behavior of the conversion bank. P_0 and P_c, for the share-per-link wavelength convertible switch employing the NSA conversion resources algorithm, are derived as follows:

$$P_0 = \frac{1}{1 + \sum_{k=1}^{c} (\frac{1}{k!}) \cdot (\frac{w!}{(w-k)!}) \cdot \rho^{k \cdot F + k}} \tag{4.8}$$

$$P_c = \frac{1}{c!} \cdot \frac{w!}{(w-c)!} \cdot \rho^{c \cdot F + c} \cdot P_0 \tag{4.9}$$

$$P = (1 - ((1-\rho^F) + \rho^F \cdot (1-P_c) \cdot (1-\rho^{w-1}))^H)^w \tag{4.10}$$

The lower case c denotes the number of converters with respect to the total number of wavelengths available on each fiber w, where $w = \frac{W}{F}$ and $c = w \cdot DC$.

The analytical models for the NSA as well as analytical models for the ISA (inclusive search algorithm) and the OSA (optimized search algorithm) are investigated in [26] to assess the share-per-link convertible switch performance when a different control algorithm is deployed. The research in [26] concludes that the best achievable performance could be reached without the need for increasing the switching node complexity in the share-per-link wavelength convertible switch.

4.4.3 Blocking Probability for Share-per-node Multi-fiber Paths

In this section, a performance model for the multi-fiber wavelength-routed network employing a share-per-node conversion scheme is developed. In the share-per-node wavelength convertible switch, the lightpath request will be forwarded to a special port that leads to the conversion bank, internally in the switch, if the requested wavelength is busy on all F output parallel fibers. Consequently, in this switch model, the request for a converter is the arrival rate per wavelength λ given that the same wavelength is busy in all output fibers with a probability of ρ^F. A conversion request rate due to a single wavelength is obtained by $\rho^F \cdot \lambda$. Accordingly, the transient rate from state i (i converters are busy) to state *i+1* is $(W-1) \cdot \rho^F \cdot \lambda$. The probability of zero converter being busy, P_0, and the probability of all converters being busy, P_C, are derived from the M/M/C/C model shown in Figure 4.9.

$$P_0 = \frac{1}{1 + \sum_{k=1}^{C} (\frac{1}{k!}) \cdot (\frac{W!}{(W-k)!}) \cdot \rho^{k \cdot F + k}} \tag{4.11}$$

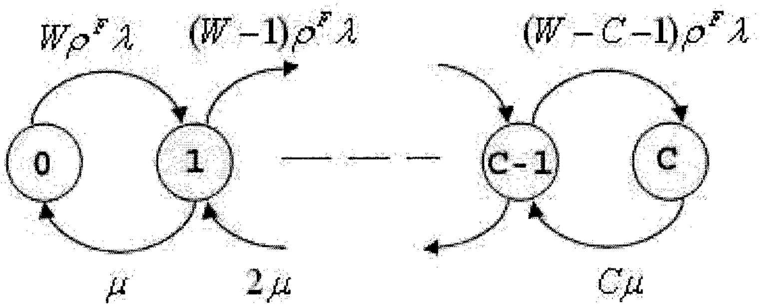

Figure 4.9: M/M/C/C model for the share-per-node conversion bank.

$$P_C = \frac{1}{C!} \cdot \frac{W!}{(W-C)!} \cdot \rho^{C \cdot F + C} \cdot P_0 \tag{4.12}$$

C denotes the number of wavelength converters in the wavelength conversion banks with respect to the total number of wavelengths available on all fibers W, where $W = w \cdot F$ and $C = W \cdot DC$. P_C depends on the number of converters (C), the number of incoming wavelengths, W, and the load per wavelength ρ. The availability, A, along the path in the case of the multi-fiber multi-hop with share-per-node conversion is shown in Figure 4.10. The blocking probability obtained for the limited wavelength conversion case employing share-per-node WCS is given by:

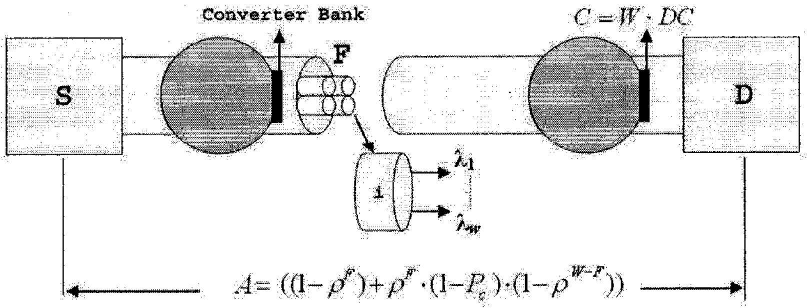

Figure 4.10: End-to-end availability for multi-fiber multi-hop with share-per-node.

$$P = (1 - ((1-\rho^F) + \rho^F \cdot (1-P_C) \cdot (1-\rho^{W-F}))^H)^w \tag{4.13}$$

This architecture shows an improvement in performance compared to the network performance obtained by using a share-per link wavelength convertible switch when the number of total converters is the same in both cases; this result will be demonstrated and explained in Section 4.7. This performance improvement comes at the cost of increasing switch hardware complexity, as explained in Chapter 2.

4.4.4 Blocking Probability for Dedicated Wavelength Conversion Paths

When dedicated wavelength conversion is considered, the blocking probability, shown in expression 4.14, for the multi-fiber multi-hop, converges with the blocking probability of the single-fiber full conversion case.

$$P = (1 - (1 - \rho^{W})^{H})^{w} \tag{4.14}$$

The utilization gain obtained in this case comes at the cost of adding wavelength conversion. The gain increases as the blocking probability of a given path decreases due to the use of wavelength conversion. The performance improvement aspects and the effects of different parameters are presented in Sections 4.6 and 4.7.

4.5 End-to-End Blocking Probability General Analytical Model

The end-to-end blocking probability model could be written in general form as follows:

$$P = (1 - ((1 - \rho^{F}) + \rho^{F} \cdot (1 - P_C) \cdot (1 - \rho^{W-F}))^{H})^{w} \tag{4.15}$$

Any of the above scenarios, either with or without wavelength conversion, could be driven from this general form by setting the correct parameters. For example, to get single-fiber without conversion end-to-end blocking probability, the following parameters are set as $F = 1$ and $P_C = 1$ in the general form to get Barry's model [7]: P

$$P = (1 - (1 - \rho)^{H})^{W} \tag{4.16}$$

This shows that Barry's model in [7] is a special case of the general model developed in this research. The performance gain is calculated as the ratio between the blocking probabilities for multi-fiber partial conversion case, which is considered as the general form, and the dedicated wavelength convertible switch (multi-fiber full conversion option) case.

$$g = \frac{1-(1-((1-\rho^F)+\rho^F\cdot(1-P_C)\cdot(1-\rho^{W-F}))^H)^w}{1-(1-(1-\rho^W)^H)^w} \tag{4.17}$$

Table 4.1 summarizes all the different scenarios for the different switches considered in this research according to their blocking probabilities.

Switch	Blocking Probability P
Multi-fiber share-per node	$(1-((1-\rho^F)+\rho^F\cdot(1-P_C)\cdot(1-\rho^{W-F}))^H)^w$
Multi-fiber share-per-link	$(1-((1-\rho^F)+\rho^F\cdot(1-P_c)\cdot(1-\rho^{w-1}))^H)^w$
Single-fiber partial conversion	$(1-((1-\rho)+\rho\cdot(1-P_C)\cdot(1-\rho^{W-1}))^H)^w$
Dedicate conversion	$(1-(1-\rho^W)^H)^w$
Multi-fiber no conversion	$(1-(1-\rho^F)^H)^w$
Single-fiber no conversion	$(1-(1-\rho)^H)^W$

Table 4.1: Switch blocking probability.

In the following sections, the previously developed models are compared and evaluated. Also, we discuss the performance improvements in the multi-fiber environment over the single-fiber case considering all network parameters in the absence of wavelength conversion. Different cost-effective conversion options are evaluated and compared to the dedicated wavelength conversion. The results show that using a multi-fiber without wavelength conversion mechanism enhancs the network performance and also increases the network reach. Using a limited number of converters in the multi-fiber environment could give the best performance that could be achieved. These results are discussed in detail in the following sections.

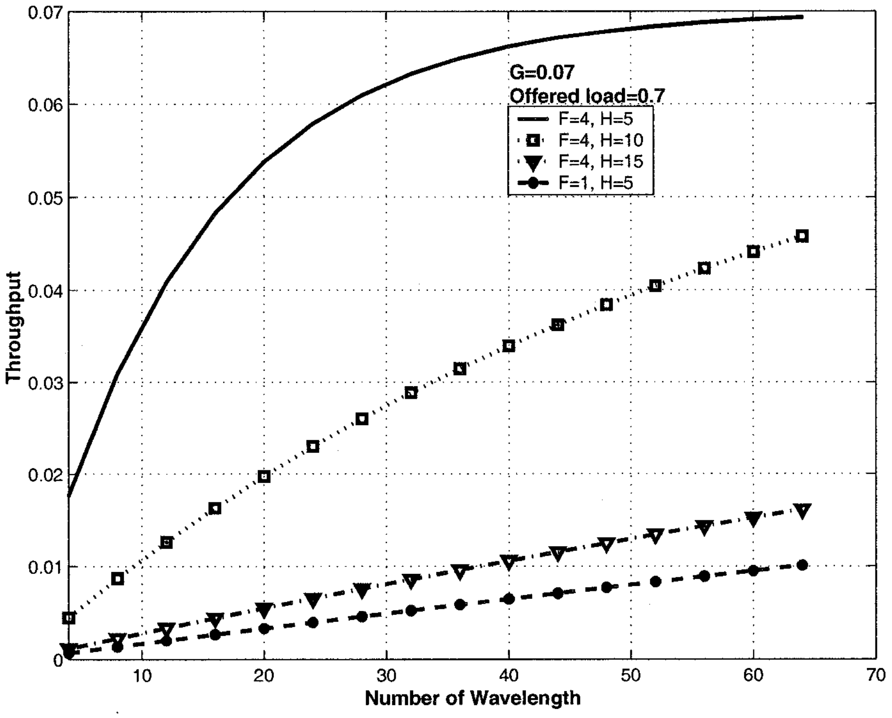

Figure 4.11: Throughput vs. number of wavelengths

4.6 Results for Paths without Wavelength Conversion

In Figure 4.11, the source-destination throughput (expression 4.3) with respect to G, which is significantly less than the capacity of the link, is plotted as a function of the number of wavelengths in the multi-fiber case for different path lengths (intermediate hops) for $\rho = 0.7$. It shows that utilization decreases when the number of hops, H, increases for the same number of wavelengths in the fiber, w. Throughput always increases along with the number of wavelengths, which can provide additional network capacity to handle the traffic. The throughput improvement due to the use of multi-fiber in a network with a diameter of 5 hops is considerable with respect to the performance of the single-fiber no-conversion case. In the multi-fiber case, we have more freedom with the network diameter as opposed to the single fiber case due to the improvement in the throughput resulting from the ability to accommodate multiple requests for the same wavelength in

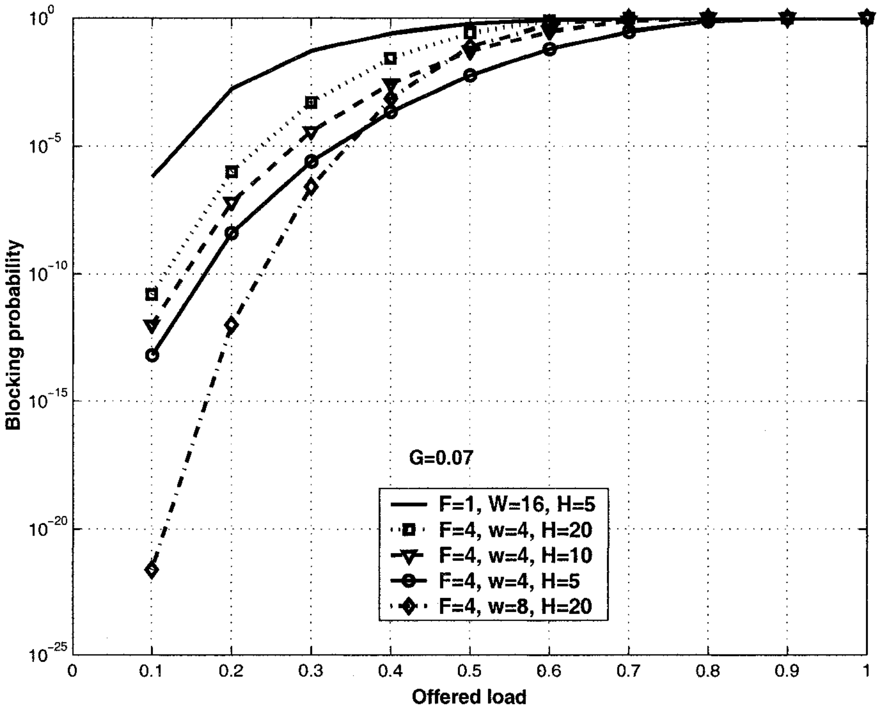

Figure 4.12: Blocking probability vs. offered load

different fibers in the same hop. Using multiple fibers compensates for the throughput degradation caused by the number of hops that a path traverses. The figure shows a considerable improvement in throughput for similar network parameters, where the number of wavelengths is divided among multiple fibers used in links between nodes.

In Figure 4.12, blocking probability is plotted as a function of offered load for the multi-fiber case with 4 fibers and 4 wavelengths per fiber for different path lengths (intermediate hops). It offers almost zero blocking until the offered load reaches approximately 40% in all multi-fiber multi-hop cases studied. An example of a 5-hop network diameter shows a dramatic improvement of blocking performance in the multi-fiber case compared to the single-fiber case. It should be noted that increasing of the number of wavelengths in the fiber will further improve the optical path blocking probability, which in turn improves the throughput perfor-

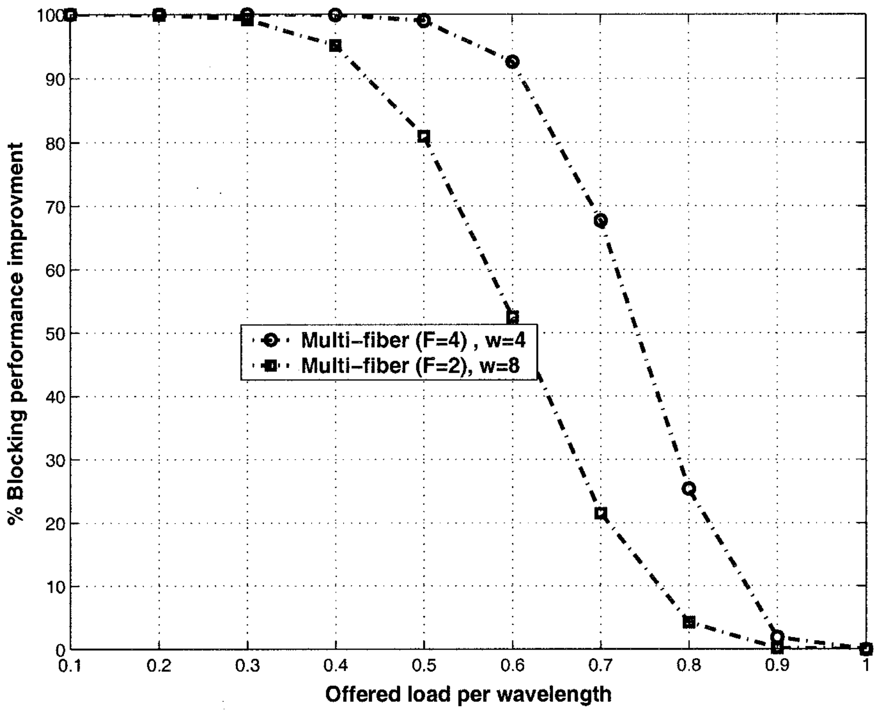

Figure 4.13: Percentage of blocking probability improvment vs. offered load.

mance of the network and increases the possible reach of the network by increasing the deployable network diameter.

The percentage improvement in the network blocking performance versus offered load is plotted in Figure 4.13. The blocking performance improvement is defined as a percentage ratio of the blocking probability difference between single-fiber and multi-fibers cases with respect to the single-fiber blocking probability. The figure shows a significant reduction in the blocking probability if the multiple fibers are utilized even though the same number of wavelengths is used in both cases. The blocking performance improvement could achieve 100% if the number just increased by 1 and the network operated at a fairly low load. This blocking performance could be maintained when the load per wavelength per hop is moderate by increasing the number of fibers. Due to the high blocking rate for both the

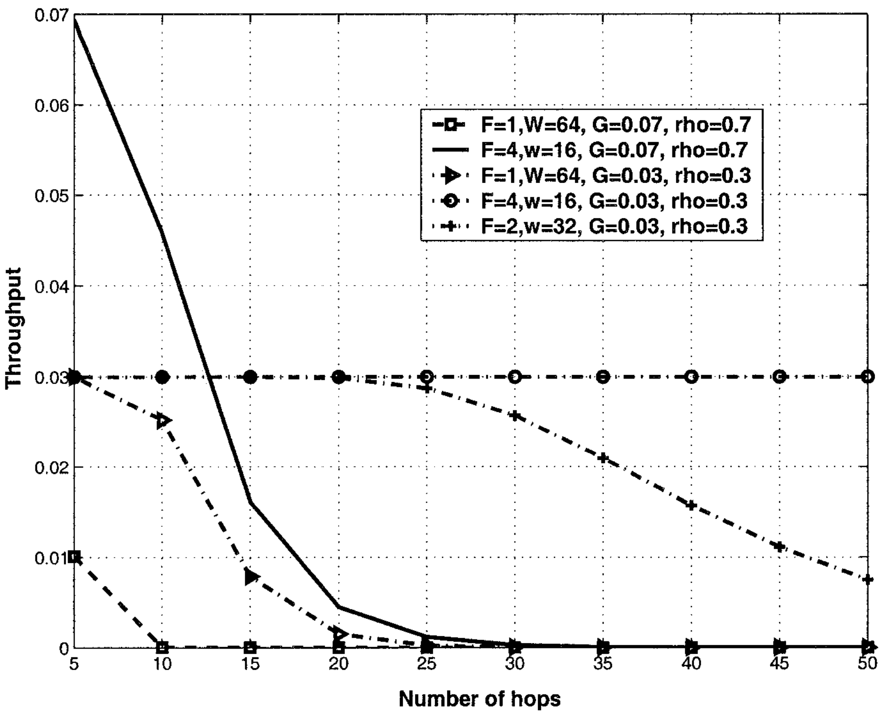

Figure 4.14: Throughput vs. number of hops.

single-fiber and multi-fiber cases, when the network is fully loaded, the blocking performance improvement is negligible.

Figure 4.14 demonstrates the imperative role of network diameter on determining throughput. In this figure, the network throughput is plotted as a function of the network diameter for different numbers of fibers per hop while the total number of wavelengths per hop is kept constants. This figure clearly shows the advantage of using multi-fiber between nodes over the single-fiber case. For instance, when the lightpath route's length is less than 5 hops, the best throughput for the single-fiber environment case could be achieved. This throughput decreases with the increasing of the lightpath length. The network throughput is demolished if the network diameter (lightpath route's length) reaches 10 hops when the load per wavelength per hop is moderately high ($\rho = 0.7$). The figure also shows that the network throughput for the multi-fiber case when 4 fibers are employed per

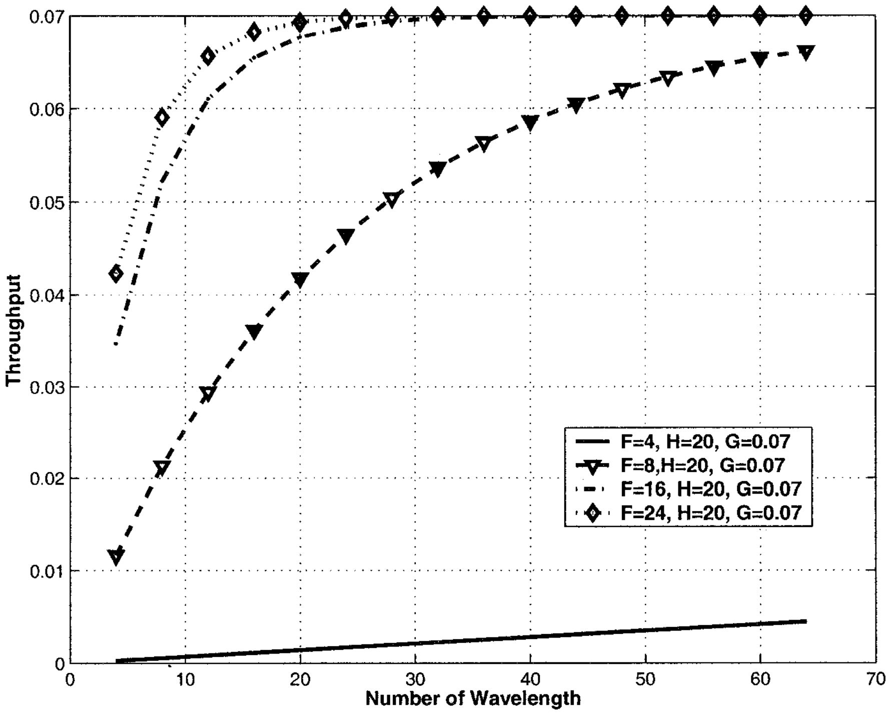

Figure 4.15: Throughput vs. number of wavelengths for multi-fiber multi-multi-hop case

hop will equal the 5-hop counts in the single-fiber environment when the lightpath route's length is equal to 18 hop counts. In addition, if the load per wavelength per hop is fairly low, the network throughput in the multi-fiber environment with 4 fibers per hop will not be affected by the network diameter changes. The figure shows that even though the network diameter has doubled, the multi-fiber case improves the throughput for the same values of network parameters.

In Figure 4.15, throughput is plotted as a function of number of wavelengths in the multi-fiber case for the same path length. The graph confirms that the increase of the number of fibers in the intermediate hops will improve the throughput performance of the network. It also shows that throughput increases with the number of fibers in each link and the number of wavelengths carried in each fiber.

In Figure 4.16, throughput is plotted as a function of offered load in the multi-fiber multi-hop case for different networks with different parameters. Using

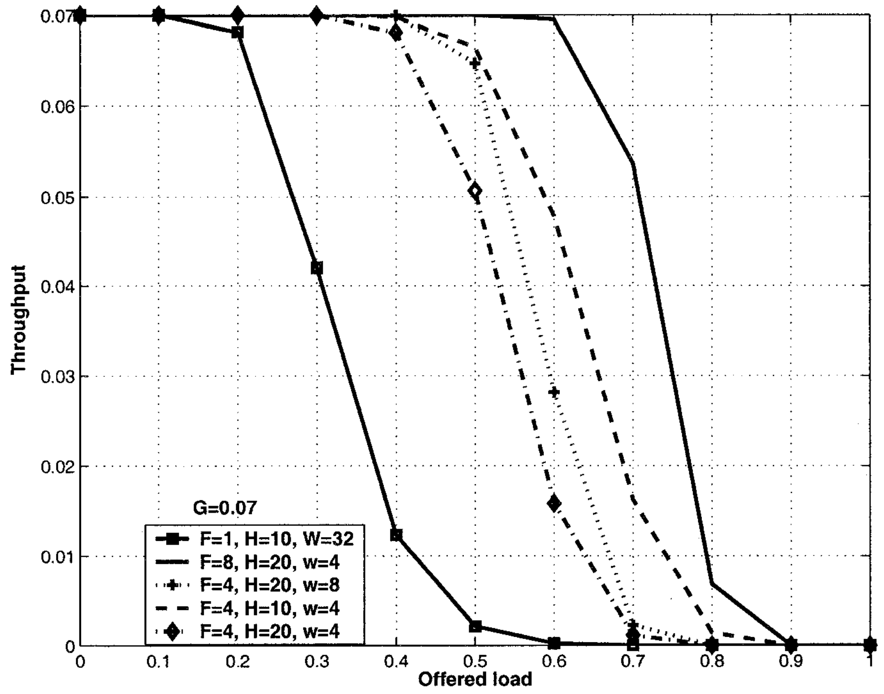

Figure 4.16: Throughput vs. offered load for different networks.

a network with parameters *w=4*, *H=20*, and *F=4* as a reference, we notice that the increase of the number of wavelengths ,*w*, will increase the throughput. It is also seen that the decrease of the number of hops, *H*, that a path traverses increases the throughput significantly at the cost of network diameter and network reach, which is not practical for the ever-growing optical domain.

The network with parameters *w=4*, *H=20*, and *F=8* confirms the main objective of this study. It shows that the increase of the number of fibers, *F*, in links deployed between nodes has the dominating effect of improving the throughput performance, reducing the path-blocking probability, and increasing the network reach by increasing the diameter. The figure also shows that the increase of offered load reinforces the negative effects of network diameter on throughput performance and, consequently, the end-to-end blocking probability.

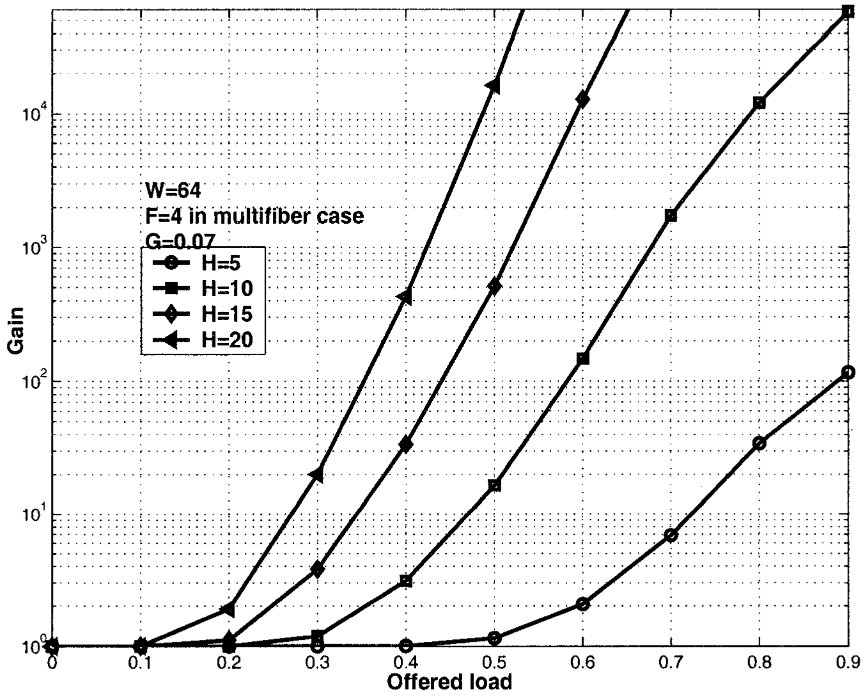

Figure 4.17: Gain vs. offered load.

As a measure of the benefit of a multi-fiber multi-hop environment, the gain was defined as the increase in achievable throughput with respect to the single fiber case. Figure 4.17 shows that the multi-fiber system offers better performance in large networks due to significantly higher gain with reference to the single-fiber system given that other network parameters are fixed. It also shows a tremendous gain that comes at the cost of adding more fibers between nodes when the offered load increases. It is economically more feasible to use a single-fiber link where the expected offered load is low because the fact that the increase in the number of fibers would not increase the throughput performance. This means that the blocking probability would be comparable quantitatively when the expected offered load is low.

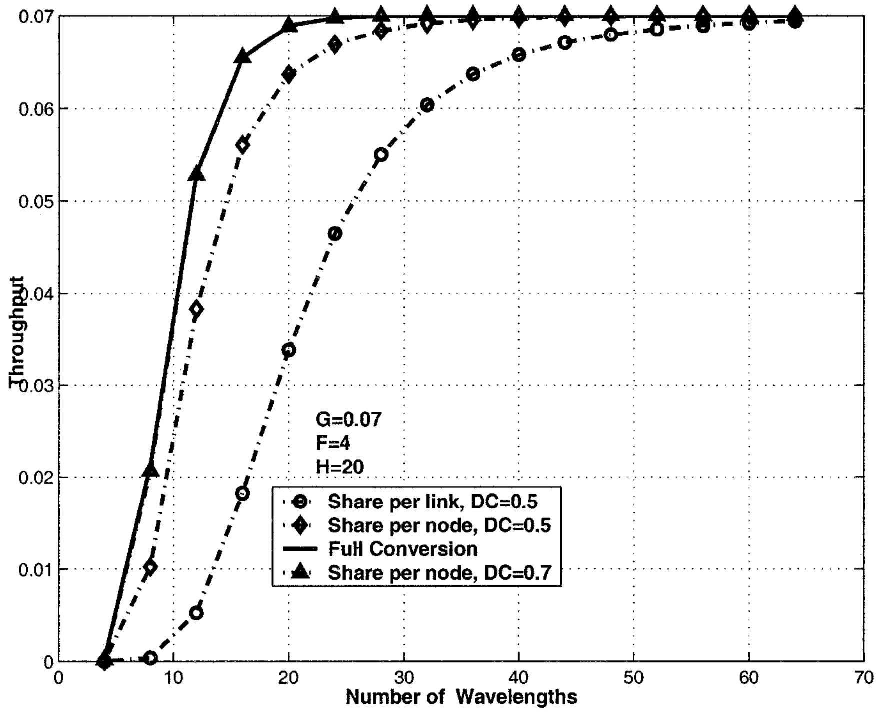

Figure 4.18: Throughput vs. number of wavelengths.

4.7 Results for Paths with Wavelength Conversion Option

In this section, we discuss results showing the performance improvements in the multi-fiber environment with various wavelength conversion options considering all network parameters. In Figure 4.18, the throughput is plotted as a function of the number of wavelengths in the multi-fiber case with 50% conversion for different wavelength convertible switch architectures and a single-path network of 20 hops. The throughput improvement due to the use of wavelength conversion depends on the degree of conversion. The use of wavelength conversion increases the degree of freedom to increase network diameter due to the improvement in throughput resulting from the ability to minimize wavelength contention. A degree of 70% conversion in the share-per node wavelength convertible switch case resulted in a performance that is very comparable quantitatively to the dedicated wavelength

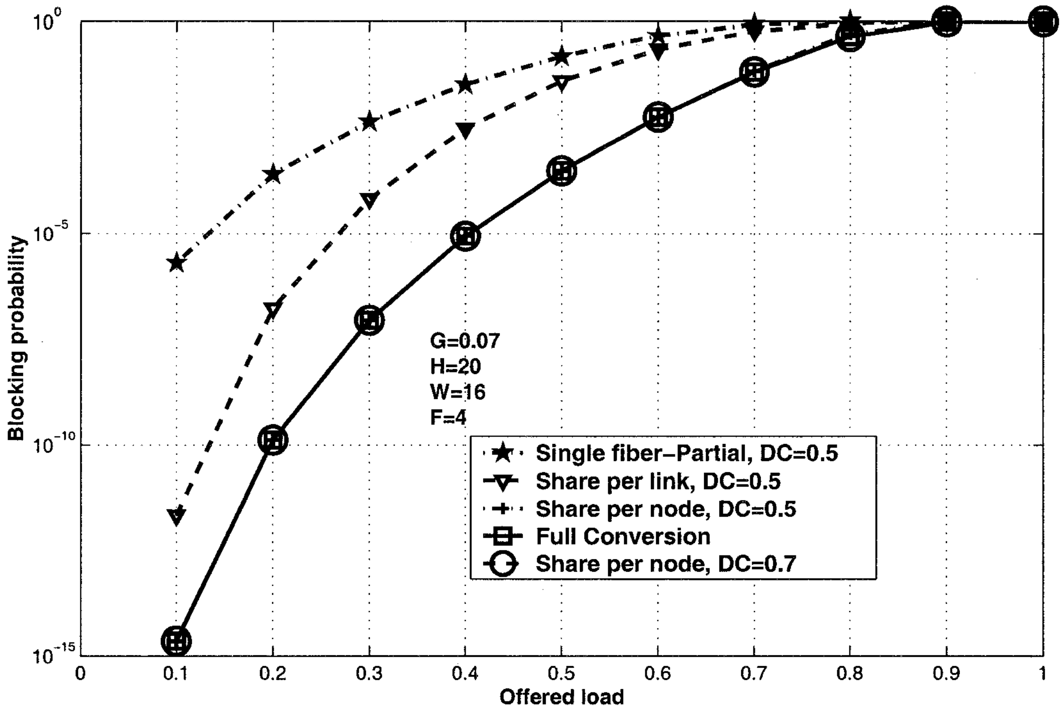

Figure 4.19: Blocking probability vs. offered load.

convertible switch. This indicates that the increase of conversion degree at the higher wavelength count does not result in a significantly higher performance. It also shows that a smaller degree of conversion can yield the same performance as full-range wavelength conversion due to low network throughput where some wavelength converters remain unused.

In Figure 4.19, the blocking probability is plotted as a function of offered load for different wavelength convertible switch architectures. In the case of shared converter architectures, it verifies that the efficiency of wavelength converters increases with the blocking performance and utilization of the system. It should be noted that, at lower rates of offered load, the blocking performance of all cases considered is comparable. The region of comparable blocking performance increases with the increase of degree of conversion. This introduces the issue of cost effectiveness when considering different switching architectures, conversion options, and hop

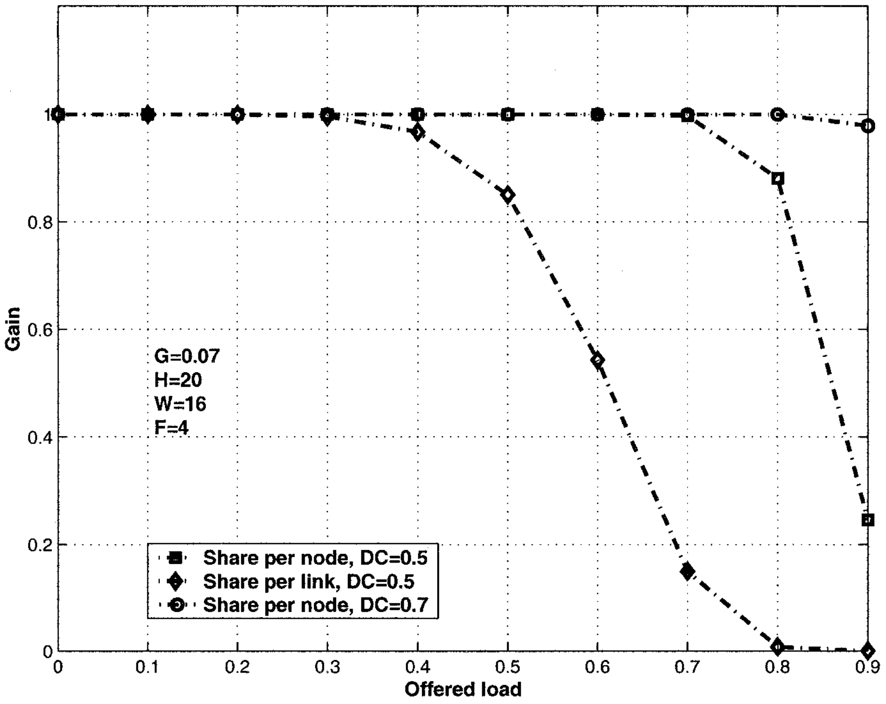

Figure 4.20: Gain vs. offered load.

configurations. It is shown that the effects of conversion are mainly reliant on path length and traffic pattern on the link.

As a measure of the benefit of wavelength conversion in the multi-fiber multi-hop environment, the gain was defined as the increase in achievable throughput with respect to the dedicated conversion case. Figure 4.20 shows better gain performance at higher rates of offered load when using switches that employ a wavelength converter with a sharing mechanism. It is more economically feasible to use wavelength convertible switches with conversion sharing and lower hardware complexity to reduce wavelength blocking probabilities and obtain better throughput gain. Results also prove that share-per-node is the most efficient and that its performance reaches dedicated conversion at 70% wavelength conversion. This means that the blocking probability would be comparable quantitatively to dedicated conversion due to the optimized use of conversion resources. Using different

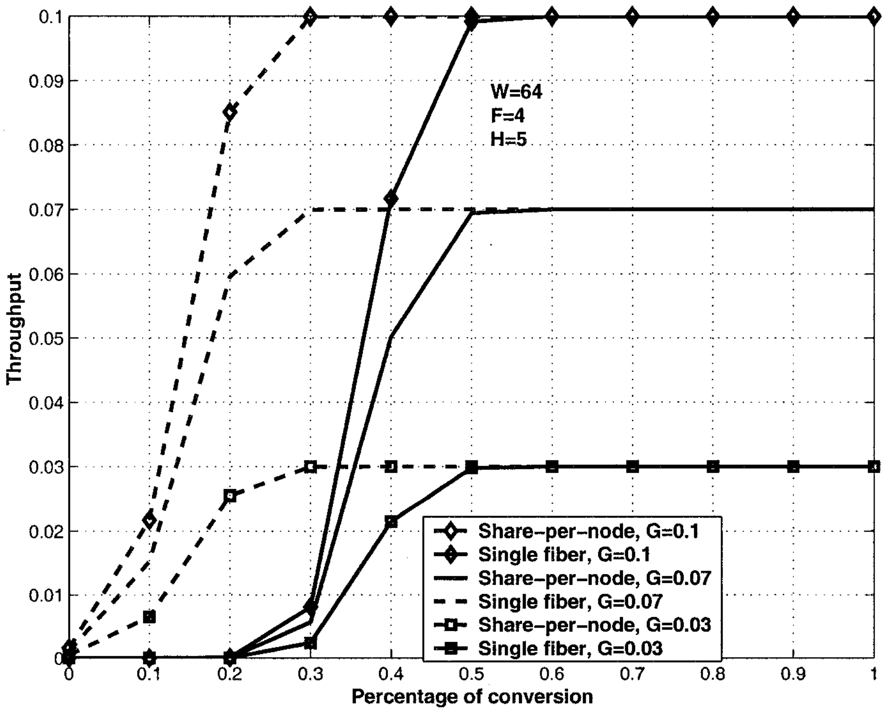

Figure 4.21: Throughput vs. percentage of conversion.

conversion resources allocation algorithms (CRAA), which dynamically accommodate more conversion requests, for a share-per-link wavelength convertible switch improves its performance. It also demonstrated that the gain increases as the number of converters increases and saturates as the number of converters because greater than some threshold and less than the total number of channels on the link, which implies that a limited number of converters under certain offered loads is sufficient to provide good performance.

In Figure 4.21, the end-to-end throughput for a source-destination pair is plotted as a function of conversion percentage (*DC*). As expected, the throughput increases as the number of converters increases. However, after an initial steep increase, the curves generally tend to flatten as the number of converters increases. This behavior is consistent with the results of earlier observations.

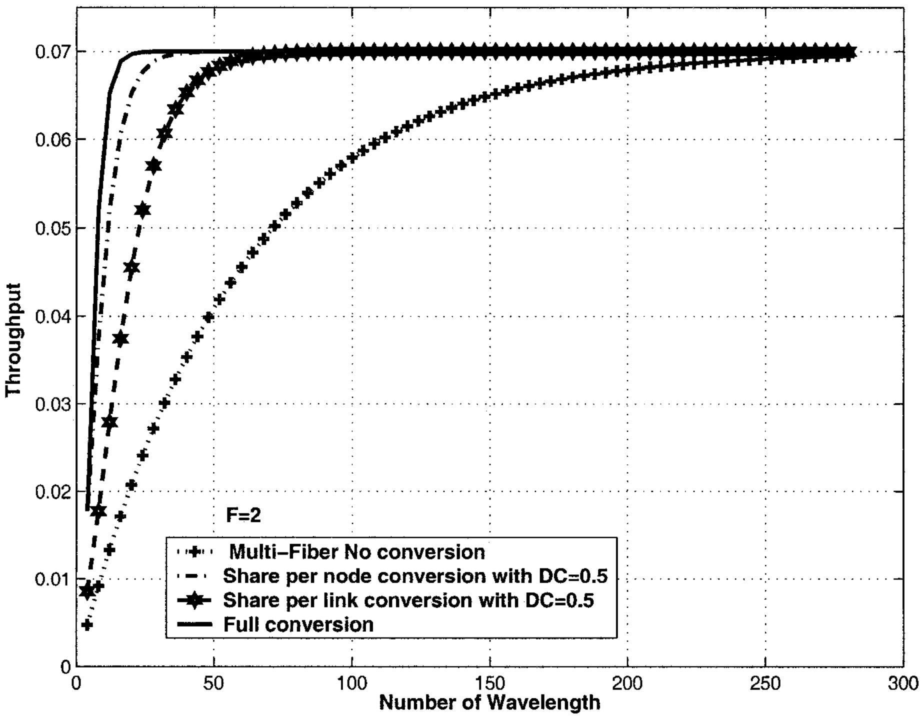

Figure 4.22: Throughput vs. number of wavelengths.

In Figure 4.22, the end-to-end throughput for a source-destination pair is plotted as a function of number of wavelengths. As expected, the throughput increases as the number of wavelengths increases. This behavior is consistent with the results of earlier observations. It can be noticed that the throughput of the multi-fiber without employing any conversion scheme reaches that of the dedicated wavelength conversion at a higher count of wavelength due to the abundance of bandwidth. At lower wavelength counts, it offers around 60% of the throughput of the dedicated conversion option (the best achievable throughput). This figure also showes that using as few as 2 fibers in a multi-fiber environment could improve the network performance considerably.

Chapter 5

PERFORMANCE OF SLOTTED OPTICAL PACKET SWITCHES

5.1 Introduction

Recently, Optical Packet Switching (OPS) has been introduced as a solution for large electronic packet switches as well as wavelength-routed networks. The OPS combines the massive bandwidth of the fiber cable provided by WDM with high-speed operation switching elements to allow fast allocation of wavelengths in an on-demand fashion. OPS suffers from the packets contention problem, which arises when two or more incoming packets on the same wavelength intend to leave the switch through the same output port, which results in packet loss and poor network performance.

Analytical and simulation performance models for the multi-fiber OPS with share-per-output link conversion resources are developed through this chapter. The two main switch design parameters (cost and performance) are taken into account during the development stages of these models. Thus, multi-fiber switch configuration is considered as the contention resolution paradigm in the developed models. Utilizing multiple fibers on the output link enhances the OPS performance due to the availability of multiple copies of the same wavelength. This configuration allows multiple packets requesting the same wavelength on the same output port to be switched and forwarded simultaneously, which results in packet loss performance enhancement. Therefore, the need for costly contention-resolving resources will be reduced. Adding a limited number of wavelength conversion resources to the multi-fiber OPS configuration enhances the OPS performance further and leads to the highest performance that could be achieved. The wavelength conversion resources-sharing schemes reduce the hardware and control costs introduced by using conversion. An analytical model for multi-fiber OPS configuration with share-per-output link conversion resources that operate in a slotted time fashion is developed in the following section.

5.2 Optical Packet Switching Networks

Wavelength Division Multiplexing (WDM) technology allow networks to scale in capacity to meet the unprecedented rate of Internet traffic growth. The tremendous transmission capacity provided by the WDM optical network motivated the network providers to adopt the optical network as a solution for the current backbone transport network bandwidth limitation. Nowadays, the optical network is implemented as an optical circuit-switched network. This optical network implementation results in inefficient use of the wavelength bandwidth due to the large time scale required to set up and tear down the light path [23]. Therefore, the implementation of the optical network as a packet-switched network has recently been focused on.

In general, optical packet-switched networks can be classified as slotted and unslotted networks. The optical packet switched network consists of a number of optical packet switches that are interconnected by fiber optic links, as shown in Figure 5.1. This chapter will focus on the slotted optical packet switched networks.

Packets can arrive at the input ports of each node at different times. The slotted optical packet-switch fabric can only be reconfigured at discrete times; therefore, all the packets arriving at the inputs of the switch are aligned before they enter the switch fabric. Thus, a switching node with a very fast headers

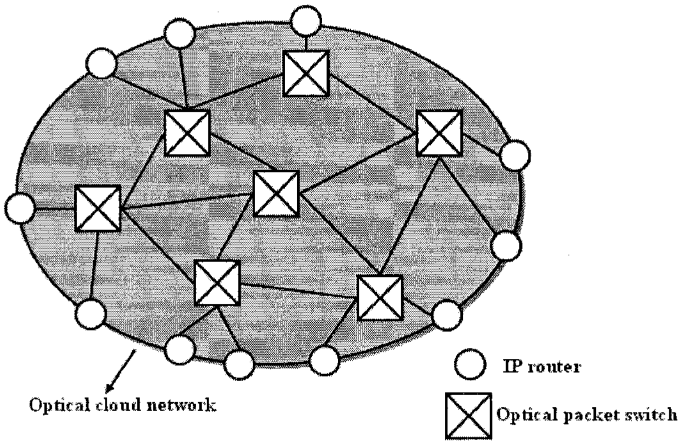

Figure 5.1: Optical packet pwitching network.

recognition and switching fabric configuration is required for the slotted optical packet switched network in order to avoid the packets' buffering at the input. The optical packet switching node switches the data in the packet-by-packet base at the optical network layer. A network built on top of an optical packet switched network will have to not only provide connectionless services by switching the data traffic in the packet by packet fashion, but must also support higher-quality connection-oriented services. This will require the network to have differentiated service qualities on the packet level.

With today's technology, the packet received by the optical packet switch splits into two different optical signals (header and payload) [11]. Since the switch control and routing logic inside the switch are performed electronically, the header has to be converted to an electrical signal to be processed and to identify the packet destination. Therefore, the switching elements inside the switch can be set up properly. The processing speed of the header has to be at least equal to the data transmission rate in order to avoid delaying the payload. So, the necessity for a very high-speed control plane and ultra-fast switching elements is the most challenging problem for optical packet networks [63]. In the following section, an analytical model for assessing the blocking performance of the optical packet switch will be developed. The results generated by the developed analytical model will be presented and discussed in detail in the results section in this chapter.

5.3 Analytical Model

In this section, we develop a performance model for multi-fiber share-per-link OPS, which operates in a slotted mode. A symmetric OPS employing share-per-link conversion resources with N inputs, coming from different sources and destined to N outputs links, each consisting of F parallel fibers, as shown in Figure 5.2, is considered. In this section *port* and *link* are used interchangeably. The switch supports a *WDM* signal with w wavelengths per fiber. A total of wF output wavelengths share a conversion bank of C converters. The incoming packets are synchronized at the inputs before they are processed at the beginning of every time slot. The packet length is assumed to be one time slot.

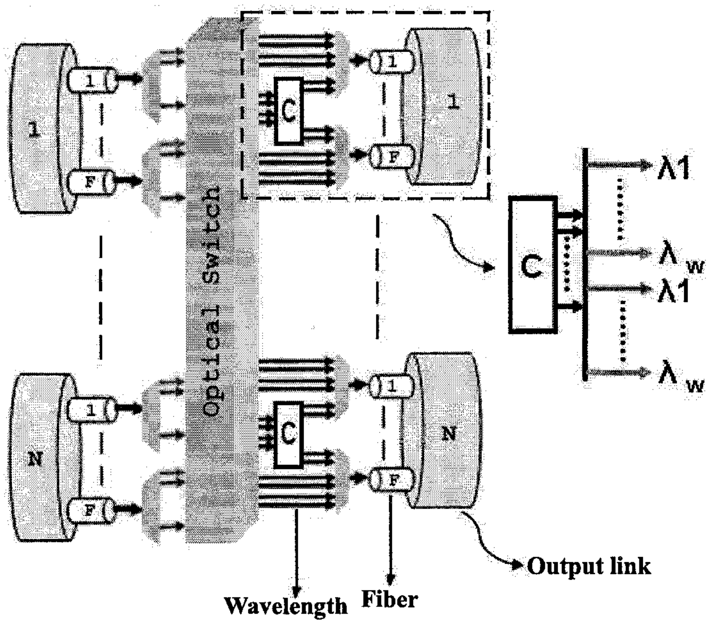

Figure 5.2: Multi-fiber synchronous OPS employing share-per-link conversion resources.

Traffic is considered equally likely to be routed out of the switch through any output port. This symmetrical assumption of the destination with respect to outputs links allows us to build the model for one output port. We consider one output link and its corresponding incoming traffic requests from the input side and study the performance behavior of that port and generalize its performance to all other ports in the switch, since the condition of symmetry holds in this architecture. Figure 5.3 is a schematic for the OPS representing the random variables that will be defined in the model development stages. Let X_i be a random variable representing the number of packets present at an output on λ_i in a given time slot as shown in Figure 5.3. There are NF fibers on the input side of the switch. The switch considered in this analysis has N input ports each with F parallel fibers, and hence NF incoming channels each at λ_i, so the maximum number of packets that could be received at an output port is $N \cdot F$. Consequently, the maximum value that could be taken by X_i is NF.

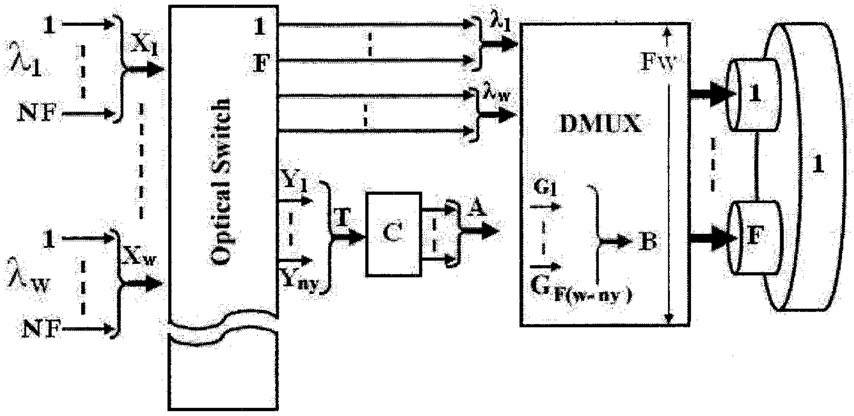

Figure 5.3: Multi-fiber synchronous OPS employing share-per-link conversion resources.

A packet is expected to switch by optical switch, as shown in Figure 5.3, from the input port it arrived on to the output port, which is connected to the destination. A packet competes for the same wavelength it arrived on in any of the outgoing fibers on the requested output port. This means that all packets arriving at the inputs on λ_i that are destinated to the same output port are competing for the same wavelength, λ_i, on the output link under consideration.

In contrast, the minimum number of packets that could be received at the output port under consideration is zero, which means that there is no arrival on λ_i at the inputs of the switch or none of the arrived packets are destined to that specific output link. Thus, the minimum value of X_i is *zero*. The probability that a packet arrives at an input on a given wavelength (λ_i) in a given time slot is denoted by ρ, which corresponds to the normalized offered load per wavelength. Load is assumed to be equally distributed among wavelengths. Packet arrivals per wavelength at different inputs are assumed to be independent of each other. Also, the number of packet arrivals on different wavelengths is identically and independently distributed; $\{X_i\}_{i=0}^{w}$ is *iid*. The number of packets competing for an output per wavelength in a given time slot and destined to the output link under consideration follows the binomial distribution, i.e.,

$$P(X_i = x) = \binom{N \cdot F}{x} \cdot \left(\frac{\rho}{N}\right)^x \cdot \left(1 - \frac{\rho}{N}\right)^{N \cdot F - x} \tag{5.1}$$

where $x = 0 \cdots NF$

For a given time slot, if the number of arrivals per wavelength that are destined to the considered output, is more than F packets, contention among them will occur since the total number of the requested wavelength on the considered output port is F one per every outgoing fiber. Thus, F packets out of the contending ones get switched by the optical switch to the output port using the wavelength of interest (λ_i) and leave the switch directly, as shown in Figure 5.3 note that *direct* refers to packets that are directly routed from inputs to outputs without going through converters. The rest of these packets are forwarded by the optical switch to special ports that lead to the conversion bank that belongs to the output port under consideration.

Let Y_i be the random variable representing the number of packets that are forwarded to the conversion bank on λ_i due to arriving more than F packets on this wavelength. When $X_i \leq F$, the number of packets that will be forwarded to the conversion bank from ρ_i equals zero ($Y_i = 0$) because every received packet at the input side of the switch on λ_i destined to the same output port can utilize a copy of λ_i and leave the switch without need for a conversion. Yet, the number of packets that will be forwarded to the conversion bank, Y_i, due to $X_i > F$ (which means there are more than F packets arriving at the inputs on λ_i destined to the same output port), varies between *one* and $F(N-1)$. It is equal to *one* when the total number of arrivals per λ_i, is equal to *F+1* ($X_i = F+1$). This means F packets will be switched by the optical switch directly to the requested output port on the same wavelength and the left packet will be switched to special ports that lead to the conversion banks belong to the requested output port.

If all arrivals at all inputs on λ_i are destined to the same output link, $N \cdot F$ packets, F packets will leave the switch directly by utilizing the F λ_is and the rest, $N \cdot F - F$, will be forwarded to the conversion bank. Thus, Y_i will equal $N \cdot F - F$, as shown in Figure 5.3. So the distribution of the packets that can be forwarded to the conversion bank from a single wavelength can be derived as follows:

$$P(Y_i = y | X_i > F) = \frac{P(Y_i = y \bigcap X_i > F)}{P(X_i > F)} \tag{5.2}$$

$$P\left(Y_i = y \bigcap X_i > F\right) = P(X_i = y + F) \tag{5.3}$$

$$P(X_i = y + F) = \binom{N \cdot F}{y + F} \cdot \left(\frac{\rho}{N}\right)^{y+F} \cdot \left(1 - \frac{\rho}{N}\right)^{N-y-F} \tag{5.4}$$

$$P(Y_i = y | X_i > F) = \binom{N \cdot F}{y + F} \left(\frac{\rho}{N}\right)^{y+F} \left(1 - \frac{\rho}{N}\right)^{N-y-F} \left(1 - \sum_{j=0}^{F} P(X_i = j)\right)^{-1} \tag{5.5}$$

The number of packets Y_i that were not served by the requested wavelength (λ_i) and forwarded by the optical switch to the conversion bank is identically independently distributed. Let N_Y be a random variable representing the number of *Y's* that are competing for conversion resources. In other words, N_Y represents the number of wavelengths that have more than F arrivals per time slot. So, N_Y could take any value from *zero* to w. When N_Y is equal to *zero*, it means that all packets that arrive on all wavelengths can leave the switch directly without any conversion, which means every wavelength on the output links under consideration has fewer than or equal F arrivals, or that none of the arrivals are destined to the output port under consideration. This implies that none of the special ports of the optical switch leading to the conversion bank are busy. This means that the number of packets lost will be zero, since all the arrived packets can leave the switching node directly. If every wavelength on the output has more than F arrivals, N_Y will equal w, which means that all wavelengths of the output are fully utilized and all packets forwarded to the conversion bank will be dropped due to wavelength exhaustion on the output. The distribution of N_Y is as follows:

$$P\left(N_Y = n_y\right) = \binom{w}{n_y} \left[\sum_{j=F+1}^{N\cdot F} P(X=j)\right]^{n_y} \cdot \left[\sum_{j=0}^{F} P(X=j)\right]^{w-n_y} \tag{5.6}$$

where $n_y = 0 \ldots w$

Conditioned on N_Y, λ_i is available on the output link if the number of arrivals per λ_i is less than F packets per time slot, which means zero packets forwarded to the conversion bank from λ_i $(Y_i = 0)$. These free $\lambda_i s$ need to be considered as available wavelengths that can be used by the conversion resources. For a given X_i (number of arrivals on λ_i at the inputs of the optical switch and destined to the considered output port), we define G_i as a random variable that represents the number of free λ_i out of the F copies on the output port, as shown in Figure 5.3. G_i takes values between *zero* and F, where *zero* means that λ_i on all F outgoing fibers are used. It implies that this wavelength has F arrivals or more. If G_i is equal to F, all λ_i on the output link are free and available for the conversion resources, which means there are no arrivals on λ_i or all arrivals are destined to other output ports. For a given number of arrivals per λ_i, we can calculate the distribution of G_i as follows:

$$P\left(G_i = g | X_i \leq F\right) = \frac{P\left(G_i = g \bigcap X_i \leq F\right)}{P\left(X_i \leq F\right)} \tag{5.7}$$

$$P\left(G_i = y \bigcap X_i \leq F\right) = P\left(X_i = F - g\right) \tag{5.8}$$

$$P\left(G_i = g | X_i \leq F\right) = \frac{P\left(X_i = F - g\right)}{\sum\limits_{j=0}^{F} P(X_i = j)} \tag{5.9}$$

The number of packets that can utilize conversion resources and leave the switch depends on the minimum of the total available wavelengths on all fibers at the output and the number of available converters. Since the packet length is fixed, all converters are available at the beginning of the time slot. Accordingly, the total number of wavelengths that are available on the output link, after all direct packets are assigned to outgoing wavelengths, must be found. To find the available wavelengths on the output link for a given number of fully used wavelengths (N_Y), we define B as a random variable representing the total number of wavelengths free of direct packets (i.e., not routed through converters) on the output link under consideration at any given time slot. The total number of available wavelengths on the considered output link is equal to the summation of the available copies of the individual wavelengths on the output. Hence, the random variable B is simply the convolution of the random variable G_i, which represents the number of fibers in the port that have no direct packet on λ_i:

$$B = \sum_{i=0}^{w} G_i \tag{5.10}$$

The total number of free wavelengths on an output port (B) varies between *zero* and $F(w - n_y)$. If every outgoing wavelength has at least F arrivals, the total number of free wavelengths, B, equals *zero*. In this case, all packets forwarded to the conversion bank are lost, as no free wavelength is available on the output port. In contrast, the total number of free wavelengths, B, equals $F(w-n_y)$ when there are $F \cdot n_y$ wavelengths on the output, each having more than F arrivals, where the remaining $F(w-n_y)$ wavelengths have less than F arrivals. Therefore, for a given N_Y (number of wavelengths that have at least one packet forwarded to the conversion bank), the distribution of B is given by:

$$P\left(B=\beta | N_Y=n_y\right)=P\left(\sum_{i=0}^{w-n_y} G_i=\beta\right) \tag{5.11}$$

Calculating the $P(B=\beta)$ using the above expression requires an extensive computation time because all the possible value permutations of G_i have to be considered for every value of n_y. So, to simplify the calculation of $P(B=\beta)$, the probability-generating function of G_i, *Z-transform*, is employed. The probability-generating function of G_i is given by:

$$H_{G_i}(z)=\sum_{g=0}^{F} P(G_i=g)\cdot z^g \tag{5.12}$$

so

$$H_B(z)|_{n_y}=\left[\sum_{g=0}^{F} P(G_i=g)\cdot z^g\right]^{w-n_y} \tag{5.13}$$

Now the distribution of B for a given N_Y can be found as follows:

$$P\left(B=\beta | N_Y=n_y\right)=\begin{cases} 0 & n_y=w \\ \frac{d^\beta}{dz^\beta}H_B(z)|_{z=0} & \text{otherwise} \end{cases} \tag{5.14}$$

Packet loss takes place when all λ_is are busy on all outgoing F fibers and the total number of packets forwarded to the conversion bank is greater than the minimum number of available converters and free wavelengths on the output. Some of the packets forwarded to the conversion bank because they could not leave the switch directly can be served and leave the switch by getting converted to one of the available wavelengths on the output, provided there are free wavelengths on the output port under consideration. The blocking probability of the switch, P_b, is then defined as the ratio of expected number of packets lost, $E[d]$, to the total number of arrivals at the output port $E\{\xi\}$:

$$P_b = \frac{E[d]}{E\{\xi\}} \tag{5.15}$$

The number of packets lost per time slot is equal to the difference between the total number of packets that need conversion and those can leave the switch after utilizing converters from the conversion bank. For a given N_Y, number of wavelengths that have more than F arrivals, the total number of packets that are directed to the conversion bank is equal to the sum of all rejected packets from all busy wavelengths (n_y). So, the maximum number of packets that can compete on $min(C, \beta)$, the minimum of the number of converters in the conversion bank that belong to the output port under consideration, C, and the total number of free wavelengths on the output link (B), are $n_y \cdot (N \cdot F - F)$. In this case, wavelength considered in n_y has NF arrivals, which means all arrivals at the inputs of the switch on all the n_y wavelengths are destined to the considered output port. Moreover, the minimum number of packets that can compete on $min(C, \beta)$ is n_y where every wavelength considered in n_y has $F + 1$ arrivals, which means the optical switch in Figure 5.3 switches one packet from every wavelength considered in n_y to the special ports that lead to the conversion bank belonging to the output under consideration. Let T be a random variable representing the total number of packets forwarded to the conversion bank. It can be obtained by taking the sum of packets forwarded to the conversion bank, Y_is, from all wavelengths that have more than F arrivals (n_y) i.e.,

$$T|n_y = \sum_{i=1}^{n_y} Y_i \tag{5.16}$$

For a given n_y, the probability of having $T = \alpha$, which takes values from n_y to $n_y \cdot (N \cdot F - F)$, packets competing on $min(C, \beta)$ is given by:

$$P(T = \alpha | N_Y = n_y) = P\left(\sum_{i=1}^{n_y} Y_i = \alpha\right) \tag{5.17}$$

To simplify the calculation of the $P(T = \alpha)$, we used the probability generating function of Y_i, *Z-transform*. The probability generating function of the Y_i is given by:

$$H_{Y_i}(z) = \sum_{y=1}^{N \cdot F - F} P(Y_i = y) \cdot z^y \tag{5.18}$$

so

$$H_T(z)|n_y = \left[\sum_{y=1}^{N \cdot F - F} P(Y_i = y) \cdot z^y\right]^t \tag{5.19}$$

for $t = ny \cdots ny(NF - F)$. Now the distribution of T for a given N_Y can be found as follows:

$$P(T = \alpha | N_Y = n_y) = \frac{d^\alpha}{dz^\alpha} \cdot H_T(z)|_{z=0} \tag{5.20}$$

For a given number of wavelengths that have more than F arrivals, n_y, the average numbers of packets forwarded to the conversion bank, $E[T]$, is equal to the summation of the average of packets forwarded to the conversion bank ($Y_i s$) from individual wavelengths considered in n_y which is given by:

$$E[T | N_Y = n_y] = E\left[\sum_{i=1}^{n_y} Y_i\right] = \sum_{i=1}^{n_y} E[Y_i] \tag{5.21}$$

since $Y_i s$ are *iid.*

$$E[T | N_Y = n_y] = n_y \cdot E[Y] \tag{5.22}$$

In general, the average number of packets forwarded to the conversion bank ($E[T]$) can be calculated by taking the summation over all possible values of the number of wavelengths that have more than F arrivals, N_Y, which is given by:

$$E[T] = E[Y] \cdot \sum_{n_y=0}^{w} n_y \cdot P(N_Y = n_y) \tag{5.23}$$

The average number of packets that are forwarded to the conversion bank per wavelength, $E[Y]$, is given by:

$$E[Y] = \sum_{j=1}^{N \cdot F - F} j \cdot P(Y = j) \tag{5.24}$$

To calculate the number of packets that can utilize the converters and leave the switch, we define a new random variable A to represent those packets that utilize the conversion bank. In a time slot, if the total number of wavelengths that forward at least one packet to the conversion bank, n_y, is less than or equal to the minimum of the total number of converters in the conversion bank and the total number of the available wavelengths on the output link, $min(C, \beta)$, and $\alpha \leq min(C, \beta)$, which is the total number of packets forwarded to the conversion bank, then $A = T$. This means that all packets forwarded to the conversion bank will utilize the conversion resource and leave the switch without any loss. Accordingly, the average number of lost packets, $E\{d\}$, equals *zero.* On the other hand, if $n_y > min(C, \beta)$ or $\alpha > min(C, \beta)$ only $min(C, \beta)$ packets can leave the switch by utilizing the wavelength conversion resources while the rest will be lost. So for a given n_y, the number of wavelengths that have more than F arrivals, and β (the total number of free wavelengths on the output port in a given time slot), the number of packets that can utilize the conversion resources and leave the switch, A, is defined as:

$$A = \begin{cases} \text{T, if } n_y \leq min(C, \beta) \; and \, T \leq min(C, \beta) \\ min(C, \beta), \text{otherwise} \end{cases} \tag{5.25}$$

Also *A's* distribution is given by expression 5.26.

$$P\left(A = \gamma \middle| N_Y = n_y, B = \beta\right) = \begin{cases} 1, & \gamma \leq min(C, \beta) \\ \\ P(T = min(C, \beta)), & \text{otherwise} \end{cases} \tag{5.26}$$

For a given n_y and β, the average number of packets that will utilize the conversion resources and leave the switch, $E[A|N_Y = n_y, B = \beta]$ can be calculated as follows:

$$E[A|N_Y = n_y, B = \beta] = \begin{cases} min(C,\beta), \text{ if } n_y \geq min(C,\beta) \\ \\ \sum_{j=n_y}^{min(C,\beta)} jP(T=j) + \\ min(C,\beta) \sum_{j=min(C,\beta)+1}^{n_y F(N-1)} P(T=j), \text{ otherwise} \end{cases} \tag{5.27}$$

From equation 5.27, if $n_y > min(C,\beta)$, the average number of packets that can be converted is equal to $min(C,\beta)$, and the remaining packets will be dropped. On the other hand, if n_y is less than or equal to $min(C,\beta)$, the packets that arrive at the conversion bank are converted, as long as their number is less than or equal to $min(C,\beta)$. Otherwise, only $min(C,\beta)$ packets, out of those arrivals, are converted. Since the maximum number of packets that can utilize the conversion and leave the switch is dependent on the $min(C,\beta)$, the average number of packets that will utilize the conversion resources and leave the switch, $E[A]$, can be calculated by taking the summation over all possible values of N_Y and B given as follows:

$$E[A] = \sum_{n_y=1}^{w} \left\{ P(N_Y = n_y) \sum_{\beta=0}^{(w-n_y)F} \Big(P(B=\beta) E[A|N_Y = n_y, B = \beta] \Big) \right\} \tag{5.28}$$

Now we can calculate the average number of packets that will be dropped, $E[d]$, as the difference between the average of total forwarded packets to conversion bank, ET, and the average accepted packets for conversion, $E[A]$, given by:

$$E[d] = E[T] - E[A] \tag{5.29}$$

The average number of packet arrivals per wavelength on the output under consideration is equal to $F \cdot \rho$. So, the total expected arrival to the output link, shown in the denominator of equation 5.15, is given by:

$$E[\xi] = w \cdot F \cdot \rho \tag{5.30}$$

Then the blocking probability can be found using equations 5.15, 5.29, and 5.30.

A new measure to evaluate the cost-effectiveness of the different resources inside the optical packet switch will be derived. This measure is the resources wastage factor. The resources wastage factor is defined as the ratio between the resources that were not utilized in the presences of the packets drop and the total resources. The wavelength transceivers and conversion devices are the two main resources that could be considered as wasted resource if they are not utilized. All wavelength transceivers that are not utilized in the presence of the packet drop are considered to be wavelengths wastage. The wavelength resources wastage could take place when the number of the converters is less than the number of the free wavelengths, $min(C, \beta) = C$, which means all wavelength transceivers that are not utilized are wasted resources. This kind of the wastage happened due to a limited numbers of conversion devices being deployed. On the other hand, if the number of free wavelengths is less than the number of converters, $min(C, \beta) = \beta$, all the converters that are not utilized are considered as conversion resources wastage. The developed measure can be used as a tradeoff measure between these two wasted resources.

In the optical packet switch without conversion option, only wavelength resources wastage could take place. We will consider this case first, and then a general resources factor will be presented. For a given number of wavelengths that have less than F arrivals, $w - n_y$, the average numbers of wavelength wastage resources, $E[\gamma]$, is equal to the average number of free wavelengths on the output under consideration ($E[B]$), which is equal to the summation of the average of free wavelengths ($G_i s$) from all individual wavelengths not considered in n_y which is given by:

$$E[\gamma | N_Y = n_y] = E[B | N_Y = n_y] \tag{5.31}$$

$$E[\gamma|N_Y = n_y] = E\left[\sum_{i=1}^{w-n_y} G_i\right] = \sum_{i=1}^{w-n_y} E[G_i] \tag{5.32}$$

$$E[\gamma|N_Y = n_y] = \sum_{i=1}^{w-n_y} E[G_i] \tag{5.33}$$

since $G_i s$ are *iid.*

$$E[\gamma|N_Y = n_y] = n_y \cdot E[G] \tag{5.34}$$

In general, the average number of wasted wavelength resources ($E[B]$) can be calculated by taking the summation over all possible values of the number of wavelengths that have less than F arrivals, N_Y, which is given by:

$$E[\gamma] = E[G] \cdot \sum_{n_y=0}^{w} n_y \cdot P(N_Y = n_y) \tag{5.35}$$

The average number of free copies of a wavelength, $E[G]$, is given by:

$$E[G] = \sum_{j=0}^{F} j \cdot P(G = j) \tag{5.36}$$

So, the wavelength resources wastage factor (WRWF) in the absences of the wavelength conversion can be calculated as follows:

$$WRWF = E[\gamma]/(F * w) \tag{5.37}$$

The wavelength resources wastage in the presences of the conversion option in the switch take place only in one case when the number of free wavelengths is larger than the number of converters, $min(C, \beta) = C$. So, the wavelength resources wastage factor has to consider those packets that can utilize the conversion devices and leave the switch ($E[A]$), which is calculated previously. Therefore, the average number of wasted wavelength resources can be defined as follows:

$$E[\gamma] = E[B] - E[A] \tag{5.38}$$

where $E[B]$ is the average of free wavelengths on the output and is given by:

$$E[B] = E[G] \cdot \sum_{n_y=0}^{w} n_y \cdot P(N_Y = n_y) \tag{5.39}$$

Therefore, the wavelength resources wastage factor for the optical packet switch with wavelength conversion is:

$$WRWF = E[\gamma]/(F * w) \tag{5.40}$$

The conversion resources wastage is defined as the number of converters that cannot be utilized in the presences of packets drop. This type of resources wastage takes place only in one case when the number of available conversion devices is larger than the number of free wavelengths, i.e., $min(C, \beta)$=beta. The average of the conversion resources wastage is defined as follows:

$$E[\zeta] = C - E[A] \tag{5.41}$$

where $E[A]$ is the average number of packets that can leave the switch after being converted to one of the free wavelengths. Therefore, the conversion resources wastage factor (CRWF) for the optical packet switch with wavelength conversion is:

$$CRWF = E[\zeta]/C \tag{5.42}$$

An expression for the conversion bank utilization considering all switch parameters can be derived. The conversion bank utilization is defined as the ratio between the average number of packets that can be converted to the total number of the converter devices that are shared among all wavelengths on a single output link. The conversion bank is 100% utilized when $min(C, \beta) \geq C$ while $\alpha \geq C$. This suggests that C packets out of T leave the switch after getting converted. On the other hand, the conversion bank will be 0% utilized if all contending packets are dropped because of unavailability of wavelengths on the output link, $(\beta = 0)$, or if all arrived packets, $X_i s$, are less than or equal to F, where the conversion bank is not needed. Accordingly, the conversion bank utilization is defined as the average number of packets leaving the switch, after conversion, over the number of shared converter devices per output link. The utilization is given by:

$$U = \frac{E[A]}{C} \tag{5.43}$$

where $E[A]$ is obtained by equation 5.28. The following section discusses the simulation and the analytical models' results in further detail.

5.4 Results

The performance of a synchronous optical packet switch with a conversion bank share-per-output-link is evaluated and discussed in this section. The switch parameters such as number of converters C, number of wavelength per fiber w, number of fibers per link F and the load per wavelength ρ are considered in this evaluation. The perfect match between the simulation-based model and the analytical model results verifies the accuracy of the OPS analytical model developed in the previous section. The simulation results were generated using the OPS discrete event simulator that was developed in this research work. The OPS simulator is explained in detail in Chapter 7, which discusses the all-optical network simulation. Conducting the performance evaluation of OPS through a simulation increases the validity and broadens the scope of the OPS analysis due to the flexibility of evaluating the OPS performance under different setting parameters in a short time with a very high accuracy.

In Figure 5.4, the packet loss probability is plotted as a function of offered load per wavelength for 3 differently configured synchronous switches. We considered 3 different switch parameters, F, w, and C, where F is the number of fibers per link, w is the number of wavelengths per fiber, and C is the number of shared

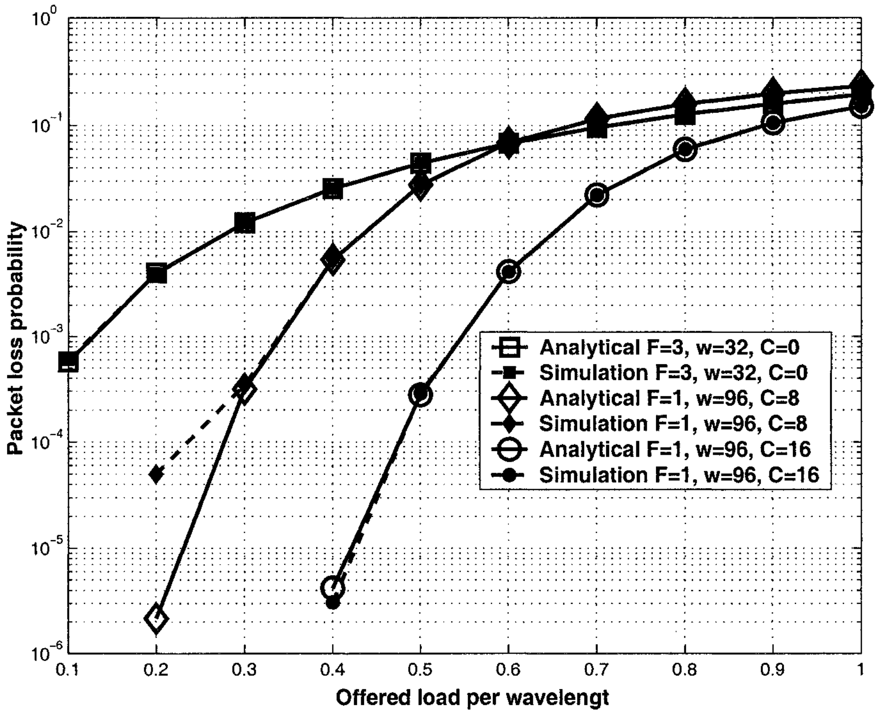

Figure 5.4: Packet loss probability vs. offered load per wavelength.

converters per link. The results for this figure consistently show that a single fiber link switch with *w=96* and *C=16* behaves considerably better than a similar switch with a smaller number of shared converters per link, *C=8*. It is noticeable that both switches behave in a similar fashion in lower rates of offered loads due to the abundance of wavelength and conversion resources compared to the offered load served. At higher offered loads, the switch with a higher count of shared converters per link offers better packet loss performance due to its ability to accommodate the higher traffic.

Figure 5.4 also shows that a switch with *F=3* and *C=0* performs better at higher loads compared to the *F=1* and *C=8* switch. This indicates that using a multiple fiber link in higher offered load regions has far better a effect on performance than the converter count even though the number of total wavelengths is the same in both cases. In the case of multiple fibers, dividing the total number of

ρ	1^{st} *Switch configuration: F=1, C=8, w=96*		
	$P_{Analysis}$	$P_{Simulation}$	%Error
0.1	0.0000	0.0000	-
0.2	0.0000	0.0000	-
0.3	0.0003	0.0003	0
0.4	0.0054	0.0054	0
0.5	0.0275	0.0276	0.3623
0.6	0.0683	0.0684	0.1462
0.7	0.1145	0.1146	0.0873
0.8	0.1579	0.1580	0.0633
0.9	0.1971	0.1972	0.0507
1.0	0.2330	0.2331	0.0429

Table 5.1: Packet loss probability and % error for the first switch configuration in Figure 5.4.

wavelengths available into sets of similar wavelengths in each fiber offers a better utilization of wavelength resources and, accordingly, better packet loss behavior at higher rates of offered loads. In lower offered loads, the effect of shared converter counts in the single fiber case has more effect on utilization than the multi-fiber link case due to the role the conversion plays in resolving contention and the high availability of the wavelengths on the outgoing fibers. These results show a total agreement between results obtained by simulation and the results obtained by the analytical model, confirming the accuracy of the analysis.

The actual values of packet loss probability obtained by analytical and simulation models for all switch configurations considered in Figure 5.4 are listed in Tables 5.1, 5.2, and 5.3 to show the accuracy and the perfect matching between the analytical and simulation models for different switch configuration parameters. Also, in these tables the percentage error is presented. The percentage error is calculated using the following expression:

$$\%Error = 100 * \frac{|P_{Simulation} - P_{Analysis}|}{P_{Simulation}} \tag{5.44}$$

ρ	2^{nd} *Switch configuration: F=1, C=16, w=96*		
	$P_{Analysis}$	$P_{Simulation}$	%Error
0.1	0.0000	0.0000	-
0.2	0.0000	0.0000	-
0.3	0.0000	0.0000	-
0.4	0.0000	0.0000	-
0.5	0.0003	0.0003	0
0.6	0.0042	0.0041	2.4390
0.7	0.0221	0.0221	0
0.8	0.0593	0.0593	0
0.9	0.1052	0.1051	0.0951
1.0	0.1498	0.1499	0.0667

Table 5.2: Packet loss probability and % error for the second switch configuration in Figure 5.4.

ρ	3^{rd} *Switch configuration: F=3, C=0, w=32*		
	$P_{Analysis}$	$P_{Simulation}$	%Error
0.1	0.0006	0.0006	0.0000
0.2	0.0041	0.0040	2.5000
0.3	0.0121	0.0121	0.0000
0.4	0.0255	0.0254	0.3937
0.5	0.0441	0.0442	0.2262
0.6	0.0675	0.0677	0.2954
0.7	0.0950	0.0953	0.3148
0.8	0.1258	0.1262	0.3170
0.9	0.1589	0.1592	0.1884
1.0	0.1936	0.1938	0.1032

Table 5.3: Packet loss probability and % error for the third switch configuration in Figure5.4.

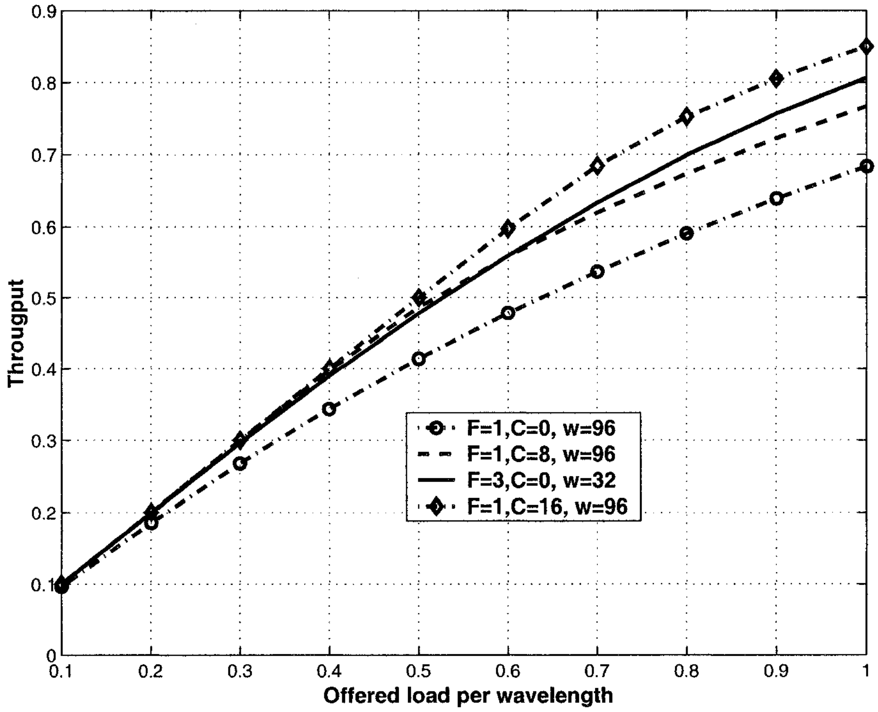

Figure 5.5: Throughput Vs. Offered Load Per Wavelength

The OPS throughput vs. the offered load per wavelength is plotted in Figure 5.5. The OPS throughput is plotted for all switch configuration cases that were considered in Figure 5.4 as well as the OPS throughput for a single fiber without conversion. The results presented in this figure prove one of the main objectives of this research, that OPS performance could be enhanced without any additional costly resources. This observation could be easily drawn for the figure through comparing the multi-fiber throughput to the single-fiber without conversion and with conversion in case of *C=8*. It is clearly shown that as few as two more fibers provides the OPS with a better throughput than adding eight converters. These added conversion resources introduce more overhead cost to the OPS, whereas the cost of the extra hardware needed inside the OPS to implement the multi-fiber scenario is insignificant compared to the cost of the conversion device. Even when the individual wavelength is highly loaded, the throughput of the multi-fiber switch

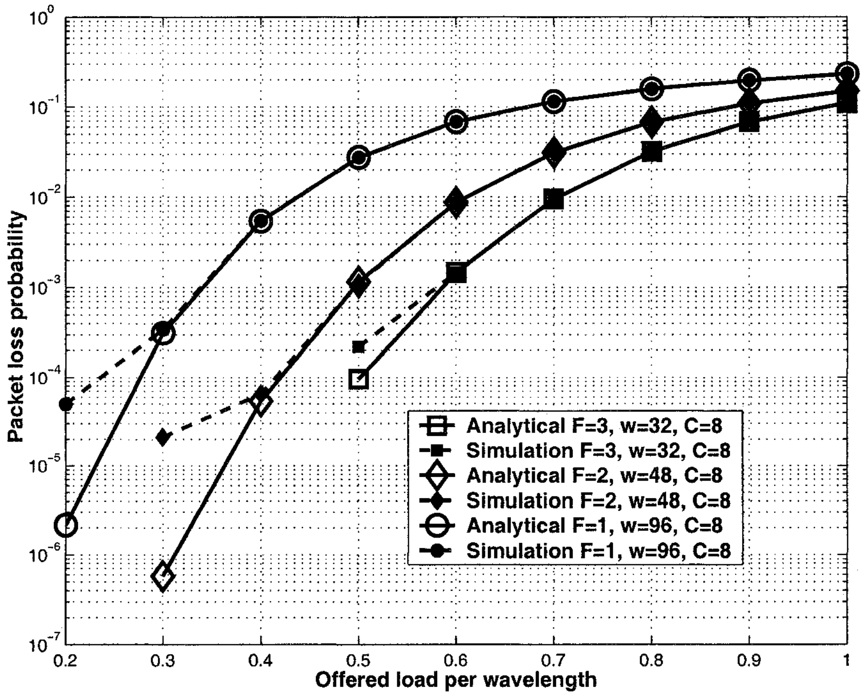

Figure 5.6: Packet loss probability vs. offered load per wavelength.

is better than the switch configuration with 8 converters per output. Thus, the conversion device's utilization saturation is reached faster in the higher load. This observation will be explained and presented in more detail in the following figures.

In Figure 5.6, the packet loss probability is plotted as a function of offered load per wavelength for 3 switches with different values for F and w, where w is constant. This figure confirms the earlier observation that the effect of using multiple fibers per output link has a dominant effect over the count of shared converters in the conversion bank even though the total wavelength count in the 3 different cases is the same, the total wavelengths in each case being w=*96*. It also confirms that the simulation results totally substantiate the accuracy of the developed analytical model. It can be deduced that the number of shared converters can be significantly reduced when using multiple fibers due to the higher utilization the multi-fiber link offers for the same set of total wavelengths in the system. A similar number of

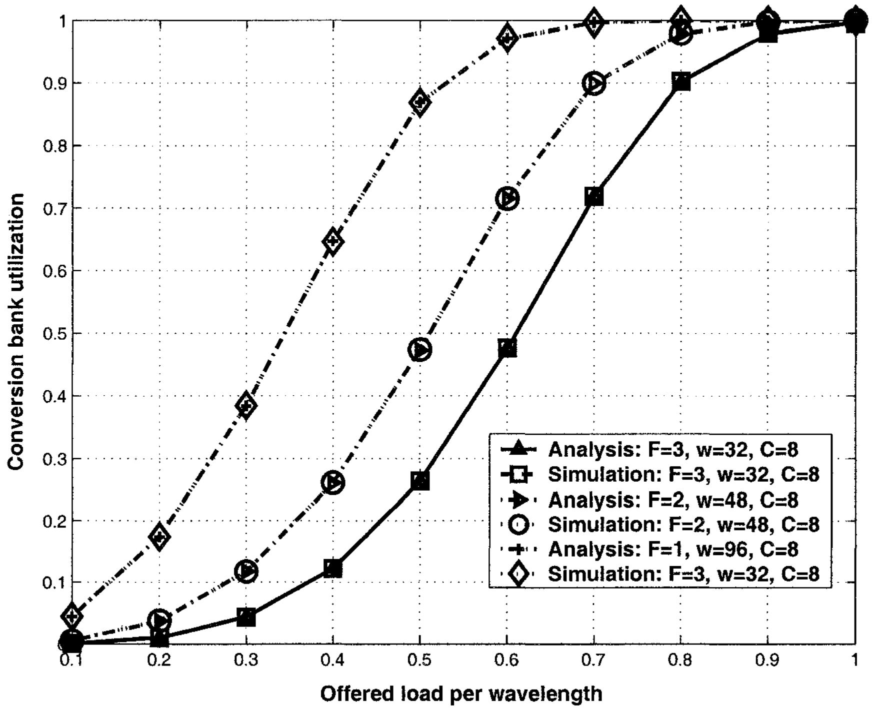

Figure 5.7: Conversion bank utilization vs. offered load per wavelength.

shared converters in all of the presented cases, *C=8*, is considered to emphasize the effect of using multiple fibers. This result offers an opportunity for cost tradeoffs between using multiple fiber links and the shared converter count in the network. These results are more evident in higher offered loads due to the need to better manage the fiber, wavelength, and shared converter resources.

In Figure 5.7, the conversion bank utilization is plotted as a function of offered load per wavelength for the same switches considered in Figure 5.6. This figure shows that output links with fewer fiber counts have higher conversion bank utilization while switches with higher fiber count in their output links have lower utilization rate for the conversion resources. The opportunity for cost tradeoffs is much clearer in this case where less conversion bank utilization means that the need for conversion is less in the multi-fiber case when the total number of wavelength is the same. From this, it can be deduced that a synchronous switch equipped with

ρ	1^{st} *Switch configuration: F=1, C=8, w=96*	
	$P_{Analysis}$	$P_{Simulation}$
0.1	0.044254687	0.0442075
0.2	0.174069851	0.174063
0.3	0.383998745	0.38457
0.4	0.647230397	0.646409
0.5	0.869085054	0.868846
0.6	0.972303712	0.9725
0.7	0.996826382	0.996886
0.8	0.999803957	0.999813
0.9	0.999993394	0.999988
1.0	0.999999875	0.999995

Table 5.4: Conversion bank utilization for the first switch configuration switch configurations in Figure 5.7.

full conversion would have the least conversion utilization rate, indicating that the use of a switch with a lower converter count, partial conversion, would offer better switch resources utilization and comparable packet loss behavior. Furthermore, the need for the costly resources in the optical network would be minimized through utilizing the multi-fiber environment while the network performance will be enhanced. To show the accuracy of the conversion bank utilization model that was developed and utilized to generate the results in Figure 5.7, the actual values for the conversion bank utilization for all switch configurations in this figure are listed in tables 5.4, 5.6, and 5.6. These values show the perfect match between the analysis and simulation for the conversion bank utilization model.

ρ	2^{nd} *Switch configuration: F=2, C=8, w=48*	
	$P_{Analysis}$	$P_{Simulation}$
0.1	0.004931556	0.00514125
0.2	0.037054684	0.0375025
0.3	0.117445394	0.117325
0.4	0.261173434	0.261374
0.5	0.47257443	0.473389
0.6	0.715605	0.715687
0.7	0.899812452	0.899701
0.8	0.979380653	0.979414
0.9	0.997666291	0.997602
1.0	0.999831689	0.999834

Table 5.5: Conversion bank utilization for the second switch configuration switch configurations in Figure 5.7.

ρ	3^{rd} *Switch configuration: F=3, C=8, w=32*	
	$P_{Analysis}$	$P_{Simulation}$
0.1	0.00068585	0.00069375
0.2	0.009727875	0.010045
0.3	0.04364732	0.0437587
0.4	0.122237081	0.122126
0.5	0.263913694	0.263863
0.6	0.475575901	0.475687
0.7	0.718916195	0.719347
0.8	0.90179557	0.902464
0.9	0.980023137	0.980234
1.0	0.996879211	0.996889

Table 5.6: Conversion bank utilization for the third switch configuration switch configurations in Figure 5.7.

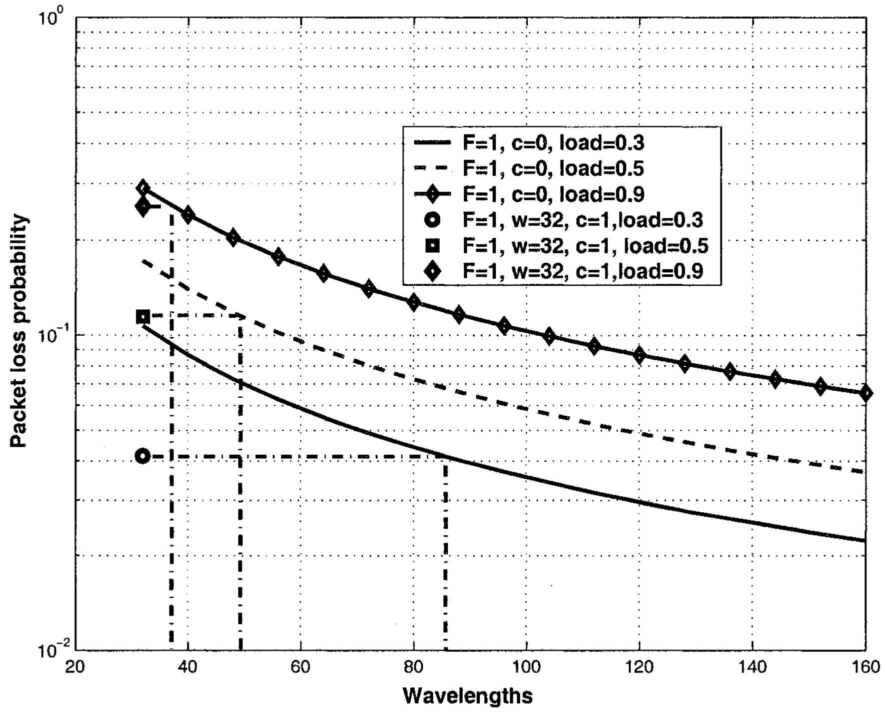

Figure 5.8: Conversion device worthiness.

In this study, the worthiness of a single conversion device in terms of wavelengths is investigated. The conversion device worthiness is defined as the additional wavelengths that are needed by OPS without wavelength conversion to achieve the same packet loss probability when only one converter is utilized. In Figure 5.8, the packet loss probability for OPS without wavelength conversion is plotted as a function of the number of wavelengths for three different load settings. In this figure, the total input load to the switch is kept constant as $w * loadperwavelength$, where $w = 32$. The load per wavelength is adjusted whenever additional wavelengths are added to the switch by dividing the original total input load ($w * Load\ per\ wavelength$) by the total number of wavelengths (including the new added wavelengths). The packet loss probability decreases with the number of wavelengths is increased due to the redundance of resources available and the reduction of the load per wavelength. Also, the packet loss probability for

OPS with a single converter and 32 wavelengths is plotted for the same different load settings considered in the non-conversion case. This figure shows that the conversion device is worth more in the low-load environment; for example when the original load per wavelength is 0.3, the converter is worth 54 additional wavelengths $(86 - 32)$ due to the lower competition per wavelength. In the highly loaded network, the worth of the conversion device is insignificant compared to the overhead cost that will be added to the OPS cost implementation. In this figure, the single wavelength converter in the high load per wavelength case is worth 6 wavelengths. These results confirm that using multi-fiber in the high loaded environment enhances the blocking performance more than utilizing the high cost conversion resources.

Also, the multi-fiber environment worthiness in terms of wavelengths is investigated. The worthiness of the fiber is defined as the additional wavelengths that are needed by OPS without wavelength conversion to achieve the same packet loss probability when only one additional fiber is added. Figure 5.9 shows the packet loss probability that was plotted in Figure 5.8, but this time the packet loss probability is compared to the multi-fiber environment blocking performance when an additional fiber is added. The total input load to the switch is kept constant as $F * w_{perfiber} * load\ per\ wavelength$.

In Figure 5.9, the OPS parameters are set to $F = 2$, $w = 16$, and $C = 0$ and the packet loss probability is generated for different load per wavelength. Then the number of fibers at the input and the output side are set to 1 to generate the single-fiber curves as the number of wavelengths changes. As in Figure 5.8, the load per wavelength is adjusted whenever additional wavelengths are added to the switch. This figure shows that the worthiness of the added fiber is more than the worthiness of the single converters in the low-load environment. Adding one fiber to an OPS switch is worth 80 wavelengths in a light-loaded single fiber OPS, while,

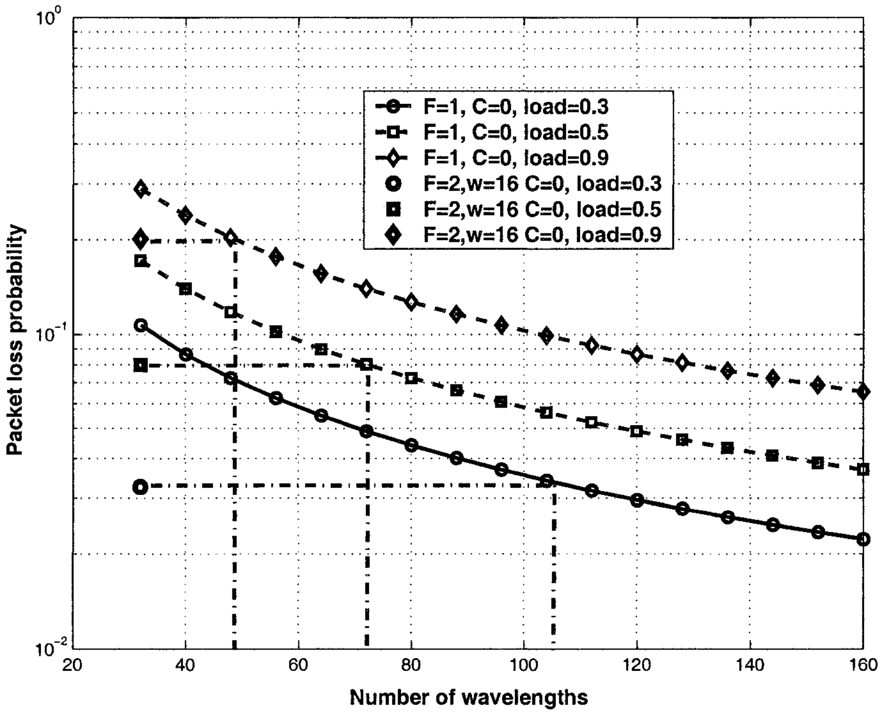

Figure 5.9: Fiber worthiness

as shown in Figure 5.8, the converter device was worth 54. In the highly-loaded network, the worth of the added fiber is more than double the converter worth; in addition, the improvement in the OPS performance does not come at the price of adding a costly device. These results confirm that the use of the multi-fiber environment enhances the all-optical network performance without extra overhead cost.

Figures 5.10 and 5.11 consider packet loss probability with respect to the number of converters in the shared conversion bank where the offered load per wavelength, ρ, is 1.0. Both graphs in the figure consider the same set of *F=1, 2, and 3.*

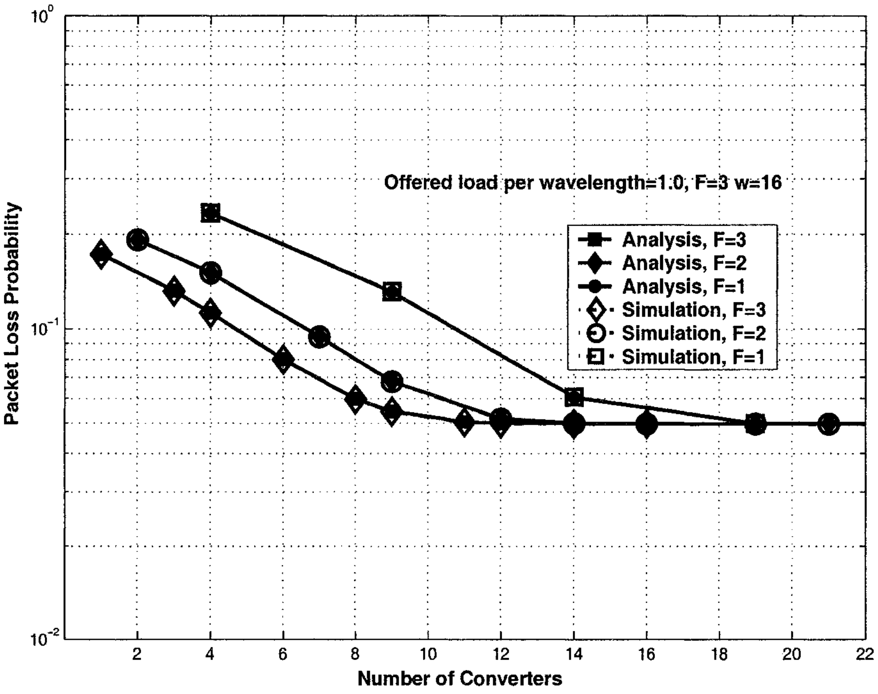

Figure 5.10: Packet loss probability vs. number of converters for *w=16.*

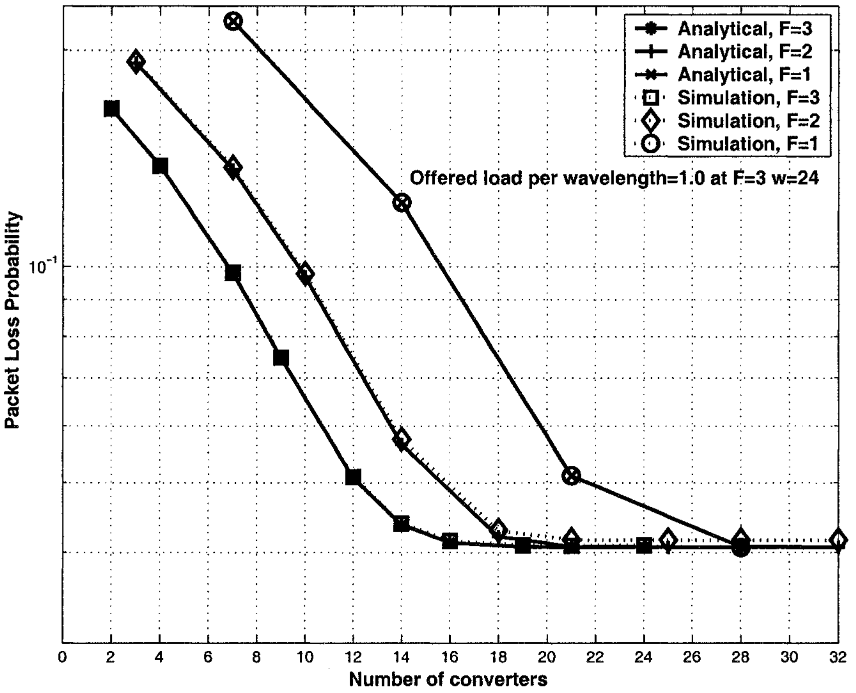

Figure 5.11: Packet loss probability vs. number of converters for *w=24.*

Figure 5.10 considers *w=16* per fiber and Figure 5.11 considers *w=24*. In both cases, the packet loss probability decreases with the increase of the converter count in the shared conversion bank until it reaches a point where the number of converters doesn't make any difference. The point where the addition of converters would not improve the packet loss probability depends on the fiber count on the output link. The higher the fiber count, the faster the switch reaches the point where adding converters would not be necessary. In Figure 5.10, for instance, when *F=3*, having more than 12 converters in the shared conversion bank would not improve the performance further. This number increases when using lower fiber counts in the output link. Given the wavelength count in the output link, there is a limit on how much the packet loss performance can improve. In Figure 5.10, given the indicated switch parameters and *w=16*, the packet loss probability improvement is bounded by $P_b = 0.05$. When the wavelength count is higher, as shown in Figure 5.11, more packet loss probability improvement can be achieved, but higher counts of shared converters are required to reach such a point.

In Figure 5.12, the OPS percentage performance enhancement due to utilizing the wavelength conversions or the multi-fiber as contention-resolving techniques when compared to OPS without employing any contention-resolution technique, is plotted as a function of the total wavelengths for three different loads, 0.3,0.5, and 0.9. In this figure, the percentage of the performance enhancement decreases with increasing the number of wavelengths in all cases due to the abundance of the resources availability and the reduction of the load per wavelength as the number of wavelengths increases. The percentage of the performance enhancement for the OPS employing multi-fiber as a contention-resolution scheme is more than the performance enhancement of the OPS with wavelength conversion as a contention-resolution technique in all cases.

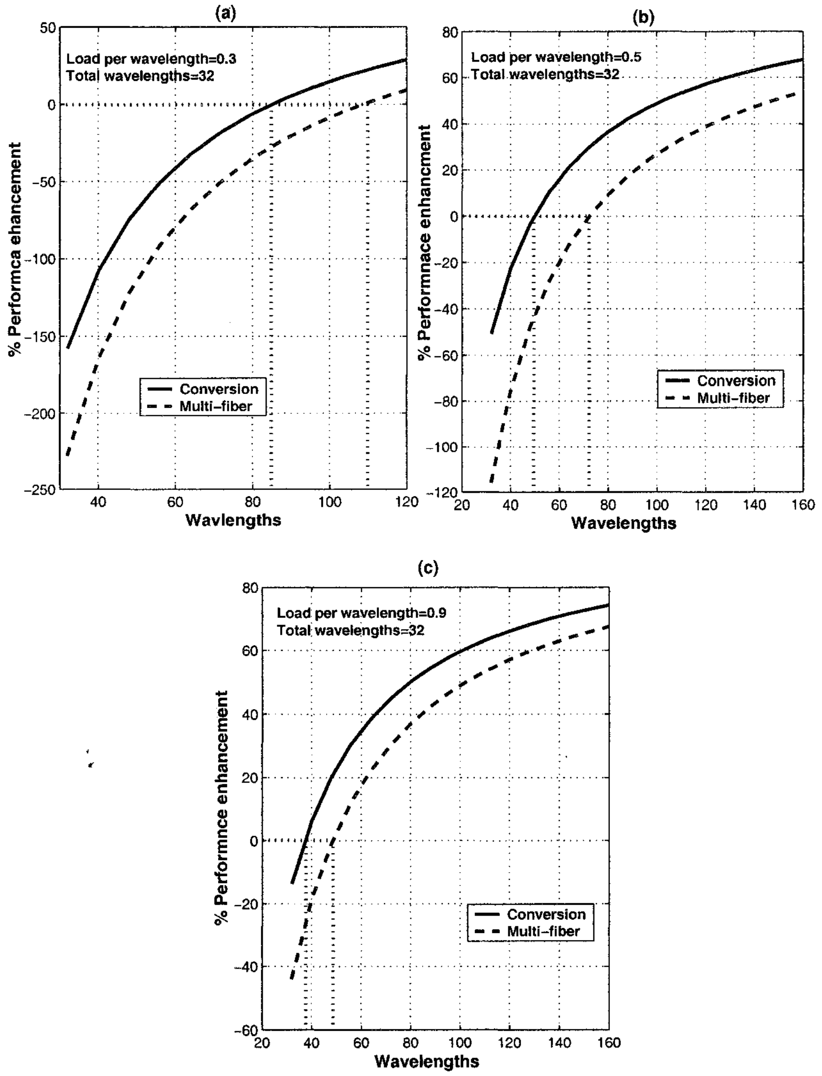

Figure 5.12: % Perfomance enhancement vs. number of wavelenhgths.

The Consistency of this result with the previous observation and earlier results confirms and verifies the main hypothesis of this research, which is that utilizing the multi-fiber environment in the optical networks could enhance their performance without the very high overhead cost that will be added when wavelength conversion is utilized. The percentage of the performance enhancement is very significant when the load per wavelength is low, as shown in Figure 5.12(a), because the worthiness of any contention scheme with respect to the number of wavelengths is

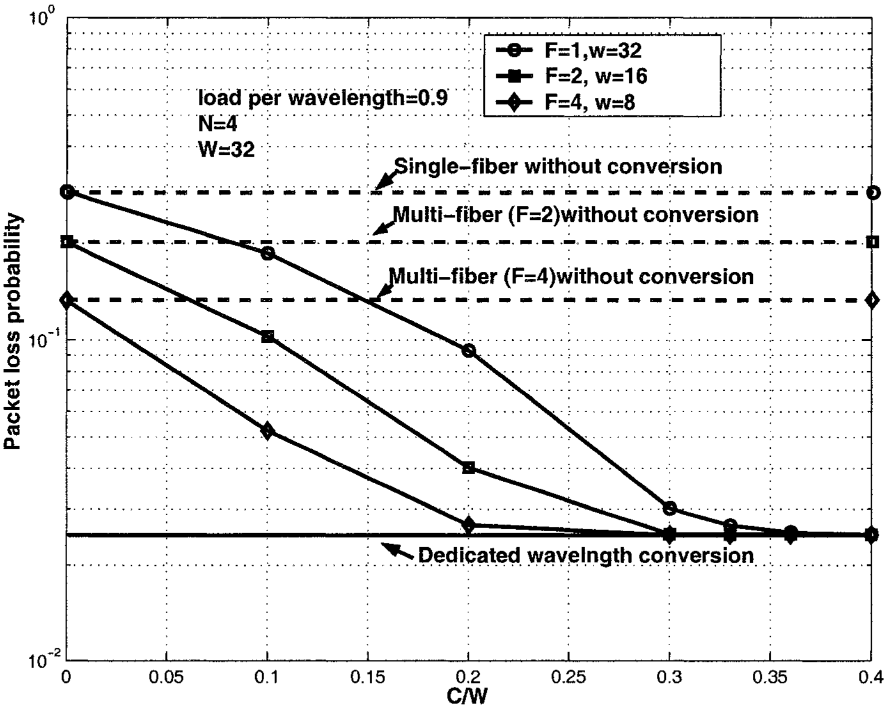

Figure 5.13: Packet loss probability vs. C/W.

very high in this case, as shown by Figures 5.8 and 5.9. Moreover, the minimum performance enhancement that could be achieved when the load per wavelength is high (0.9) and the wavelength conversion is utilized as a contention-resolution technique, as shown in Figure 5.12(c), is 28%, whereas in the multi-fiber case it is 45%.

The packet loss probability is plotted as a function of $\frac{C}{w}$ in Figure 5.13 to show the saving in the number of conversion resources when the multi-fiber switch configuration is utilized. The switch performance has upper and lower boundaries, as is true for any switch of parameters configuration. The upper boundary for the OPS performance is the packet loss probability of the switch when no wavelength-conversion is employed. The OPS upper boundary decreases with an increasing number of fibers, as shown by the horizontal lines in Figure 5.13.

The best performance that can be achieved determines the lower boundary of the switch performance. The best performance could be achieved for the OPS performance when a dedicated converter device is assigned to every outgoing wavelength. This lower boundary is equal for the all 3 configurations and cannot be enhanced further with any other parameter settings. Consequently, the 3 different switch configurations that are considered in this figure fall between these two boundaries. As the ratio of $\frac{C}{w}$ increases, which means the number of converters in the switch increases, the packet loss probability decreases.

The sharpness in the slope of the single-fiber curve is due to the effectiveness of any additional conversion resources added. This means that adding more converters in the single-fiber environment has more effect on the switch performance than adding the same number of converters in multi-fiber. However, the total number of converters that are needed to reach the dedicated switch configuration performance reduces when the number of fibers increases. For instance, if the number of converters is 20% of the total number of the wavelengths in the OPS in the case of $F = 3$, this is sufficient to reach the dedicated performance configuration. In contrast, in the single-fiber case the number of the converters in the switch has to be at least 33% of the number of the wavelengths to achieve the same performance. Also, this figure confirms that a limited number of converters inside the optical switch is sufficient to give the best performance that can be achieved, which validates the main objective for this research.

In Figures 5.14 and 5.15, the conversion bank utilization is plotted as a function of the number of converters for the same switches discussed in Figures 5.10 and 5.11. Both graphs in the figures consider the same set of *F=1, 2, and 3.* Figure 5.14 considers *w=16* per fiber whereas Figure 5.15 considers *w=24*. In both cases, the conversion bank utilization decreases with the increase of the converter count in these shared conversion banks. The utilization of the conversion bank

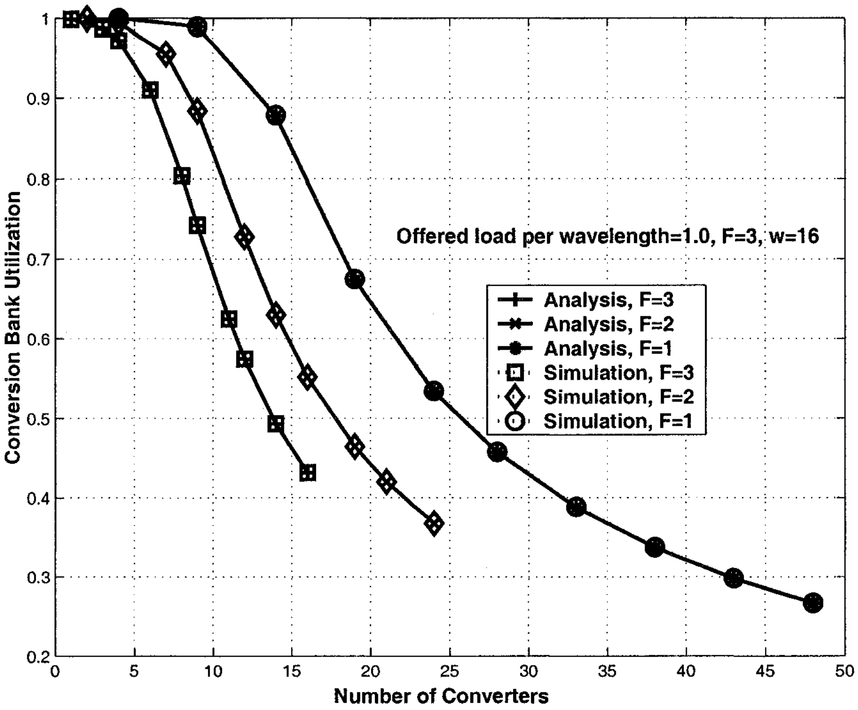

Figure 5.14: Conversion bank utilization vs. number of converters in the Conversion bank for *w=16*.

is less for the case of the multi-fiber link, in which more converters are required. For a given number of converter counts in Figures 5.14 and 5.15, where *w=16 and 24* respectively, the conversion bank utilization in the system that has a higher wavelength count is better than a system that employs fewer wavelengths. This means that a switch with a higher wavelength count has a better conversion bank utilization for a given number of shared converters. Figures 5.10, 5.11, 5.14, and 5.15 illustrate the relationship between packet loss probability, conversion bank utilization, and the number of shared converters per link. These figures confirm the argument that full wavelength conversion, where each wavelength has its own converter, is considered as the least optimized switch from the point of view of utilization. As we discussed in Figures 5.10 and 5.11, given the parameters of the switch, there is a point where adding more converters would not result in better packet loss performance. Accordingly, any further addition of conversion resources

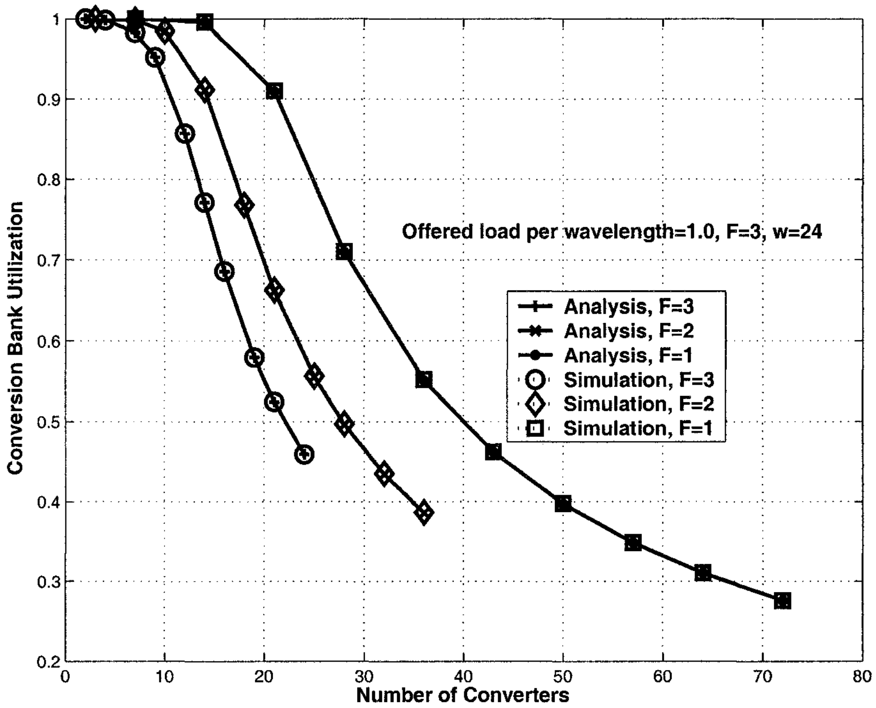

Figure 5.15: Conversion bank utilization vs. number of converters in the conversion bank for *w=24*.

would reduce the utilization of the shared conversion banks, resulting in wastage of resources, especially at lower counts of wavelength per fiber.

In this study, the wasted optical resources are investigated. All optical resources that are not utilized in the presence of the packets drop are considered to be wasted resources. The wavelength transceivers (Transmitters/ Receivers) and conversion devices are the two main resources that could be considered as wasted resource if they are not utilized. Thus, the optical resources wastage factor is defined as the ratio between the resources that were not utilized in the presences of the packets drop and the total resources. In Figure 5.16, the wavelength recourses wastage factor (WRWF) is plotted as a function of the load per wavelength for 3 differently configured synchronous switches. A WRWF decrease with the load per wavelength is increased due to the high competition on the wavelength resources on the output under consideration. The wasted wavelength resources in

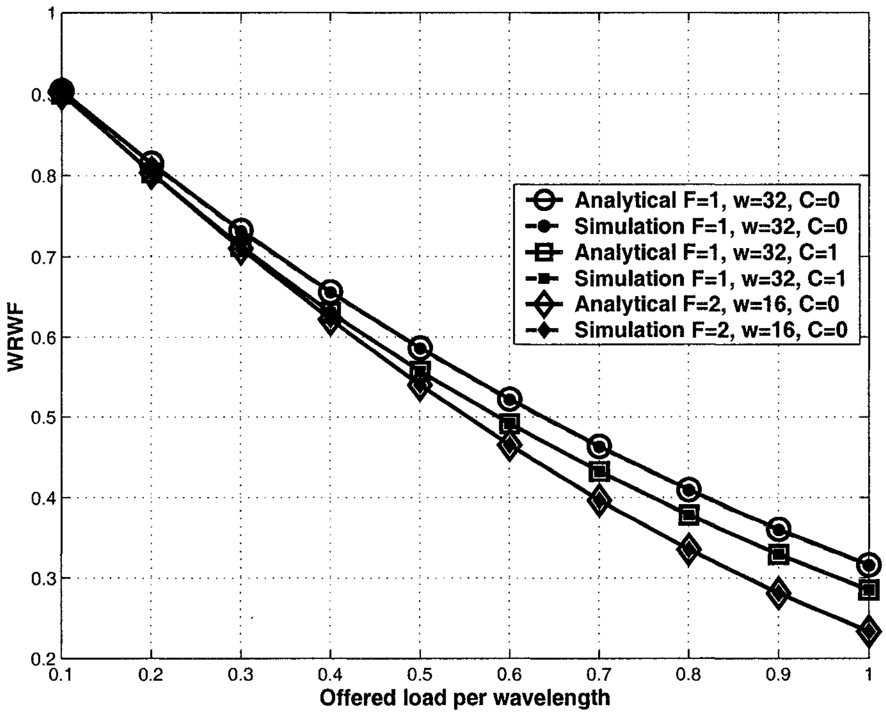

Figure 5.16: WRWF vs. the offered load per wavelength

the multi-fiber environment are considerably less than that of the single-fiber case due to the added configuration flexibility provided by the multi-fiber in supporting multiple requests on the same wavelength simultaneously. This figure also shows that utilizing the wavelength conversion in the OPS reduces the number of wasted wavelength resources. In a highly loaded network, the effect of the multi-fiber setting is much better compared to the wavelength conversion setting.

In Figure 5.17, the conversion recourses wastage factor (CRWF) is plotted as a function of number of wavelengths for two different load configurations. CRWF decrease as the number of wavelengths increases due to the high competition on the conversion resources on the output under consideration. In a highly loaded network, the packets drop occurs due to the lack of conversion resources; therefore CWRF decreases as the load per wavelength increases. So, employing wavelength

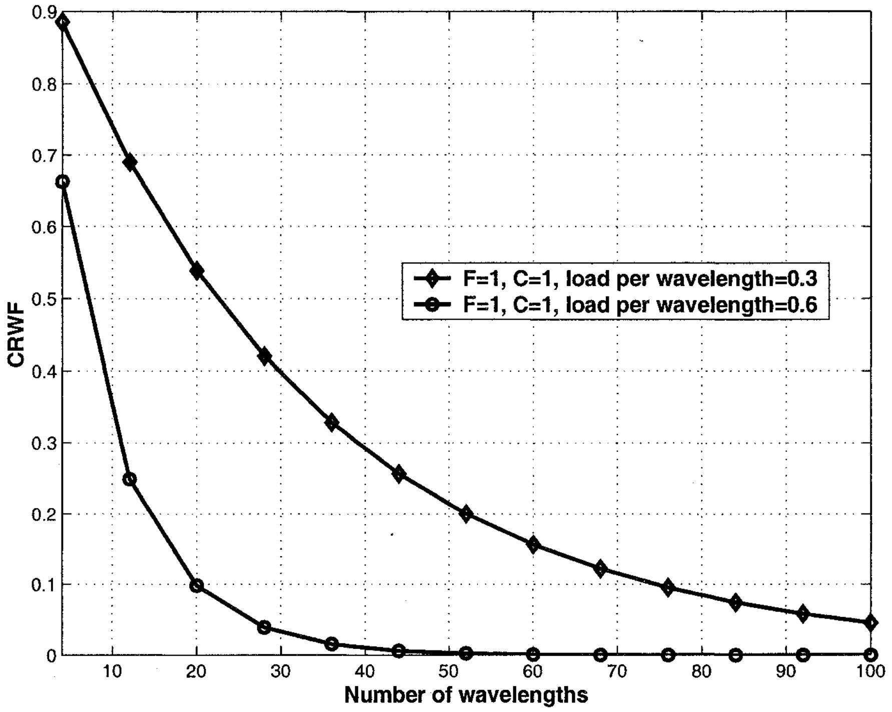

Figure 5.17: CRWF vs. number of wavelengths

conversion in the highly loaded networks will result in the most efficient utilization of the most costly optical resources.

Finally, in Figure 5.18, the gain is plotted as a function of the offered load. The gain in this figure is in referenced to the performance of a switch that has *F=3*, *w=32* per fiber, and *C=8* for a link. As expected, the gain, compared to the reference switch, declines when *F=2*, even though the total number of wavelengths employed by these fibers is the same as the total wavelengths employed in the reference switch and *C=8*. This shows that increasing the fiber count, *F*, has a significant effect on performance, confirming results discussed earlier. If *F=3*, *C=0* and *w=32* per fiber, the figure shows that such a switch has the lowest gain in the region of low offered loads, whereas a switch with *F=1*, *C=8* and *w=96* outperforms the earlier switch in the similar offered load region. In the higher region of offered loads, these two switches behave in an opposite manner.

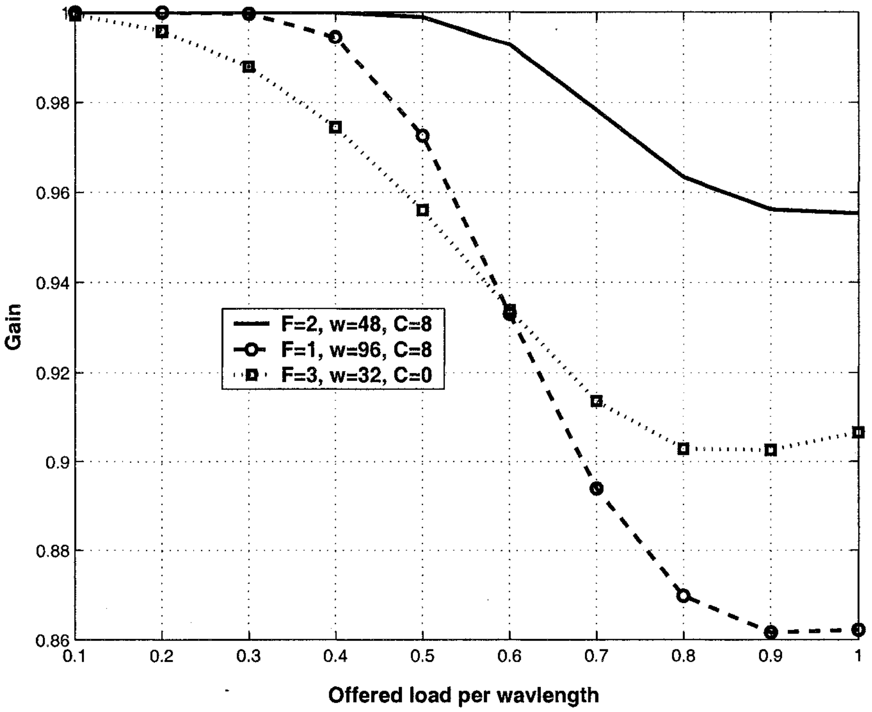

Figure 5.18: Performance gain vs. the offered load

5.5 Chapter Summary

A slotted optical packet switch performance with a share-per-link conversion bank was evaluated analytically and by means of computer simulation. The perfect match between the analytical and simulation results confirms the accuracy of the analysis developed. The results showed that using multiple fibers per link offers better packet loss performance due to its ability to accommodate more traffic. Therefore, the number of shared converters can be significantly reduced when using multiple fibers due to the improvement and the higher utilization the multi-fiber link offers. The worthiness of the one extra fiber is much higher than the worthiness of a single conversion device, which confirms that the performance improvement introduced by an additional fiber is much better than the improvement due to adding a converter.

Chapter 6

ALL-OPTICAL NETWORK SURVIVABILITY

6.1 Introduction

All-optical networks based on WDM technology can potentially transfer hundreds of gigabits per second of data on each fiber link in the network. However, the high capacity of a link has the drawback that a link failure can potentially lead to the loss of a large amount of data as well as connectivity and interfere with a large amount of traffic flow. Thus, the survivability performance of WDM networks is an important research and design issue. In general, network survivability is defined as the ability of a network to maintain or restore an acceptable level of performance in the event of failures, such as link failures or node failures [56]. Two types of survivability measures have been proposed: deterministic [43] and probabilistic [56, 14]. A deterministic survivability measure depends on the topology of the network and is concerned with the number of available paths between a pair of nodes. On the other hand, the probabilistic survivability measure depends on the probability of occurrence of different failures and also on the reliability of each component in the network. It is not easy to uniquely quantify the network survivability. For example, compare two network architectures A and B with blocking probabilities of 0.001 and 0.005, respectively; without additional information, it seems that network A appears to be more survivable than B. Suppose the steady state availabilities of networks A and B are 0.99942 and 0.99965 (5 hours and 3 hours down time each year); with this additional information, it seems that network B has better survivability than network A. Therefore Network survivability should include a system availability analysis to discover the cost due to system downtime as well as system failure impact analysis to dicover the transient performance degradation when failure occurs.

In this chapter, we develop a hierarchical survivability model for multi-fiber WDM networks that consists of availability and transient performance analysis. This composite model is used to evaluate network survivability in the presence of path failure. The availability analysis evaluates the steady state behavior of the network even when the available paths between nodes have common links using the algorithm proposed in [30]. These two models are combined to construct a hierarchical network survivability evaluation model.

6.2 Multi-Fiber Survivable WDM Network

In this section, the survivability of a multi-fiber WDM network with a dynamic traffic is investigated and evaluated. In multi-fiber WDM networks, the physical link between nodes consists of F parallel fibers. This configuration provides better performance in the case of no failure. But in the case of link failure, for example due to a fiber cut, all the fibers between the nodes may fail simultaneously. So the survivability of this implementation for a mutli-fiber WDM network is very poor and will suffer if any failure occurrs due to massive traffic loss.

First, we will present a new approach to implementing a survivable multi-fiber WDM network. This approach utilizes the networks physical topology and intermediate nodes to realize the survivability of the multi-fiber network. Such a design will carry with it the improved performance associated with multi-fiber design while also enhancing the survivability of the network. The transient performance analysis for a multi-fiber WDM network implemented using this new approach will not differ from the analysis of the multi-fiber networks that was presented in Chapter 4. The availability analysis for this mutli-fiber WDM network will be presented in this section.

6.2.1 Survivable Multi-Fiber Connectivity Topology

Our study is focused on a homogeneous multi-fiber network, in which the number of fibers in each link is the same and the number of wavelengths in each fiber is the same. An easy extension and a similar analysis can be applied to more general multi-fiber networks. Having F parallel fibers on the same physical link that is connected a pair of nodes in the multi-fiber WDM network provides a high blocking performance, but the survivability of this multi-fiber configuration is very low due to the failure of all fibers when a link failure occurs. Therefore, in survivable multi-fiber WDM networks, the multiple fibers between nodes have to be in different cables in physical topology even though the fiber connectivity topology is shown as a direct connection between nodes. This may mean having a passive connection at the intermediate nodes between source and destination. Figure 6.1a shows a sample physical topology, which provides three fibers between nodes. The fibers between nodes 1 and 2 are from three different cables, while nodes 5 and 6 could be passive connectors between these two nodes. If a link between nodes 1 and 2 fails, the traffic on this link can be restored using available fibers in other cables, which connect these two nodes. Figure 6.1b illustrates the fiber connectivity between nodes 1 and 2 as three direct connections between these two nodes. The fiber connectivity topology is considered in the transient performance evaluation and availability analysis. Our results show that the multi-fiber connectivity topology provides better performance and higher survivability for multi-fiber WDM networks.

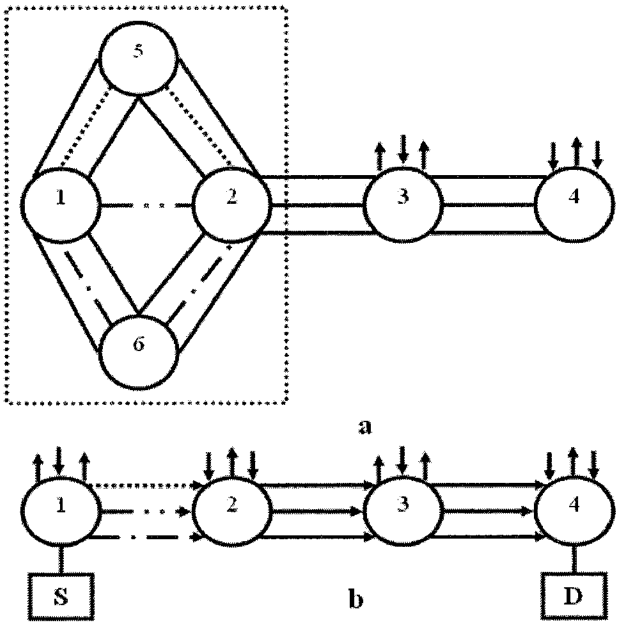

Figure 6.1: Fiber connectivity in survivable multi-fiber WDM networks a) Multi-fiber physical topology. b) Multi-Fiber connectivity topology.

6.2.2 Transient Performance Analysis

Since the same performance analysis for a multi-fiber WDM network that was presented in Chapter 4 holds for this new implementation of the multi-fiber WDM network, this section will give a brief overview of the transient performance analysis for this network. This transient performance analysis considers the fiber connectivity topology shown in Figure 6.1b for two scenarios with and without wavelength conversion. We assume the same number of fibers in all hops in the logical path between the source-destination pair of interest (S, D).

In the case that there are no wavelength conversion resources that can be utilized in the network, the blocking probability P is the probability that each wavelength in every fiber is used on at least one of the intermediate hops. P is thus given by:

$$P = (1 - (1 - \rho^F)^H)^w \tag{6.1}$$

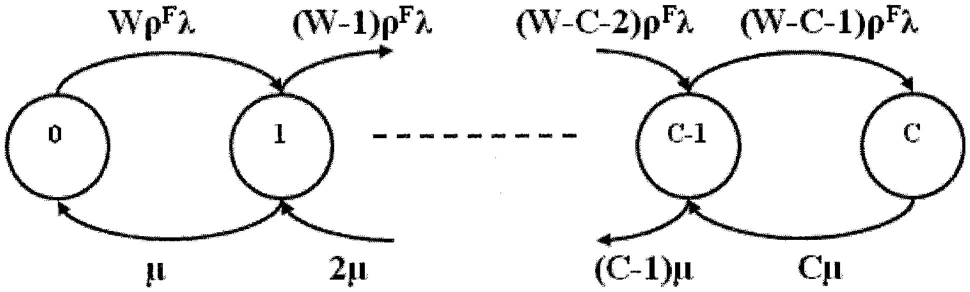

Figure 6.2: M/M/C/C model for share-per-node conversion bank.

where F is the number of fibers, w is the number of wavelengths per fiber, H denotes the number of hops along a path, $\rho = \frac{\lambda}{\mu}$ is the probability that a wavelength is used on a hop, and λ, μ are the arrival and the service rate per wavelength, respectively. ρ is also a measure of the load per wavelength.

If there are wavelength conversion resources that can be utilized in the network, the blocking probability P is the probability that all wavelengths in all fibers are busy on one of the intermediate hops or that there is no wavelength converter available in one of the intermediate nodes.

The probability of zero converters being busy, P_0, and the probability of all converters being busy, P_C, are derived from the $M/M/C/C$ model shown in Figure 6.2 to predict the behavior of the conversion bank. C denotes the number of wavelength converters in the wavelength conversion banks. W is the total number of wavelengths on all fibers, which is equal to $W = w \cdot F$. C can be defined with respect to W as $C = W \cdot DC$, where DC is a fraction that represents the number of converters with respect to W.

The request for a converter is the arrival rate per wavelength λ given that the same wavelength is busy in all output fibers with a probability of ρ^F. So, a converter request rate due to a single wavelength is $\rho^F \cdot \lambda$. Consequently, the transient rate from state i *(i converters are busy)* to state *i+1 (i+1 converters are busy)* is $(W-1)\cdot\rho^{F}\cdot\lambda$. P_0 and P_C for share-per-node WCS are derived as follows:

$$P_0 = \frac{1}{1+\sum_{k=1}^{C}(\frac{1}{k!})\cdot(\frac{W!}{(W-k)!})\cdot\rho^{k\cdot F+k}} \tag{6.2}$$

$$P_C = \frac{1}{C!}\cdot\frac{W!}{(W-C)!}\cdot\rho^{C\cdot F+C}\cdot P_0 \tag{6.3}$$

Since in such a model, conversion request is either accepted or rejected in its entirety, P_C depends on the number of converters, C, number of incoming wavelengths, and the load per wavelength, ρ.

The blocking probability obtained for share-per-node WCS is given by:

$$P = (1-((1-\rho^{F})+\rho^{F}\cdot(1-P_C)\cdot(1-\rho^{W-F}))^{H})^{w} \tag{6.4}$$

Table 6.1 summarizes all the different scenarios for the different switches considered in this paper according to their blocking probabilities.

Switch	Blocking Probability
Multi-fiber share-per node	$(1-((1-\rho^{F})+\rho^{F}\cdot(1-P_C)\cdot(1-\rho^{W-F}))^{H})^{w}$
Single-fiber partial conversion	$(1-((1-\rho)+\rho\cdot(1-P_C)\cdot(1-\rho^{W-1}))^{H})^{w}$
Dedicate WCS	$(1-(1-\rho^{W})^{H})^{w}$
Multi-fiber no conversion	$(1-(1-\rho^{F})^{H})^{w}$
Single-fiber no conversion	$(1-(1-\rho)^{H})^{W}$

Table 6.1: Blocking probability connection requests.

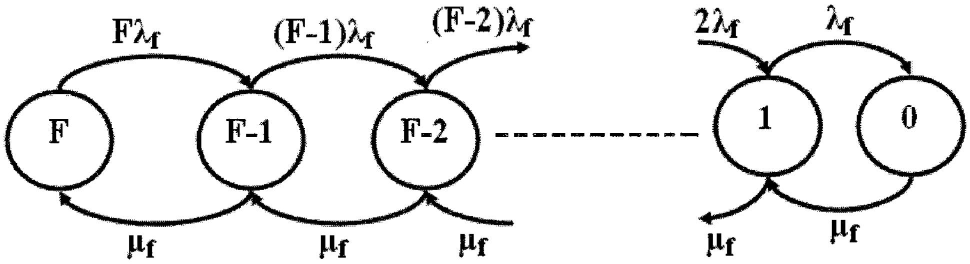

Figure 6.3: General availability model for multi-fiber WDM networks.

6.2.2.1 Multi-Fiber WDM Networks Availability Analysis

In this section, we construct a continuous-time Markov chain for paths availability between pairs of nodes, and solve it for the steady state probability where the path is up. We assume the same number of fibers in all hops in the path between the source-destination pair of interest (S, D). Assume that there are F fibers available between nodes and the times to link failure and repair are exponentially distributed, with mean $\frac{1}{\lambda_f}$ and $\frac{1}{\mu_f}$, respectively. Also assume that a single repair facility is shared by all the links. The availability model is then a homogenous CTMC with the state diagram shown in Figure 6.3.

Here, the state index denotes the number of nonfailed fibers in the system. The steady-state probability for the number of nonfailed fibers in the system is given by:

$$\pi_i = \frac{1}{i!}(\frac{\mu_f}{\lambda_f})^i \pi_0, \quad i = 1, 2, \cdots, F \tag{6.5}$$

where the steady-state system unavailability is:

$$U = \pi_0 = \left[\sum_{i=0}^{F} \frac{1}{i!}(\frac{\mu_f}{\lambda_f})^i \right]^{-1} \tag{6.6}$$

6.3 General Hierarchical Survivability Model

As mentioned above, a network survivability performance evaluation should include *a)* system availability analysis to discover the cost due to system downtime, and *b)* system failure impact analysis to discover the transient performance degradation when failure occurs. Pure performance evaluation of the network without considering failure-repair behavior in the system is too optimistic, and availability analysis without considering performance parameters of the network is too conservative. In this section a hierarchical network survivability, which includes both performance and availability analyses of the network, is developed.

Gracefully degrading systems may be able to survive the failure of one or more of their active components and continue to provide service at a reduced level. One of the most commonly used techniques for modeling of gracefully degradable system is the Markov reward model (MRM) [59]. The Markov Reward Model is a Markov chain with a reward rate (real number) assigned to each state. It is assumed that the system can be in one of a specified set of states. Each state is associated with a specific performance level as a reward rate. Over time, the system moves through the states as failures and repairs occur.

The path availability between pairs of nodes in the network is modeled, and the model is solved for the steady state probability that the path is up. The steady state probability that there are i non-failed paths in the network is denoted as π_i. Then an availability model is turned into a Markov Reward Model (MRM) [59], where the reward rates come from the system transient performance model. The reward rate for each state can be found using the transient failure impact analysis of the network, and is denoted as r_i. The reward rate, r_i, can be calculated using the WDM blocking model that was developed in Chapter 4 and presented in this chapter in Section 6.2.2. The general formula for this model is given by Expression 6.7.

$$r_i = (1 - ((1-\rho^i) + \rho^i \cdot (1-P_C) \cdot (1-\rho^{W-i}))^H)^w \tag{6.7}$$

where $1 \leq i \leq F$

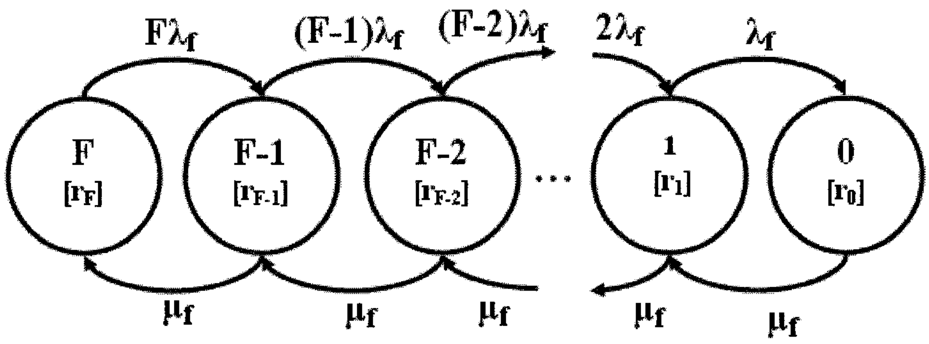

Figure 6.4: General network survivability model for multi-fiber WDM networks with and without conversion.

Transient failure impact analysis usually depends on the topology, capacity constraints and restoration strategy of the system. Figure 6.4 illustrates the proposed general model of network survivability. By combining the availability model and transient behavior of the network in the presence of failure as a MRM, we can find the survivability of the network. Attach a reward rate, r_i, to the state i of the availability model, and find the total loss due to unavailability of path(s) and also due to capacity constraints on the restoration paths between nodes. We call the total loss the susceptibility of the network, and it is given by:

$$Susceptibility = \sum_{i=0}^{k} r_i \pi_i = \sum_{i=1}^{k} r_i \pi_i + \pi_0 \tag{6.8}$$

In state π_0, no path is available between the nodes; therefore, all the traffic is lost and the reward rate is 1. Now we define survivability as:

$$Survivability = 1 - Susceptibility \tag{6.9}$$

6.4 Multi-fiber WDM Network Survivability Results

In this section, we discuss the network survivability improvements in the multi-fiber environment over the single-fiber case considering all network parameters. The procedure for finding the susceptibility and survivability of the network involves the following steps:

- Using equation 6.7, find the blocking probability for each case of F fibers, given the number of converters, number of hops, offered load and number of wavelengths per fiber (reward rates).
- Solve the steady state availability model to find the probability of states using equations 6.5 and 6.6.
- Find susceptibility and network survivability using equations 6.8 and 6.9.

For example, assume there are 4 fibers available between source and destination in Figure 6.1b. The blocking probability for this case can be found using Expression 6.7 knowing the number of converters, number of hops, offered load and number of wavelengths per fiber. Using the same procedure, the blocking probability for cases of 3, 2, and 1 fibers can also be calculated. These results are used as the reward rate in Expression 6.8. π_0, π_1, π_2, π_3 and π_4 can be found using Expressions 6.5 and 6.6. These results are combined to find the susceptibility and survivability of the network using Expression 6.8 and 6.9.

We assume a general network topology and single source-destination pair with fixed routing algorithm between source and destination. The number of wavelengths on all the fibers is 16 (W=16). In Figure 6.5, we study the effect of availability on the survivability of the network by changing the repair rate. We assume that the number of fibers is 4 ($F = 4$), the number of hops is 15 ($H = 15$), and the wavelength conversion is 50% ($DC = 0.5$). In the first scenario, we analyze the performance of the network without considering failure-repair behavior in the system, which indicates an availability of 1. In scenarios 2 and 3, we consider different failure rates, μ_f equals 0.1 and 0.05, with $\lambda_f = 0.01$. The performance and availability models are combined to find the survivability of the network. As we decrease the steady state availability by decreasing the repair rate, the survivability of the network decreases. These results show that system survivability

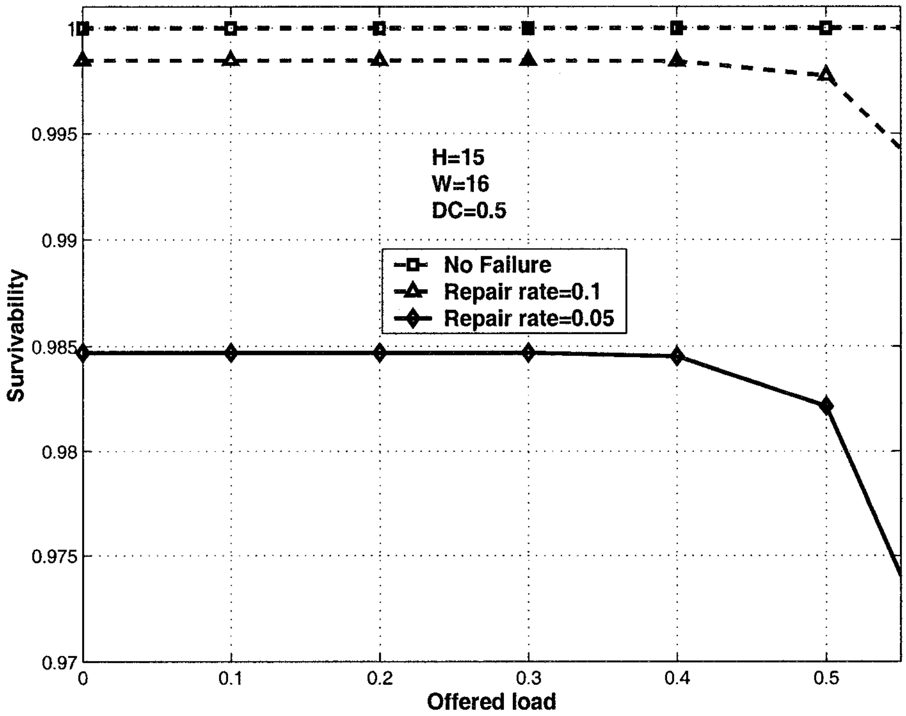

Figure 6.5: Survivability for different steady state availability vs. offered load.

performance evaluation should include both system availability analysis and performance evaluation of the network where pure performance analysis tends to be optimistic.

In Figures 6.6 and 6.7, the source-destination susceptibility and survivability are plotted as a function of the offered load for different numbers of fibers assuming 50% wavelength conversion. The amount of cross-traffic is increased when the number of fibers is increased for the same ρ, as ρ is the normalized load with respect to capacity. More fibers between a source-destination pair can provide additional routes between the nodes and increases the survivability of the network in case of failure. As we see in Figure 6.7, having two fibers instead of one can improve the survivability of the network significantly. Even with a load of 0.6, the survivability of the traffic between source and destination is still 0.95.

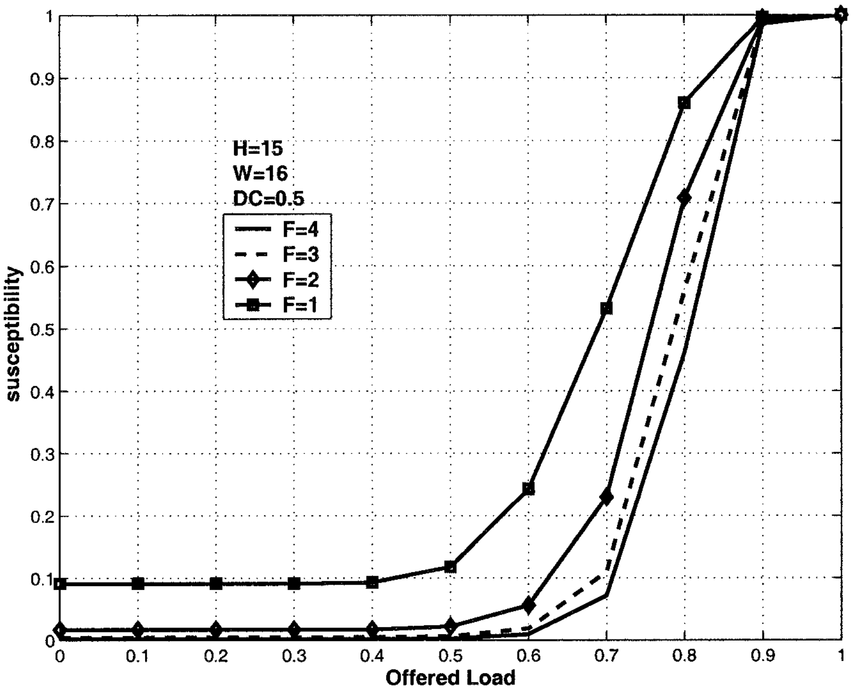

Figure 6.6: Susceptibility vs. offered load.

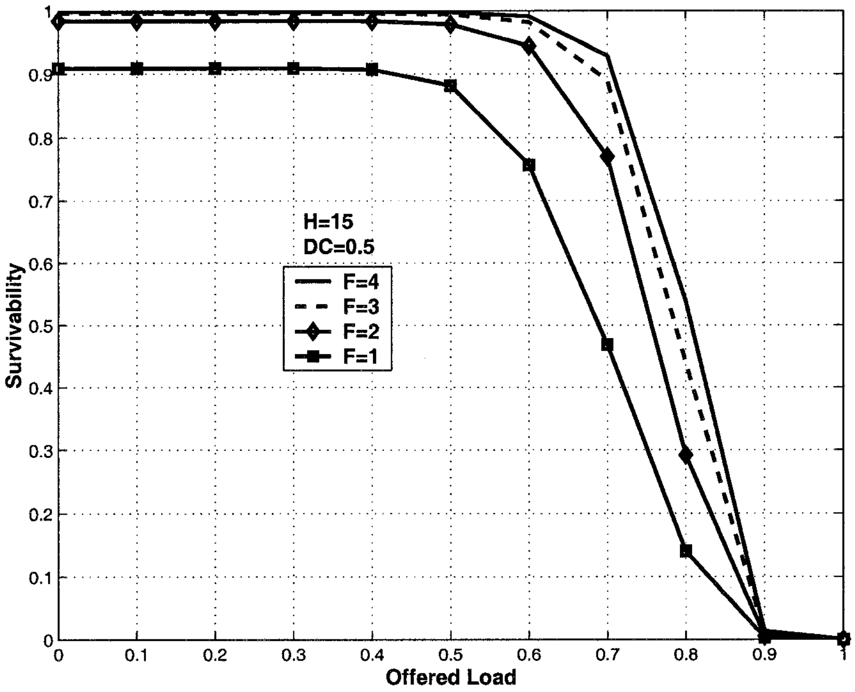

Figure 6.7: Survivability vs. offered load.

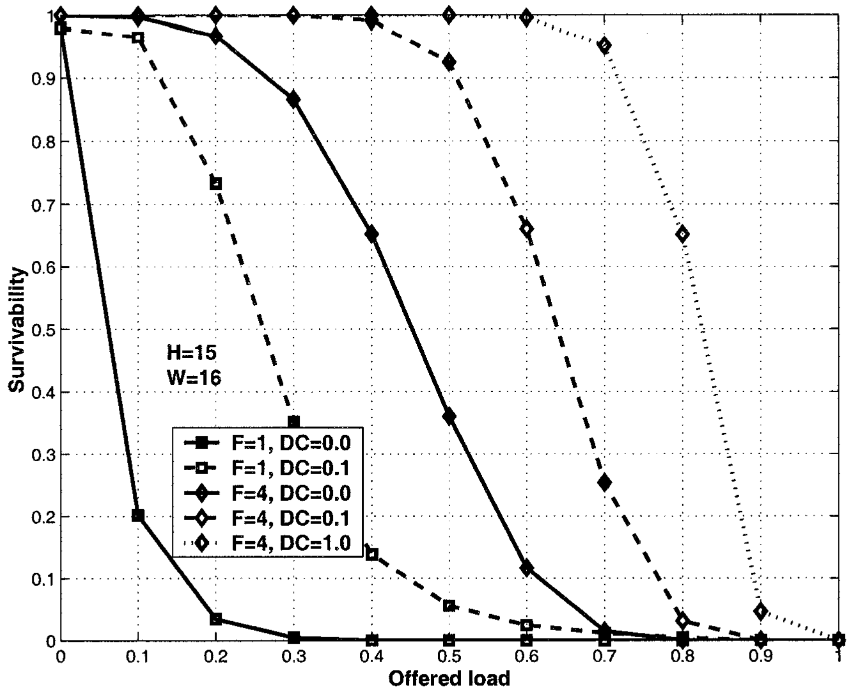

Figure 6.8: Survivability under different degree of conversion vs. offered load.

In Figure 6.8, we show the effect of wavelength converters on the survivability of the network. Having one fiber and partial conversion (10%) can improve the network survivability. Adding more fibers, even with no conversion, can improve network survivability more significantly. If we increase the number of fibers to 4 and have partial conversion (10%), network survivability improves much more significantly at higher load.

In Figure 6.9, we increase the size of the network by increasing the number of hops. Increasing the number of hops decreases network survivability as expected. But using two fibers instead of one can improve network survivability in a large network.

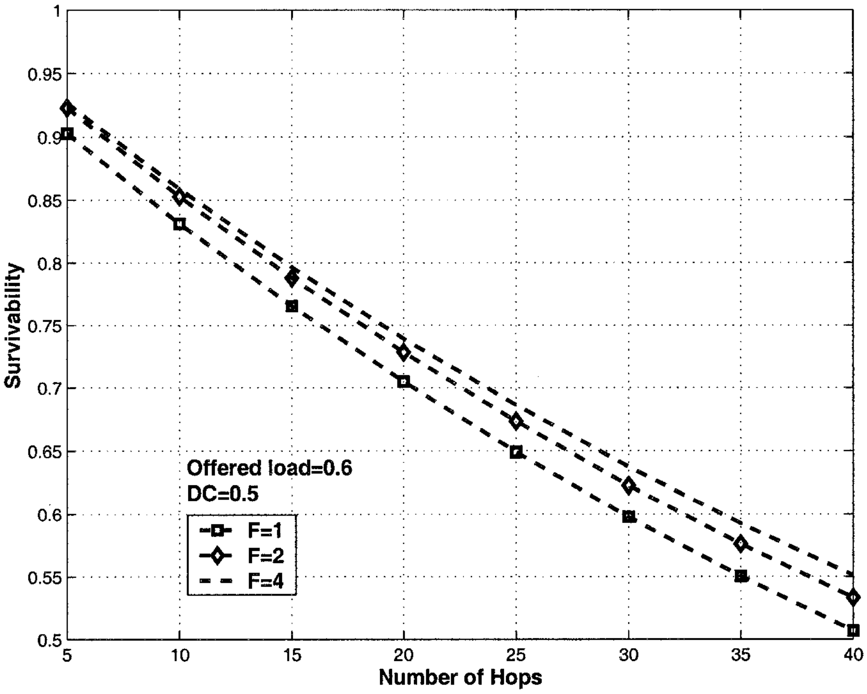

Figure 6.9: Survivability vs. number of hops.

6.5 Chapter Summary

A hierarchical model to evaluate system survivability performance was developed. The end-to-end performance of circuit-switched all-optical networks in multi-fiber multi-hop environments is modeled and evaluated with and without wavelength conversion. These models were utilized to evaluate the performance degradation when a failure occurs. The performance degradation model and the availability analysis model were combined to construct a hierarchical network survivability evaluation model.

The study showed that the use of multiple fibers at the connectivity topology level of WDM networks provides additional network capacity and increases network survivability by reusing the same set of wavelengths in multiple fibers in case of link failure. The survivability of the WDM network decreases as the steady state availability decreases. This result proves another objective goal for this dissertation that the accurate survivability measure of the optical network has to consider both the performance degradation during the time of failure and the resources availability at the time of failure. A new approach for a more survivable optical network was presented and evaluated. This new approach provides multiple fibers between the nodes, but these fibers are not on the same physical link and have the intermediate nodes behaving as passive connectors.

Chapter 7

OPTICAL NETWORK SIMULATORS

7.1 Introduction

The all-optical network blocking performance is one of the main measures that can be used to quantify the capability of the optical network to be considered as a promising solution for backbone transport network limitations. The performance evaluation for any system could be conducted by one of the following techniques:

1. Measurement
2. Analytical modeling
3. Simulation

Measurement is not a feasible method for the performance evaluation of many systems since this technique requires either the system or testbed of the system to exist. Therefore, evaluating the all-optical network performance through measurement is limited to available testbeds. Moreover, the cost of building an optical network testbed is very high. The performance evaluation conducted through measurement sometimes does not give an accurate measure of the real network performance since it reflects the network parameters setting at the time of the measurement. Also, changing the network parameters and gathering new measurements requires a very long time. These limitations make the all-optical network performance evaluation using measurement an improper performance evaluation technique.

Analytical modeling is the second technique for performance evaluation. Analytical modeling does not require a testbed for the system under consideration. Also, it provides an insight about the possible system performance in a very short time. Developing an analytical model for a system depends on many assumptions and simplifications which affect the accuracy of the evaluation. Also, in many cases it is very difficult to develop an analytical model for a system due to system complexity and number of parameters. Therefore, analytical modeling gives an insight and approximate performance evaluation for simple systems.

Using simulation is a useful, easy, and convenient technique for characterizing system performance. As with analytical modeling, the simulation technique does not require the system to exist. But, the simulation requires fewer assumptions and less simplification, which makes it closer to reality. Varying the network performance parameters in the simulation technique can be done easily. So, performance assessment for an all-optical network using a simulation modeling technique is more convenient and easy to conduct.

In this chapter, different techniques for building multi-fiber optical network simulations will be presented and discussed in detail. In this research, two different simulators were built to evaluate the end-to-end blocking performance for wavelength-routed and slotted optical networks. Also a wavelength conversion-ranging simulator was built to investigate the effect of the conversion ranging on the optical node performance. Further more, the performance of a slotted optical node with share-per-link conversion-sharing schemes was explored by developing a discreet event simulator that can assess the node performance under different parameter settings. The simulators were used to investigate the multi-fiber optical network performance under different network scenarios. These simulators were written in C++ computer language as independent multiple modules, which are easily modified and tested. The main components of these simulators will be described in the following sections.

7.2 Main Components of Multi-Fiber Optical Network Simulators

The object-oriented model that was followed in the implementation of the simulators leads to having a common component for these simulators. That common component will be described and discussed in the following subsection, and then the different components for the different simulators will be explained separately.

7.2.1 Random Number Generator

Any simulator that simulates the dynamic nature of a system relies on a random number generator. Generating a random number with a specific distribution involves two steps. First, it generates a uniform random number between 0 and 1, and then it transforms this uniform random value to the desired distribution. In this section, generating a uniform random number will be explained [22].

All simulators were developed in this research utilizing a common random generator module. This module is based on the Multiplicative Linear-Congruential Generator (MLCG) [5]. MLCG generates the nth value according to the following equation:

$$x_n = a \cdot x_{n-1} modm \tag{7.1}$$

where a and m are the multiplier and modulus of the generator, respectively. The modulus (m) has to be a power of 2 so that the time required to compute the mod operation will be negligible. In the current implementation for this random generator $a = 7^5$ and $m = 2^31-1$. This setting of the MLCG random generator will produce the full period of the random values. This period is equal to *2,147,483,646*. To generate the first random value x_0, a selected seed is passed to the MLCG and then the current random value is used as a seed for the next value and so on.

7.2.2 Event

The activities that have to take place at a certain time on the optical network are defined as events. These events describe the optical network resources status, or packet processing. Any optical network simulator has to have at least the following basic events:

1. Packet event
2. Resources reservation event
3. Resources release event

In our simulators we define a common object (packet object) for these events. The packet object was implemented as a class data structure. Figure 7.1 shows the packet object class and the main fields in this class.

According to the *Event-Type* field, all events can be classified into two main categories, *Packet event* or *Resources event.* In the case of packet events, the *Event-Type* field is set up at the time of packet generation to 1; if the event is a resources event, this field set to 0. The *Initial-Time* field for a new generated packet is set to the simulation current time. In the case of a resources event, this field is set to 0. The *Time-of-Arrival* is set to 0 for new generated packets and will be updated while the packet traverses through the optical network by a cumulative value of the transmission and propagation time to the next node. This event field will be set to 0 for the resources event. The *Processing-Time* field is set according to the distribution that describes the packet arrival, or according to the resources holding time in the case of the resources event. The wavelength field contains the wavelength that needs to be reserved or released in the case of the resources event or that was utilized by the packet on the previous hop. To keep track of the packets, *Source-Node*, *Destination-Node*, and *Current-Node* fields are defined. The

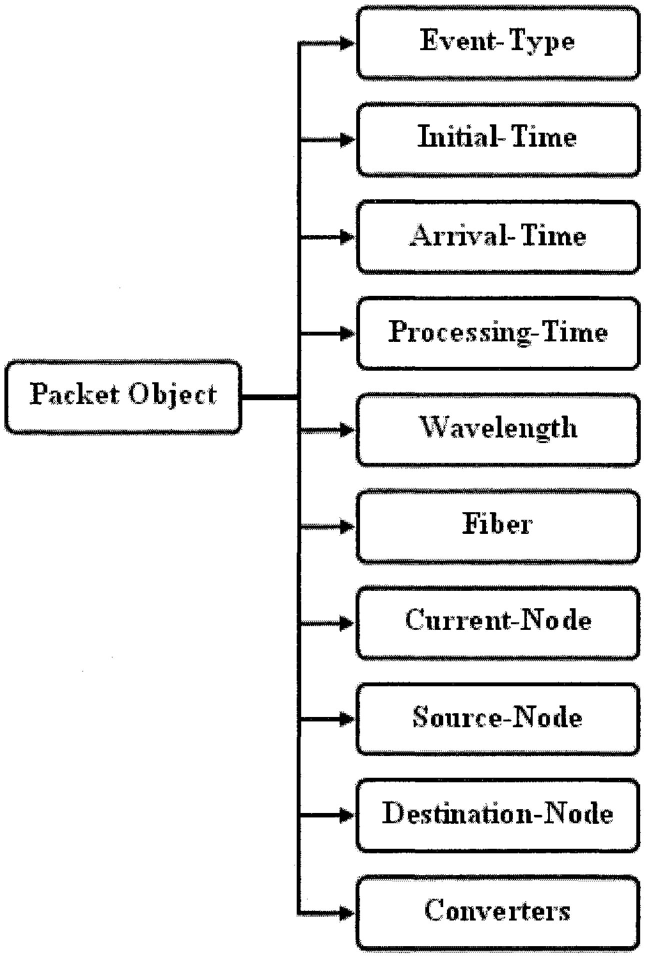

Figure 7.1: Packet class and the main fields.

Current-Node event field is utilized in the case of a resources event to identify the node at which the resources status has to be updated.

The events that are generated during the running of the simulation are listed in a time-ordered list. This list is called an event list. Initially, the event list is empty and then every generated event is inserted on the list according to the *Processing-Time* event field. Figure 7.2 shows a schematic presentation of a typical event list. The *Header* pointer at the top of the list always points to the next event that will be processed and the *Tail* pointer at the bottom of the list points to the end of the list. If *Header* and *Tail* point to the same memory location, that means

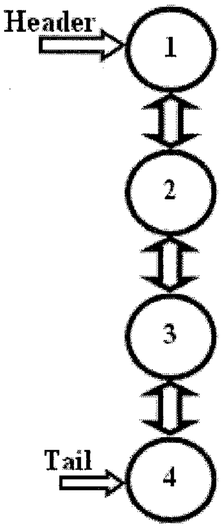

Figure 7.2: Event list schematic presentation.

there is only one event in the list. When the event that pointed to by the header is picked and processed, one or more of the following events could be generated if this event is a packet event:

1. Resources release event if the packet gets forwarded.
2. New packet event if the event reaches the destination or is dropped at one of the intermediate nodes.
3. Arrival packet event if the event gets forwarded and did not reach the destination

The optical network resources will be affected by processing a resources event. So if the *Header* points to a resources event, one of the following actions will be executed to update the network resources status:

1. Release the wavelength specified by the *Wavelength* field on the fiber indicated by the *Fiber* field and the converter indicated by the *Converter* field if a converter was reserved by the packet event, which generates this release event.

2. Reserve the wavelength specified by the *Wavelength* field on the fiber indicated by the *Fiber* field and the converter indicated by the *Converter* field if the packet event, which generates this release event, utilizes a converter.

7.3 Unslotted Optical Node Simulator

This simulator is implemented to study the blocking performance of a single optical node deploying different wavelength conversion configurations and sharing schemes. This simulator utilizes the above simulation common components. In addition. other modules are implemented to reflect the practical behavior of different node architecture. These modules are:

- The wavelength assignment module: In this module, two different wavelength assignments are implanted (first-fit wavelength assignment and random wavelength assignment).
- Packet-generating module: In this module, the inverse transformation of the uniform random generator explained above is implemented to generate random values that follow exponential distribution. The method is utilized by the simulator to find the time between successive packets' arrival as well as the packet size.
- Ranging conversion module: This model was implemented to study the effects of the wavelength converters-ranging limitations on the node performance. Four different ranging functions were implemented in this module. These ranging functions are:
  - Any-to-Any: In this function, the general conversion technique was implemented such that a converter can have any wavelength as an input wavelength and this wavelength can be translated to any other wavelength.

- Any-to-Range: In this ranging function, the input to the converter could be any wavelength out of the total input range but it can translate to a pre-specified set of wavelengths.
- Range-to-Any: This function was implemented to study the effect of the ranging on the input side. In this function, the total number of wavelengths at the input side of the switch is divided into a number of sets equal to the number of the converters that are available at the node. Then every set is assigned to a converter. Therefore, every converter can accept only those wavelengths assigned to its input and can translate them to any output wavelength.
- Range-to-Range: In this function, the ranging limitation was imposed on both sides of the converter. This means every converter can accept only those wavelengths assigned to its input and can translate them to a pre-assigned range of wavelengths.

Figure 7.3 shows a general flowchart for this simulator. Node initializtion function is invoked when the simulator starts. In this function, the initial parameters such as number of inputs, number of outputs, number of fibers, number of wavelengths per fiber, number of converters, inter-arrival time and mean packet size are fed to the simulator. The status of all output wavelengths and all converters is set to one, which means all these resources are available. The random generator seeds are also initialized in this function. The event handler function is then called to start generating traffic on all wavelengths in all inputs and to start building the initial event list. This event list is fed to the event list processing function to start the actual simulation. The header of the initial event list is picked and processed and its *Time-to-be-Processed* field is captured as the simulation starting time. When a packet event is dropped, a new packet is generated and inserted

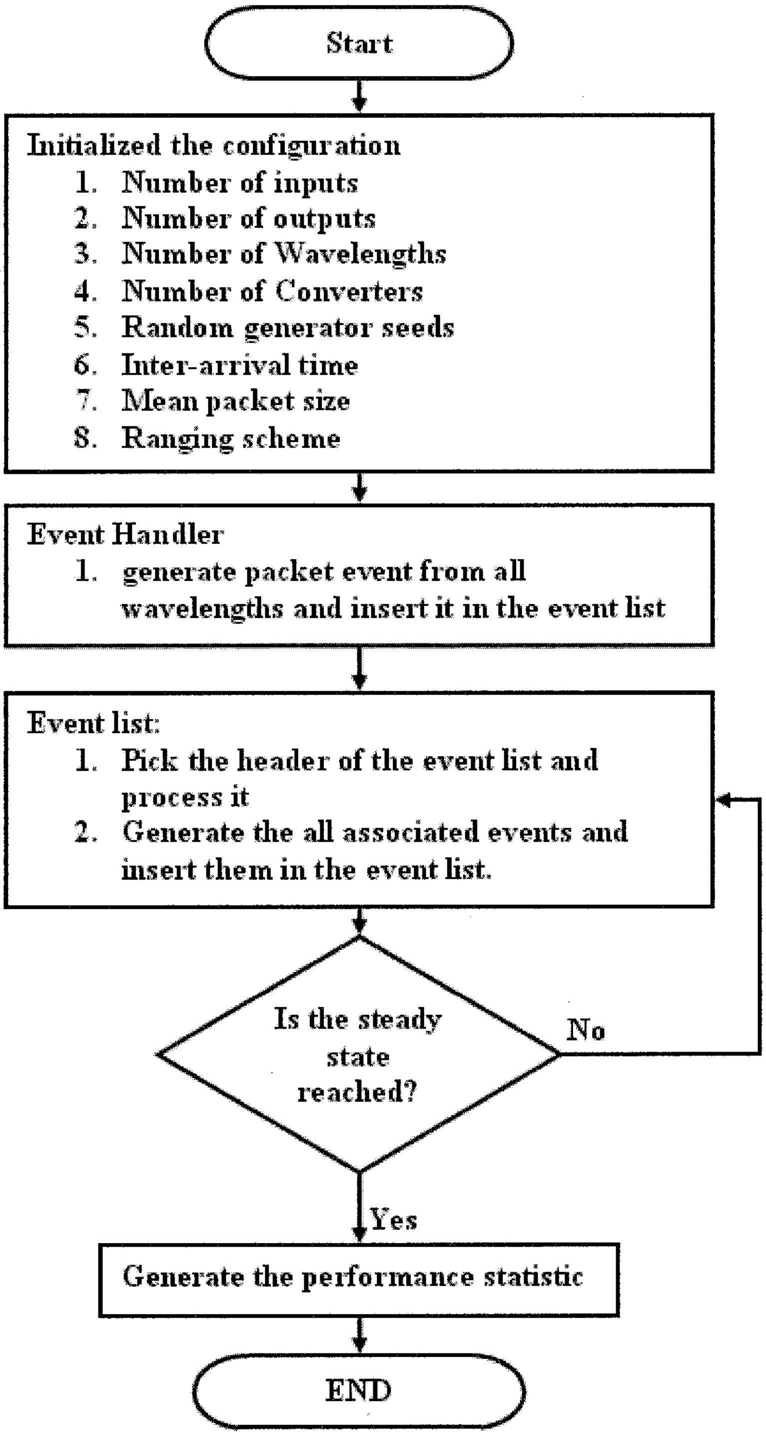

Figure 7.3: Unslotted optical node simulator flowchart.

into the event list according to its *Time-to-be-Processed* field. When a resources event is processed the node resources status gets updated. The number of packets dropped is accumulated. When 100,000 packet events have been processed, the packet loss probability is calculated according to the following equation:

$$Blocking\ Probability = \frac{Number\ of\ packets\ dropped}{Total\ number\ of\ packets\ processed} \tag{7.2}$$

The point when the variation in the packet loss probability from the previously calculated value is less than 1% is considered as a starting point of the steady state of the simulator. So, the statistical data are cleared and the simulator runs for another 100,000 packet events and the collected statistics in this simulation window are considered as the final results of the simulation.

7.4 Slotted Optical Node Simulator

This simulator is developed to study a single optical node that operates in the slotted time mode. A share-per-output links conversion scheme is considered in this simulator as a packet-contention resolution. The common simulators components are utilized by this simulator. In this simulator, the node initialization function is called at the beginning of every time slot since the packets switching took place in a slot-by-slot manner, and the packet size is fixed so the output wavelengths and all other switch resources are initialized to free. The packet drop in this switch will occur only if more than one packet needs to leave the switch on the same wavelength through the same output port and there is no converter free or all output wavelengths on the requested output are busy. The simulator runs till it reaches the steady state, as explained in the previous section, before it starts collecting the statistical data. The schematic flow chart in Figure 7.4 illustrates the general functions and their tasks in this simulator.

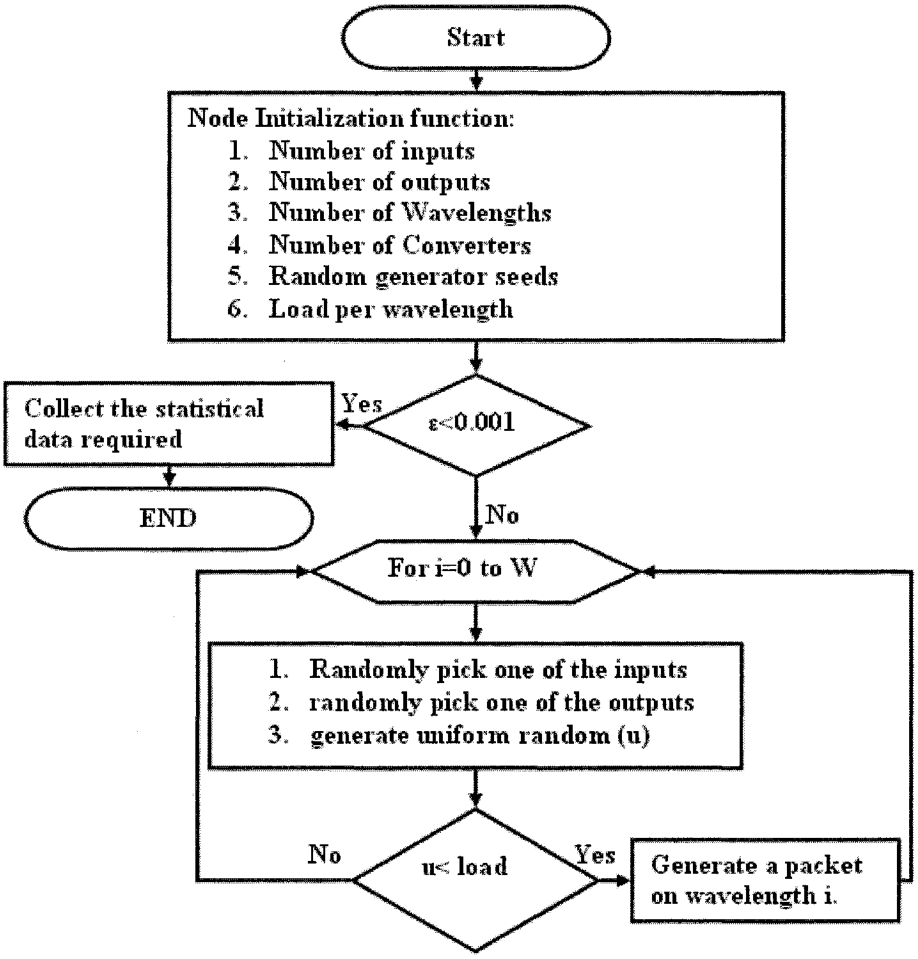

Figure 7.4: Slotted optical node simulator flowchart.

7.5 Optical Network Simulator

This simulator was developed to study the effects of the optical network parameters on the whole network performance. Two different versions of this simulator were implemented; the first one was implemented for an unslotted optical network and the second one for a slotted optical network. In both versions, the fixed routing algorithm between a single source-destination pair was considered and only a random wavelength assignment algorithm was utilized in these simulators. Figure 7.5 shows the network topology that was considered in these simulators.

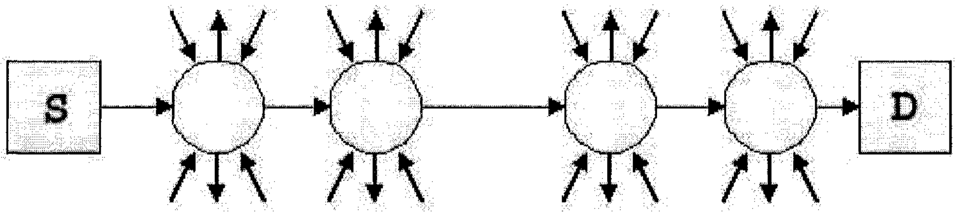

Figure 7.5: Simulated network topology.

The unslotted optical network simulator utilizes the simulator that was described previously for the unslotted optical node. The following extra functions were implemented to capture the real network behavior:

- *Construct_Topology*: In this function, a network topology file is read and the nodes connectivity topology is constructed by declaring the next node and previous node for every single node in the network. Also, number of wavelengths, fibers and distance between nodes are specified. The propagation delay needs to be calculated to be able to update the *Arrival-Time* field of the packet event. For that reason, the distance between nodes is declared. The propagation delay of a packet between two nodes is calculated according to the following equation:

$$Propagation\ Delay = \frac{Physical\ distance\ between\ nodes}{The\ light\ speed\ in\ fiber\ cable} \tag{7.3}$$

- *Get_Path*: In this function, a grid data structure, as shown in Figure 7.6, is constructed for every packet event under the processing step. This data structure is arranged in columns and rows. The column represents the status of a wavelength, either free or busy, in all hops and the rows represent the status of all wavelengths on a specific hop. The box at the intersection of row i and column j gives detailed information about wavelength j in

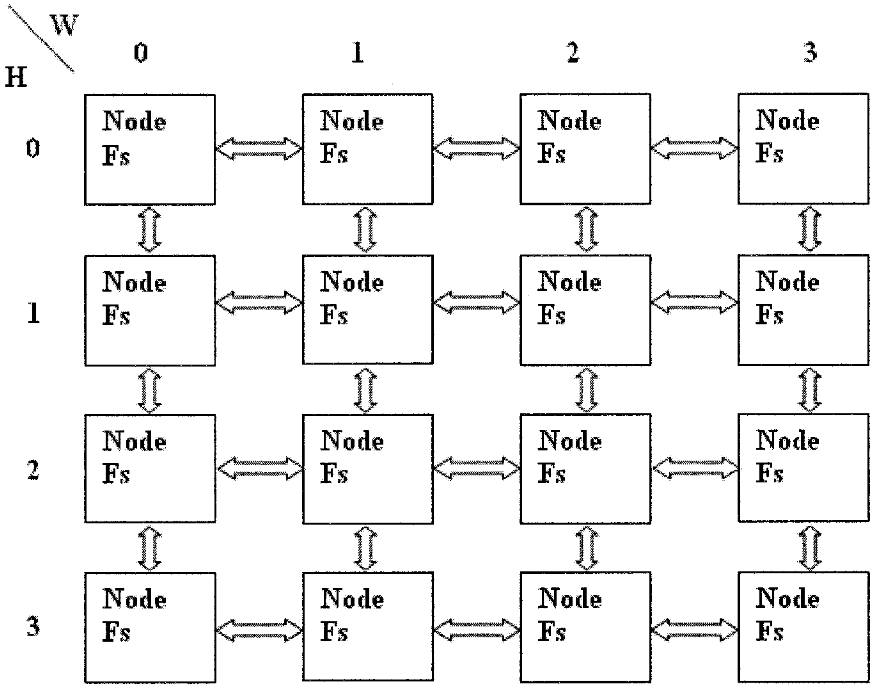

Figure 7.6: Path grid data structure.

all fibers on hop *i*. The first row in this data structure represents the status of all wavelengths at the source node of the packet while, the last row represents the wavelength status in the last node before destination. After constructing this data structure, a new process will start to search for the resources that are necessary to establish the lightpath for the current packet event. The lightpath establishment process was implemented to reduce the use of conversion resources as much as possible. This process was done in two steps. The first is to try to find a wavelength that is available on all hops by searching this data structure column by column. If the step of finding a continuous wavelength from source to destination failed, then the search for the longest continuous lightpath possible starts. At the end of this lightpath, a conversion resource has to be utilized to finish the lightpath establishment process. If the conversion resource cannot be utilized at the node where this lightpath ended, the next longest wavelength is sought, and so on untill the lightpath is established. The lightpath request will be blocked if a single row of the grid data structure has no wavelength available or the lightpath establishment process explained above is terminated with a fail flag.

- *Packet_Process*: All packet event information is updated in this function. A comparison between the packet's current node and destination node is performed at the beginning of this function to identify the status of the current packet event. If the destination of the current packet event is equal to the next node of the current node, network resources that are necessary for forwarding this packet on the last hop (hop between current node and destination) will be reserved. Then a release event for these resources and a new packet event from the same source and same wavelength will be generated. These two new events will be inserted into the event list and the current packet event deleted. On the contrary, if the next node of the current packet event is not its destination, the *Arrival-Time*, *Time-to-be-Processed*, and *Current-Node* fields will be updated and the network resources required by this packet event will be reserved. A release event for these resources will also be generated. These new events will be inserted into the event list according to their *Time-to-be-Processed* fields.

- *Release_Recourses*: If the header pointer of the event list points to a resources release event, this function will be invoked. The wavelength, fiber, node number, and the converter number, if it is set to any value other than -1, will be extracted from this event at the beginning of this function. Then the status of these resources is updated to free and this event is deleted from the event list.

The slotted version of this simulator works similarly to the unslotted version, as explained above. The only difference between these simulators is that the grid data structure in the unslotted version is not necessary any more becuase each node in the network operates in a slotted fashion. During every time slot, each node checks its inputs for arrived packets and processes those packets, which means no release event will be generated in this type of simulation since the packet size is fixed. To free the resources, the Release-Resources function is called at the beginning of every time slot.

A garbage collection function was developed in all these simulators to free the memory locations reserved by events processed.

7.6 Simulator Verification

Due to the large verity of the optical nodes and network configuration settings and the huge number of parameters that need to be considered in the development stages of these simulators, a number of verification techniques were used in this work. All or some of these verification simulation techniques are applied to every simulation discussed above. These verification techniques are discussed in the following subsections.

7.6.1 Modular Programming Technique

The modular programming technique is a general programming technique used to reduce the effort of verifying and correcting the errors in huge program projects. This technique is based on dividing the whole simulation model program into modules or subprograms so each subprogram or module can be verified independently; sometimes a dummies program is needed to verify a subprogram. In all simulators developed in this research, the modularity technique was followed. For example, different class structures are used for different entities in the simulation such as node class structure, packet class and converters class. In all these classes, all fields that are related to individual entities are defined as private parameters for these classes, so these fields cannot be changed from outside the class environment; also, interface functions for these classes were developed to allow other simulation subprograms to change these fields.

Also, the classes were tested individually by implementing print function inside every class to print the class initial setting, and during simulation time the class fields print to check to consistency of changing the fields. In addition, at the first stage of implementation of each simulator, the forced change inside each one was made and printed to verify the correctness of the changing that took place.

7.6.2 Trace Technique

The trace technique is one of the most powerful verification techniques in software development process. The trace technique of any program is based on printing any changes that took place after every statement in the code. In the simulation programs, this step could be done after any event processed.

During the development of every simulator in this work, a print function was developed and invoked before and after every event was processed. This print function prints out the detailed information about the event that is going to be processed such as *Arrival-Time*, *Time-to-be-Processed*, *Current-Node*, *Source-Node*, *Destination-Node* and the node/network resources status. After the event gets processed, this detailed information is printed again and compared to the event and resources initial values to see if there are any unexpected changes and to check the consistency of this operation for all monitored events. All new generated events are checked to find out if they are inserted appropriately into the event list. The whole simulator operations are suspended until a keyboard key is hit so that event-by-event information can be verified. Another verification done in this print function is to compare the total number of packets generated untill the time this function is invoked with summation of the dropped and forwarded packets. This function is deleted from the simulators after they are finalized and verified.

7.6.3 Test Boundary Conditions Technique

In this test, the computer program is tested by running a simple set of inputs parameters so the output of the program could be verified. This technique is used to verify all simulators developed. For example, a single wavelength, single fiber and one converter are set as the input parameters for the simulator. The converter utilization and the busy period of the wavelength are observed. The converter utilization is expected to be zero since there is only one wavelength on the output, so it cannot be utilized by any packet. Also the wavelength busy period is anticipated to be equal to the load. In the case of the network simulators, one hop with one wavelength and one converter is used as an initial value for the simulation parameters. In this case, simulation blocking has to be equal to the single node simulator in order to verify the correctness of the simulator.

The simulators run at the extreme boundaries settings. First, the number of wavelengths is set to zero, which means all packets generated will be dropped. Therefore, the blocking probability is equal to one. Another boundary condition is verified by setting the number of fibers to zero and the blocking probability is observed. The other extreme condition that these simulators are tested on is done by manually setting the wavelengths and all other resources to free; in this case, the number of packets dropped will be zero, so the blocking probability will equal zero in this case.

7.7 Chapter Summary

These simulators are used to evaluate the optical network performance that operated in a circuit-switched or packet-switched fashion. The results obtained by these simulators are used to verify the accuracy of the analytical models as discussed in the previous chapters. The common components of the simulators are presented in the header files in Appendix C. The wavelength-routed optical network simulator is shown in Appendix E. Finally, the slotted optical node simulator is presented in Appendix G.

Chapter 8

CONCLUSION AND FUTURE WORK

This research considers the multi-fiber optical network configuration, where every pair of switching elements in the network is connected using multiple-fiber links. Each fiber carries the same set of the wavelengths. Using multiple fibers on each link in WDM networks, which helps overcoming the wavelength continuity constraint in the wavelength-routed network as well as the packet contentions in the optical packet-switched network, has become a promising solution of the ever growing backbone transport network traffic. The performance and cost trade-offs of using a multi-fiber environment as physical implementation of the optical network was investigated in this dissertation. Different optical network performance parameters such as number of wavelengths, number of fibers, lightpath length and different conversion options were considered in both performance evaluation, survivability analysis, and the cost trade-offs study. The optical packet switch operating in a time-slotted fashion was considered and evaluated under multi-fiber implementation. In addition, the optical network survivability was evaluated through analytical and simulation based models. The significances and the outcome of this research will be explained in some detail in the following sections.

8.1 Wavelength-routed Optical Network

The study showed that employing multiple fibers in multi-hop wavelength-routed networks reduces the end-to-end blocking probability by reusing the same set of wavelengths in multiple fibers and by increasing the availability of the end-to-end paths. Increasing the resources availability in the multi-fiber environment by reusing the same set of wavelengths enhances the optical network throughput, and adds more configuration flexibility and scalability to the optical network infrastructure. This configuration flexibility comes from the reduction of the wavelength continuity constraint effects due to the ability of the network to set up multiple lightpaths on the same wavelength in different fibers on the same hop. Throughput always increases along with the number of wavelengths, which can provide additional network capacity to handle the traffic. The use of multiple fibers compensates for the utilization degradation caused by the increase in the hop count that a path traverses. It also adds more flexibility in increasing network reach (diameter) without affecting utilization significantly. This flexibility comes from the fact that every output link within the switch has multiple fibers that have similar sets of wavelengths, so the incoming traffic, given a specific requested wavelength, has multiple choices depending on such number of fibers.

The results presented in this dissertation show a significant reduction in the blocking probability if multiple fibers are used for the same set of optical network resources. For example in network with 1 fibers, 16 wavelengths per fiber and network diameter 10 hops, the blocking performance improvement could achieve 100% if only one fiber is added to the optical network when it operates at a fairly low load. This blocking performance could be maintained when the load per wavelength per hop is increased by increasing the number of fibers between nodes. Also, the gain study showed a tremendous throughput improvement provided by the multi-fiber environment compared to the single-fiber. This observation raises a core question for this research work. For a given optical network blocking performance, the minimum number of fibers needed to be lit between nodes in order to meet the blocking performance constraint under different operating loads is to be investigated in more depth within the context of this dissertation.

Furthermore, using multi-fiber helps reducing the cross-talk problem, which is a direct result of having a large number of channels in a single-fiber using DWDM fine spacing techniques. It helps to overcome the capacity exhaustion problem in the C-band window by offering the same set of wavelengths on multiple fibers. It reduces the expansion network cost through using an off the shelf transceivers (transmitters/receivers) instead of redesigning them to be able to transmit and receive the new added wavelengths. In addition to that, the optical switch needs to be redesigned when a new wavelength is added to the single-fiber case to be able to identify and switch the new wavelengths.

The use of wavelength conversion increases the improvement in network utilization due to the ability to minimize wavelength contention using such conversion resources. The results presented in this work confirm that the best performance of the optical network could be achieved by using a limited number of converters. It is noted that a 70% conversion degree, which means number of converters employed by the switch are equals to 70% of the total number of wavelengths, in the share-per-node wavelength convertible switch will offer a very comparable blocking performance compared to the dedicated wavelength convertible switch configuration, where every wavelength has its own wavelength converter. This indicates that the increase of conversion degree at the higher wavelength count does not result in a significantly higher performance. The results also showed that a smaller degree of conversion can yield the same performance as a dedicated wavelength conversion due to low network throughput where some wavelength converters remain unused.

Wavelength conversion may possibly improve the performance of the network by resolving the wavelength contention problem, but it incurs increased cost, hardware complexity, and space requirements implying potential trade-offs between the performance and the number of wavelength converters needed. This introduces the issue of cost effectiveness of the wavelength conversion options. So, it is more economically feasible to use wavelength convertible switches with conversion sharing and lower hardware complexity to reduce wavelength blocking probabilities and obtain better network performance.

In addition, the results showed that the performance gain increases as the number of converters increases and saturates as the number of converters becomes larger than some threshold and less than the total number of channels on the link, which implies that a limited number of converters, compared to the total number of wavelengths, under certain offered loads is sufficient to provide good performance. While the share-per-node wavelength convertible switch suffers from the increased hardware complexity, the share-per-link convertible switch provides the optical network with high flexibility configuration without introducing more hardware.

In this research the performance of this switch architecture in the multi-fiber environment is investigated and evaluated by developing analytical and simulation models. Using multi-fiber reduces the number of converters needed to achieve the best performance.

8.2 Slotted Optical Packet Switch

We developed a performance model for multi-fiber share-per-link OPS, which operates in a slotted mode. A symmetric OPS employing share-per-link conversion resources with N inputs, coming from different sources and destined to N outputs links, each consisting of F parallel fibers is investigated and analyzed. The results of this investigation showed a perfect match between the analytical and the simulation, which confirms the accuracy of the analysis developed in this research. Results further showed that at higher offered loads, the switch with a higher count of shared converters per link offers better packet loss performance due to its ability to accommodate more traffic. They also showed that using multiple fibers per link has far better effects on performance than the converter count at higher loads. At lower offered loads, the effects of shared converter counts in the single fiber case is more pronounced on utilization than in the multi-fiber link case due to the role the conversion plays in resolving contention.

Moreover, the results showed that output links with fewer fibers counts have higher conversion bank utilization, whereas switches with higher fiber count in their output links have a lower utilization rate for the conversion resources. From this, it can be deduced that a slotted switch equipped with dedicated conversion would have the least conversion resources utilization, indicating that the use of a switch with a lower converter count would offer better conversion resources utilization and comparable packet loss behavior. Thus, the number of shared converters can be significantly reduced when using multiple fibers due to the improvement and the higher utilization the multi-fiber link offers. These performance results confirm the main research point of this dissertation, which is that the OPS performance could be enhanced without any additional costly resources through utilizing the multi-fiber environment. The cost trade-offs are more evident at higher offered loads due to the need for better managing the fiber, wavelength, and shared converter resources. The model presented can be used to evaluate such trade-offs.

A cost tradeoff study was conducted to have a sense of the performance contribution, in terms of the number of wavelengths add when conversion is not introduced, of a single wavelength converter in the OPS switch system. A similar cost tradeoff scenario for the case of introducing multiple fiber counts was also considered. The results showed that the conversion resources, as well as the multiple fibers, worth less with the increase of a single wavelength offered load. This is due to the high competition on the limited switch resources at higher offered loads per wavelength. It is notable that the introduction of additional fiber to the OPS switch worth more, in terms of the number of wavelengths needed to achieve similar performance, than the introduction of a conversion resource. This confirms that the performance improvement introduced by additional fiber is much greater than the improvement made by adding a converter.

8.3 Optical Network Survivability

A hierarchical model to evaluate the system survivability performance was developed. An algorithm was presented to analyze the availability of different non-disjoint edge paths.

The end-to-end performance of circuit-switched all-optical networks in multi-fiber multi-hop environments was modeled and evaluated with and without wavelength conversion. These models were utilized to evaluate the performance degradation when a failure occurs. The performance degradation model and the availability analysis model were combined to construct a hierarchical network survivability evaluation model.

The study showed that the use of multiple fibers at the connectivity topology level of WDM networks provides additional network capacity and increases network survivability by reusing the same set of wavelengths in multiple fibers in case of link failure. The use of wavelength conversion increases survivability by resolving the wavelength contention problem. However, using wavelength conversion can increase cost, hardware complexity, and space requirements to the network, implying potential trade-offs between the performance and the number of wavelength converters needed. The survivability of the WDM network decreases as the steady state availability decreases.

This result proves another objective goal for this dissertation, that the accurate survivability measure of the optical network has to consider both the performance degradation during the time of failure and the resources availability at the time of failure. Moreover, the results showed that having multiple fibers between nodes on the same physical link enhances the optical performance; however, this optical realization will result in a massive traffic loss if a link failure occurs between a pair of nodes. Therefore, in this dissertation a new approach for a more survivable optical network was presented and evaluated. This new approach provides multiple fibers between the nodes, but these fibers are not on the same physical link by having the intermediate nodes behaving as passive connectors. Also, the results showed that increasing the number of hops decreases network survivability, as expected. Using multiple fibers in the optical network implementation will improve network survivability and will decrease the effect of the network size on survivability.

8.4 Future Research

Several issues related to the multi-fiber optical networks can be further investigated and studied as an extension of this work. Evaluating performance of the multi-fiber network for different network topologies is an extension of this research. Also, studying the impact of different routing and wavelength assignment algorithms on the multi-fiber optical network performance could be explored more fully. Proposing new wavelength assignment algorithms that can efficiently utilize the abundance of the resources availability provided by the multi-fiber environment is one of the most promising areas that may enhance overall network performance. New switch architectures such as the *have-clear switch architecture* that can be utilized in the multi-fiber environment need to be investigated and evaluated for the high cost reduction and simple control management needed. The performance of switch investigated in this work could be enhanced by utilizing an optical buffer on the output. The analytical and simulation models in this study could be extended to evaluate this new switch architecture.

Bibliography

[1] F. Al-Zahrani, A. Habiballa, and A. Jayasumana, "Path Blocking Performance in Multi-Fiber Wavelength Routing Networks with and without Wavelength Conversion," Proc. Thirteenth International Conference on Computer Communications and Networks (ICCCN03), Oct. 2003, pp. 580-583.

[2] F. Al-Zahrani, A. Habiballa, and A. Jayasumana, "Performance Merits of Multi-Fiber DWDM Networks Employing Different Shared Wavelength Conversion Resources Architectures," Proc. IEEE Region 5 Conference, April 2004, pp.59-67.

[3] S. Banerjee and C. Chen, "Design of Wavelength-Routed Optical Networks for Circuit Switched Traffic," Proc. IEEE GLOBECOM'96, Vol. 1, Nov. 1996,pp. 306 - 310.

[4] S. Banerjee, J. Yoo, and C. Chen, "Design of Wavelength-Routed Optical Networks for Packet Switched Traffic," IEEE/OSA J. of Lightwave Technology, Vol. 15, Issue 9, Sept. 1997, pp.1636-1646.

[5] J. Banks, J.S. Carson, B.L. Nelson, Discrete Event System Simulation, 3rd Ed., Prentice-Hall, New Jersey, 2001.

[6] S. Baroni, P. Bayvel, R. J. Gibbens, and S. K. Korotky, "Analysis and Design of Resilient Multifiber Wavelength-routed Optical Transport Networks," IEEE/OSA J. of Lightwave Technology, Vol. 17, No. 5, May 1999, pp. 743758.

[7] R. Barry and P. Humblet, "Models of Blocking Probability in All-Optical Networks with and without Wavelength Changers", Proc. IEEE INFOCOM'95, April 1995, pp. 402-412.

[8] R. Bhandari, "Survivable Networks, Algorithms for Diverse Routing," Kluwer Academic Publishers, 1999.

[9] A. Birman, "Computing Approximate Blocking Probabilities for a Class of All Optical Networks," Proc. IEEE INFOCOM'95, April 1995, pp. 651-658.

[10] X. Cao, V. Anand, Y. Xiong, and C. Qiao, "Performance Evaluation of Wavelength Band Switching in Multi-fiber All-optical Networks," Proc. of IEEE INFOCOM03, Vol. 3, Mar.-Apr. 2003, pp. 2251-2261.

[11] A. Carena et al.,"RingO: an experimental WDM optical packet network for metro applications," IEEE J. on Selected Areas in Communications,Vol. 22, Issue 8, Oct. 2004, pp. 1561-1571.

[12] K. Chan and T. P. Yum, "Analysis of Least Congested Path Routing in WDM Lightwave Networks," Proc. IEEE INFOCOM'94, April 1994, Vol. 2, pp. 962-969.

[13] S. Chalasani, V. Rajaravivarma, "Survivability in Optical Networks," Proc. of the 35th Southeastern Symposium on System Theory, Mar. 2003 pp. 6-10.

[14] D. Chen, S. Garg, and K. S. Trivedi, "Network Survivability Performance Evaluation: A Quantitative Approach with Applications in Wireless Ad-hoc Networks," 5th ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems, Sept. 2002, pp. 61-68.

[15] I. Chlamtac, A. Ganz and G. Karmi, "Lightpath Communications: an Approach to High Bandwidth Optical WAN's," IEEE Trans. on Commun., 1992, Vol. 40, Issue 7, July 1992, pp. 1171-1182.

[16] K. Coffman and A. Odlyzko, "Handbook of Massive Data Sets, Chapter Internet growth: Is there a Moores Law for data traffic?," Kluwer, 2001.

[17] S. L. Danielsen, C. Joergensen, B. Mikkelsen, and K. E. Stubbkyaer, "Optical Packets Switched Network Layer without Optical Buffers," IEEE Photon. Technol. Lett., Vol. 10, No. 6, June 1998, pp. 896-898.

[18] J. M. H. Elmirghani and H. T. Mouftah, "All-Optical Wavelength Conversion: Technologies and Applications in DWDM Networks," IEEE Commun. Mag., Vol. 38, Issue 3, Mar. 2000, pp. 86-92.

[19] V. Eramo and M. Listanti, "Packet Loss in a Bufferless Optical WDM Switch Employing Shared Tunable Wavelength Converters" IEEE/OSA J. of Lightwave Technology, Vol. 18, No. 12, Dec. 2000, pp. 18181833.

[20] V. Eramo et al., "A Comparison Study on the Wavelength Converters Number Needed in Synchronous and Asynchronous All-optical Switching Architectures," IEEE/OSA J. of Lightwave Technology, Vol. 21, Feb. 2003, pp. 340355.

[21] V. Eramo and M. Listanti, "Input Wavelength Conversion in Optical Packet Switches," IEEE Comm. Letters, Vol. 7, No. 6, June 2003, pp.281283.

[22] G. Fishman, Discrete-Event Simulation Modeling, Programming, and Analysis, Springer, New York, 2001.

[23] C. Fow-Sen et al., "An optical packet switch based on WDM technologies," IEEE/OSA J. of Lightwave Technology, Vol. 23,Issue 3, March 2005, pp. 994-1014.

[24] C. M. Gauger, "Trends in Optical Burst Switching," Proc. of SPIE ITCOM, Sept. 2003, pp. 2-12.

[25] S.S. Gokhale, S.K. Tripathi, "Effect of Unreliable Nodes on QoS Routing Network Protocols," Proc. of 7th International Conference on Network Protocols, Oct.-Nov. 1999, pp. 173-181.

[26] A. Habiballa, F. Al-Zahrani, A. Jayasumana, "Wavelength Conversion Resources Allocation Algorithms for Share-per-link Wavelength Convertible Switch," Proc. IEEE Region 5 Conference, April 2004, pp. 131-139.

[27] D. Hunter, M. Chia, and I. Andonovic, "Buffering in Optical Packet Switches," IEEE/OSA J. of Lightwave Technology, Vol. 16, No. 12, Dec. 1998, pp. 2081-2094.

[28] J. P. Jue and G. Xiao, "Analysis of Blocking Probability for Connection Management Schemes in Optical Networks", Proc. IEEE GLOBECOM01, Nov. 2001, Vol. 3, pp. 1546-1550.

[29] M. Keshtgary, F. A. Al-Zahrani, A.H. Jahangir, and A. P. Jayasumana, "Survivability of Multi-Fiber WDM Networks with and without Wavelength Conversion," Proc. 2nd IASTED Conference on Communication and Computer Networks (CCN 2004), Nov. 2004, pp. 36-41.

[30] M. Keshtgary, F. A. Al-Zahrani, A.H. Jahangir, and A. P. Jayasumana, "Network Survivability Performance Evaluation with Applications in WDM Networks with Wavelength Conversion," Proc. 29th IEEE Conference on Local Computer Networks, Nov. 2004, pp. 344-351.

[31] M. Keshtgary, F. A. Al-Zahrani, A. P. Jayasumana, and A. H. Jahangir, "Survivability Performance Evaluation of WDM Networks with Wavelength Converters," To Appear in Photonic Network Communications.

[32] M. Kovacevic and A. Acampora, "Benefits of Wavelength Translation in All-optical Clear-channel Networks," IEEE J. on Selected Areas in Commun., Vol. 14, No. 5, 1996, pp. 868-880.

[33] C. Lee, "Analysis of Switching Networks," Bell System Technical Journal, Vol. 34, Nov. 1955, pp. 1287-1315.

[34] K. C. Lee and V. O. K. Li, "A Wavelength-Convertible Optical Network," IEEE/OSA J. of Lightwave Technology, Vol. 11, no. 5, May 1993, pp. 962-970.

[35] L. Li and A. Somani, "A New Analytical Model for Multifiber WDM Networks," Proc. GLOBECOM'99, Vol. 1B, Dec. 1999, pp.1007-1011.

[36] L. Li and Arun Somani, "Fiber Requirement in Multifiber WDM Networks with Alternate-Path Routing," Proc. International Conference on Computer Communications and Networks (ICCCN'99), Oct. 1999 pp. 338-343.

[37] L. Li and A. K. Somani, "Blocking Performance Analysis of Fixed-Paths Least-Congestion Routing in Multifiber WDM Networks," International J. of Commun. Systems, Mar. 2002, Vol. 15, Issue 2-3, pp. 143-159.

[38] K. Lu, G. Xiao, I. Chlamtac, "Analysis of Blocking Probability for Distributed Lightpath Establishment in WDM Optical Networks," IEEE/ACM Trans. on Networking, Vol. 13, Issue 1, Feb. 2005 pp. 187-197.

[39] Y. Luo, N. Ansari, "Restoration with Wavelength Conversion in WDM Networks," Electronics Letters, Vol. 38, Issue 16, Aug. 2002, pp. 900-901.

[40] Y. Luo and N. Ansari,"A Computational Model for Estimating Blocking Probabilities of Multifiber WDM Optical Networks," IEEE Commun. Lett.,Vol. 8, No. 1, Jan. 2004, pp. 6062.

[41] Y. Mei, and C. Qiao, "Efficient Distributed Control Protocols for WDM All-optical Networks", Proc. International Conference on Computer Communications and Networks (ICCCN'97), Sept. 1997, pp. 150-153.

[42] D. Mitra and J. B. Seery, "Comparative Evaluations of Randomized and Dynamic Routing Strategies for Circuit-switched Networks," IEEE Trans. Commun., Vol. 39, Jan. 1991, pp. 102-116.

[43] S. D. Moitra, , E. Oki, and N. Yamanaka, "Some New Survivability Measures for Network Analysis and Design", IEICE Trans. on Communication, Vol. E80-B, No. 4, April 1997, pp. 625-631.

[44] H. Obara, H. Masuda, K. Suzuki, and K. Aida, "Multifiber Wavelength Division Multiplexed Ring Network Architecture for Tera-bit/s Throughput," Proc. IEEE Int. Conf. Communications, Vol. 2, June 1998, pp. 921925.

[45] A. Odlyzko, "Data Networks are Mostly Empty and for Good Reason," IT Professional, Vol. 1, No. 2, Mar. 1999,pp.6769.

[46] K. Papagiannaki, N. Taft, Z. L. Zhang, and C. Diot, "Long-term Forecasting of Internet Backbone Traffic: Observations and Initial Models," Proc. IEEE INFOCOM 2003, Vol. 2, Mar. 2003, pp. 1178-1188.

[47] B. Ramamurthy and B. Mukherjee, "Wavelength Conversion in WDM Networking," IEEE J. on Selected Areas in Commun. Vol. 16, Issue 7, Sept. 1998, pp. 1061-1073.

[48] S. Ramamurthy, L. Sahasrabuddhe, and B. Mukherjee, "Survivable WDM Mesh Networks," IEEE/OSA J. of Lightwave Technology, Vol. 21, Apr. 2003, pp. 870883.

[49] S. Ramamurthy, "Optical Design of WDM Network Architectures," Ph.D. Dissertation, University of California, Davis, 1998.

[50] S. Ramamurthy and B. Mukhrjee, "Fixed-Alternate Routing and Wavelength Conversion in Wavelength-Routed Optical Networks," Proc., IEEE GLOBECOM'98, Vol.4, Nov. 1998, pp. 2295-2302.

[51] R. Ramaswami and G. Sasaki, "Multi-wavelength Optical Networks with Limited Wavelength Conversion," IEEE/ACM Trans. on Networking, Vol. 6, Dec.1998, pp. 744-754.

[52] R. Ramaswami and K.N. Sivarajan, "Routing and Wavelength Assignment in All-optical Networks," IEEE/ACM Trans. on Networking, Vol. 3, No. 5, Oct. 1995, pp. 489-500.

[53] M. Renaud, F. Masetti, C. Guillemot, and B. Bostica, "Network and System Concepts for Optical Packet Switching," IEEE Commun. Mag., Vol. 35, Apr. 1997, pp. 96102.

[54] RHK, "United States Internet traffic experiences annual growth of 100%, but just 17% revenue growth," Press release #157, RHK, Telecommunication Industry Analysis, May 2002.

[55] D. Saha, "A comparative Study of Distributed Protocols for Wavelength Reservation in WDM Optical Networks", Optical Network Mag., Vol. 3, No. 1, Jan. 2002, pp. 45-52.

[56] J. Shi, and J. P. Foneska, "Traffic-Based Survivability Analysis of Telecommunications Networks", Proc. IEEE GLOBECOM'95, Vol. 2 , Nov. 1995, pp. 936-940.

[57] S. Subramanium , M. Azizoglu, and A. Somani, "Connectivity and Sparse Wavelength Conversion in Wavelength-Routing Networks," Proc. IEEE INFOCOM'96, Vol. 1, Mar. 1996, pp. 148-155.

[58] T. Tripathi and K. N. Sivarajan, "Computing Approximate Blocking Probabilities in Wavelength Routed All-optical Networks with Limited Range Wavelength Conversion," IEEE J. Select. Areas Commun.,Vol. 18, Oct. 2000, pp. 2123-2129.

[59] K.S. Trivedi, "Probability and Statistics with Reliability, Queuing and Computer Science Applications," Willey, 2002.

[60] X. Wang, H. Morikawa, and T.Aoyama, "Distributed Wavelength Assignment Algorithm for Optical Bursts in WDM Networks," Proc. OECC2001, July 2001, pp. 191-193.

[61] G. Xiao and Y. W. Leung, "Algorithms for Allocating Wavelength Converters in All-optical Networks," IEEE/ACM Trans. on Networking, Vol. 7, Issue 4, Aug. 1999, pp. 545-557.

[62] Y. Xin and G. N. Rouskas, "A Study of Path Protection in Large-scale Optical Networks," Photonic Network Communications, Vol. 7, No. 3, May 2004, pp. 267-278.

[63] S. Yao, S. J. B. Yoo, B. Mukherjee, and S. Dixit. "All-Optical Packet Switched Networks: A Study of Contention-resolution Schemes in an Irregular Mesh Network with Variable-Sized Packets," OPTICOMM 2000, Oct. 2000, pp.235-246.

[64] S. Yao, S. J. B. Yoo, and B. Mukherjee, "A Comparison Study between Slotted and Unslotted All-optical Packet-switched Network with Priority-based Routing," Proc. OFC01, Vol. 2, Mar. 2001, pp. TuK21Tuk23.

[65] X. Yuan, R. Melhem, R. Gupta, Y. Mei, and C. Qiao, "Distributed Control Protocols for Wavelength Reservation and Their Performance Evaluation", Photonic Network Communications,Vol. 1, No. 3, 1999, pp. 207-218.

[66] L. Yuanqiu, N. Ansari, "Split Restoration with Wavelength Conversion in WDM Networks,"' Proc. of IEEE International Conference on Communications, Vol. 2, May 2003, pp. 1423-1427.

[67] H. Zang, J. P. Jue, and B. Mukherjee, "A Review of Routing and Wavelength Assignment Approaches for Wavelength-routed Optical WDM Networks," Optical Network Mag.,Vol. 1, No. 1, Jan. 2000, pp. 47-63.

[68] P. Zhou and O. Yang,"How Practical is Optical Packet Switching in Core Networks?," Proc. IEEE GLOBECOM 2003, Vol. 5, Dec. 2003, pp. 27092713.

[69] D. Zhou and S. Subramaniam, "Survivability in Optical Networks," IEEE Network, Vol. 14, Issue 6, Nov.-Dec. 2000, pp. 16-23.

[70] Y. Zhu, G. N. Rouskas, and H. G. Perros, "A Comparison of Allocation Policies in Wavelength Routing Networks," Photonic Network Communications, Vol. 2, No. 3, Aug. 2000, pp. 265-293.

APPENDIX

Appendix A

ANALYTICAL MODEL SOURCE CODE: BLOCKING PROBABILITY OF WAVELENGTH-ROUTED NETWORK

The source code of the analytical model, which is written in Matlab, to evaluate the blocking probability of the wavelength-routed network is shown below. This code is used for the following network configurations:

- Single-fiber wavelength-routed network
- Wavelength-routed network with dedicated wavelength conversion option
- Share-per-node wavelength-routed network

The blocking probability models for these wavelength-routed network configurations explained in detail in Chapter 4.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% File name: Blocking-Probability.m
%%
%% Purpose: To calculate the blocking probability of the
%% wavelength-routed network. Three differenet
%% wavelength-routed network are considered:
%% 1. Single-fiber wavelength-routed network
%% 2. Wavelength-routed network with dedicated conversion
%% 3. Multi-fiber share-per-node wavelength-routed network
%%
%% Inputs:
%%    DC : Degree of conversion
%%   rho: Load per wavelengt
%%   G  : Source-destination per load
%%   H  : Number of hops
```

```
%%  F   : Number of fibers
%%  W   : Total number of wavelengths per hop
%%  c   : Number of converters
%%
%% Outputs:
%%  ps : Single-fiber environment blocking probability
%%  Us : Single-fiber environment throughput
%%  p0s: Probability of zero converters is busy in the
%%        single-fiber case
%%  pcs: Probability of all converters are busy in the
%%        single-fiber case
%%  pn : Multi-fiber share-per-node environment blocking
%%        probability
%%  Un : Multi-fiber share-per-node environment throughput
%%  p0n: Probability of zero converters is busy in the
%%        multi-fiber share-per-node case
%%  pcn: Probability of all c converters are busy in the
%%        multi-fiber share-per-node case
%%  pf : Dedicated conversion environment blocking
%%        probability
%%  Uf : Dedicated environment throughput
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all;
clc;
ranage=11;
DC=0.7
rho=0:0.1:1;
G=0.07;
H=20;
F=2;
W=16;
c=ceil(DC*W);
pl=zeros(ranage,1);
pf=zeros(ranage,1);
ps=zeros(ranage,1);
pn=zeros(ranage,1);
p0s=zeros(ranage,1);
pcs=zeros(ranage,1);
p0n=zeros(ranage,1);
p0l=zeros(ranage,1)
pcl=zeros(ranage,1);
pcn=zeros(ranage,1);
Us=zeros(ranage,1);
Uf=zeros(ranage,1);
```

```
Un=zeros(ranage,1);
Ul=zeros(ranage,1)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Calculating probability of zero converters is busy%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
sump=0;
for j=1:ranage
    for i=1:c
        sump=sump+(1/factorial(i))*(factorial(W)/
        factorial(W-i))*rho(j)^(2*i);
    end
    p0s(j)=1/(1+sump);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Calculating probability of c converters are busy%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:ranage
    pcs(i)=p0s(i)*(1/factorial(c))*(factorial(W)/
    factorial(W-c))*rho(i)^(2*c);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Calculating blocking probability of  single-fiber case   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:ranage
    ps(i)=(1-((1-rho(i))+rho(i)*(1-pcs(i))*(1-rho(i)^(W-1)))^H)^W;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Calculatin throughput of single-fiber case %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:ranage
    Us(i)=(1-ps(i))*G;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Calculating blocking probability of dedicated conversion case%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:ranage
    pf(i)=(1-(1-rho(i)^(W))^H)^W;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Calculating throughput of dedicated conversion case%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:ranage
    Uf(i)=(1-pf(i))*G;
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%Calculating probability of zero converters is busy%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
sump=0;
for j=1:ranage
    for i=1:c
        sump=sump+(1/factorial(i))*(factorial(W)/
        factorial(W-i))*rho(j)^(i*F+i);
    end
    p0n(j)=1/(1+sump);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Calculating probability of c converters is busy %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:ranage
    pcn(i)=p0n(i)*(1/factorial(c))*(factorial(W)/
    factorial(W-c))*rho(i)^(c*F+c);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Calculating blocking probability of multi-fiber
%% share-per node case
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:ranage
    pn(i)=(1-((1-rho(i)^F)+(rho(i)^F)*(1-pcn(i))*
    (1-rho(i)^(W-F)))^H)^W;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Calculating throughput of multi-fiber share-per node case%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:ranage
    Un(i)=(1-pn(i))*G;
end
```

Appendix B

ANALYTICAL MODEL SOURCE CODE: CONVERSION RESOURCES ALLOCATION ALGORITHMS (CRAA)

The source code of the analytical model, which is written in Matlab, to evaluate the blocking probability of different wavelength conversion resources allocation algorithms for the share-per-link wavelength-routed network is shown below. The blocking probability model for NSA explained in detail in Chapter 4. The other algorithms were presenetd in [26].

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% File name: CRAA.m
%% Purpose: To identify the blocking probabilty of
%% the wavelength-routed network employing share-per-
%% link conversion schme.
%% Inputs:
%%    DC : Degree of conversion
%%  rho: Load per wavelengt
%%  G  : Source-destination pair load
%%  H  : Number of hops
%%  F  : Number of fibers
%%  W  : Total number of wavelengths per hop
%%  c  : Number of converters
%% Outputs:
%%  p0 : Probability of zero converters is busy
%%  pc : Probability of all c converters are busy in
%%         the single-fiber case
%%  pb : Blocking probability
%%  U  : Throughput
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all;
clc;
pc=0:0.001:1;
ranage=101;
pc2=zeros(16,1);
rho=.8;
G=0.07;
```

```
F=10;
w=1:1:32;
DC=0.2;
c=ceil(DC*w);
H=10;
pb=zeros(11,1);
n=input('n = ');
s=0;
s1=0;
U=zeros(32,1);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% The following section is the implementation of the
%% Inclusive Search Algorithm (ISA). Numerical iteration
%% method is implemented to calculate pc and p0. First p0
%% is calculated by using initial values for pc, which
%% is set above, then new values for the pc is found.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for j=1:F-1
    s1=s1+pc(1)^ j;
end
s1=s1+pc(1)^0;
for k=1:c(1)
        x=(1/factorial(k))*(factorial(w(1))/
        factorial(w(1)-k))*rho^(k*F+k)*s1^k;
        s=s+x;
end
p0=1/(1+s);
pck=(1/factorial(c(1)))*(factorial(w(1))/
factorial(w(1)-c(1)))*(rho^(c(1)*F+c(1)))*p0*s1^c(1);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% The difference between the pervious value of pc and
%% the new calculated value is considered as the tolerance
%% for this numerical iteration method. This tolerance is
%% calculated in the following code line.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
y1=sign(pc(1)-pck);
for i=1:32
for l=2:1001
    s=0;
    s1=0;
for j=1:F-1
     s1=s1+pc(l)^ j;
end
s1=s1+pc(l)^ 0;
```

```
for k=1:c(i)
    x=(1/factorial(k))*(factorial(w(i))/
    factorial(w(i)-k))*rho^(k*F+k)*s1^k;
    s=s+x;
end
p0=1/(1+s);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% The new values of pc is calcualted%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
pck=(1/factorial(c(i)))*(factorial(w(i))/
factorial(w(i)-c(i)))*(rho^(c(i)*F+c(i)))*p0*s1^c(i);
y=sign(pc(l)-pck);
if ((y~=y1)|(y==0))
    pc2(i)=pc(l);
    break;
end
y1=y;
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Blocking probability of ISA calculation.%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:32
    pb(i)=1-((1-rho^ F)+(rho^F)*(1-(pc2(i))^ F)*
    (1-rho^((w(i)-1))))^ H;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Throughput is calculation%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:32
    U(i)=(1-pb(i))*G;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% The following section is the implementation
%% of the Narrow Search Algorithm (NSA).
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
sump=0;
%%%%%%%%%%%%%%%%%%
%% Calculating p0%
%%%%%%%%%%%%%%%%%%
for j=1:32
 for i=1:c(j)
   sump=sump+(1/factorial(i))*(factorial(w(j))/
   factorial(w(j)-i))*rho^(i*F+i);
```

```
 end
 p0(j)=1/(1+sump);
end
for i=1:32
    pc(i)=p01(i)*(1/factorial(c(i)))*(factorial(w(i))/
    factorial(w(i)-c(i)))*rho^(c(i)*F+c(i));
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Calculating the blocking probability%
%% for NSA.                            %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:32
    pb(i)=1-((1-rho^F)+(rho^F)*(1-pcl(i))*
    (1-rho^((w(i)-1))))^H;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Calculating the throughput of NSA.%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:32
   U(i)=(1-pb(i))*G;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% The following section is an implementation of            %
%% Inclusive Search Algorithm (OSA). The Numerical          %
%% iteration method that is useed for ISA will be           %
%% utilized in this algorithm implementation.               %
%% The only difference between the implementation of        %
%% the numerical iteration method is taht the of wavelength %
%% conversion bank that will be searched in OSA will be     %
%% specified. First p0 is calculated by using initial values%
%% for pc, which is set above, then new values for the      %
%% pc is found.                                             %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
s=0;
s1=0;
clear pc2;
clear pck;
clear p0;
for j=1:n
    s1=s1+pc(1)^j;
end
s1=s1+pc(1)^0;
for k=1:c(1)
   x=(1/factorial(k))*(factorial(w(1))/factorial(w(1)-k))
```

```
    *rho^(k*F+k)*s1^k;
   s=s+x;
end
p0=1/(1+s);
pck=(1/factorial(c(1)))*(factorial(w(1))/factorial(w(1)-c(1)))
*(rho^(c(1)*F+c(1)))*p0*s1^c(1);
y1=0;
y1=sign(pc(1)-pck);
pc2=zeros(32,1);
for i=1:32
y=0;
for l=2:1001
   s=0;
   s1=0;
for j=1:n
   s1=s1+pc(l)^j;
  end
  s1=s1+pc(l)^0;
  for k=1:c(i)
    x=(1/factorial(k))*(factorial(w(i))/factorial(w(i)-k))
     *(rho^(k*F+k))*(s1^k);
    s=s+x;
  end
  p0=1/(1+s);
  pck=(1/factorial(c(i)))*(factorial(w(i))/factorial(w(i)-c(i)))
   *(rho^(c(i)*F+c(i)))*p0*s1^c(i);
  y=sign(pc(l)-pck);
  if ((y~=y1)|(y==0))
     pc2(i)=pc(l);
     break;
  else
     y1=y;
  end
 end
end
for i=1:32
   pb(i)=1-((1-rho^F)+(rho^F)*(1-(pc2(i)^(n)))*
   (1-rho^((w(i)-1))))^H;
end
for i=1:32
  U(i)=(1-pb(i))*G;
end
```

Appendix C

SIMULATORS HEADER FILES SOURCE CODE

The sources codes of header files for the unslotted simulators are shown below. These header files are discussed in Chapter 7. These headers files are utilized by the simulators when a new network topology is constructed or a new node configuration is set. Also the Packet.h header file contains the packet class structure that utilized whenever a new event is generated. The list .h is used by the simulators to insert or delete an event from the event list.

```
/*-----------------------------------------------------------
                           Headers file
                                By
                        Fahad A. AL-zahrani
 The following header files are common header files for
 unslotted simulators that build in this research work.
 Files name:
        Node1.h
        Packet.h
        List.h
 Inputs:

 Outputs:

-------------------------------------------------------------*/

#include <iostream.h>
#include <stdlib.h>
#include <fstream.h>
#include <math.h>
#include <time.h>
#include <stdio.h>
#include "packet1d.h"
#include "node1d.h"
/*-------------------------------------------------
              list.h file
This header file has one class that is class list.
```

```
this class has three public functions.These functions
are:
Insert _To_List: this function is used by the simulator
                 to insert a new or updated event to the
                 event list.
Detele_From_List: This function is called when an event
                  need to be deleted from the event list.
Get_Tail: This function is used to get a pointer to the
          event list tail.
---------------------------------------------------*/
class list{
public:
   packet *Insert_To_List(packet*, packet*);
   packet *Delete_From_List(packet *);
   packet *Get_Tail(packet *);
};
packet *list::Insert_To_List (packet *h,packet *c)
{
  packet *temp;
  temp=h;
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// Check if the event list is empty, which means this
// inserted event is the fisrt event. if so set the header
// of the list to point to this event return the header.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  if (temp==NULL)
  {
   temp=c;
   h=c;
   h->Next_Packet=NULL;
   h->Prev_Packet=NULL;
   return h;
  }
  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  // Check if the current event has to be processed before
  // the event that pointed to by the header pointer. if so
  // inseret this event before the event in the header and
  // update the header pointer to point to this event and
  //  return the new header pointer.
  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  if ((c->Departure_Time<temp->Departure_Time)&&(temp!=NULL))
  {
   temp->Prev_Packet=c;
   c->Next_Packet=temp;
```

```
   h=c;
   return h;
  }
  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  // The following section search the event list one by one for an
  // event that has departure time lager than the current event if
  // an event is found in the event list. Insert the current event
  // before this event and return the header, otherwise insert the
  // current event at the end of the list
  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  temp=h;
  while (temp->Departure_Time<=c->Departure_Time)
  {
    if (temp->Next_Packet==NULL)
    {
       temp->Next_Packet=c;
       c->Prev_Packet=temp;
       c->Next_Packet=NULL;
       packet *test=NULL;
       test=Get_Tail(h);
       return h;
    }else
       temp=temp->Next_Packet;
    }
    c->Next_Packet=temp;
    c->Prev_Packet=temp->Prev_Packet;
    temp=temp->Prev_Packet;
    temp->Next_Packet=c;
    temp=c->Next_Packet;
    temp->Prev_Packet=c;
    return h;
}
//----------------------------------------------------
packet *list::Delete_From_List(packet *h)
{
   packet *temp=h;
   h=temp->Next_Packet;
   h->Prev_Packet=NULL;
   temp->Next_Packet=NULL;
   delete temp;
   packet *test=NULL;
   test=Get_Tail(h);
   return h;
}
```

```
//----------------------------------------------
packet *list::Get_Tail(packet *h)
{
   packet *temp=h;
   while(temp->Next_Packet!=NULL)
      temp= temp->Next_Packet;
   return temp;
}
%
```

```
/*---------------------------------------------
               node.h file
This header file has two classes.
they are class converter and class node1.
--------------------------------------------*/
/*---------------------------------------------
                      Converter class
the class converter is used to define the conversion
option inside the node.This class has the following
publc variables:
C_No: the converter number
C_Status: represnt the status if the converter
         (Busy or Free)
In_S_Range: Indicate the the starting range of
            the converter at the input side in
            case of one of the Range-to-any conversion
            options is adopted.
In_E_Range: Indicate the input ending range of
            the converter
Out_S_Range:Indicate the the starting range of
            the converter at the output side in
            case of one of any-to-range conversion
            options is adopted.
Out_E_Range:Indicate the output ending range of
            the converter
-------------------------------------------------*/
class converter
{
public:
  int C_No;
  int C_Status;
  int In_S_Range;
  int In_E_Range;
  int Out_S_Range;
  int Out_E_Range;
  __int64 Attempt[256];
  __int64 Pbusy[256];
  converter();
};
converter::converter()
{
  C_No=-1;
  C_Status=-1;
  In_S_Range=-1;
```

```
  In_E_Range=-1;
  Out_S_Range=-1;
  Out_E_Range=-1;
  for(int i=0;i<256;i++)
  {
   Attempt[i]=0;
   Pbusy[i]=0;
  }
}
/*----------------------------------------------
                      Node class
the class node1 is used to define the node
configuartion parameters such as number of
fibers, number of wavelengths, Output ports
status, and different conversion options.
This class has the following publc variables:
Out_F;
Out_W;
C_T;
C;
Out_Port[256][256];
Arrived_Packet;
Pro_Packet;
blocked;
Th[256][256];
Also it has the following public functions
node1();
Out_Port_Initialization(int, int);
---------------------------------------------------*/
class node1 {
   public:
      int Out_F;
      int Out_W;
      int C_T;
      converter *C;
      int  Out_Port[256][256];
      long Arrived_Packet;
      long Pro_Packet;
      long blocked;
      __int64 Th[256][256];
      node1();
      void Out_Port_Initialization(int, int);
};
```

```
node1::node1()
{
for (int i=0;i<256;i++)
{
  for (int j=0;j<256;j++)
    {
      Out_Port[j][i]=0;
  }
}
blocked=0;
Arrived_Packet=0;
Pro_Packet=0;
}
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++
// In this function is called once the node is created
//inside the main simulator. All the node paramters are
//initialized such as the output prots status is set to
//free, and the conveters is numbers and their input
//and output ranges are set.
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
void node1::Out_Port_Initialization(int C,int d)
{
//+++++++++++++++++++++++++++++++++++++++++++++++
// This part initialize the output port status
//+++++++++++++++++++++++++++++++++++++++++++++++
  for (int i=0;i<this->Out_F;i++)
  {
    for (int j=0;j<this->Out_W;j++)
    {
      Out_Port[j][i]=1;
      Th[j][i]=0;
    }
  }
//++++++++++++++++++++++++++++++++++++++++++++
// Conveter object is created and initialized
//++++++++++++++++++++++++++++++++++++++++++++
this->C=new converter[C+1];
  for (i=0;i<C+1;i++)
  {
    this->C[i].C_No=i;
    this->C[i].C_Status=1;
    if (this->C_T==0)
    {
      this->C[i].In_S_Range=0;
```

```
      this->C[i].In_E_Range=this->Out_W-1;
      this->C[i].Out_S_Range=0;
      this->C[i].Out_E_Range=this->Out_W-1;
    }else if (this->C_T==1)
    {
      this->C[i].In_S_Range=0;
      this->C[i].In_E_Range=this->Out_W-1;
      int x=(i+1);
      if (x>(C-1))
      x=0;
      this->C[i].Out_S_Range=x*d;
      this->C[i].Out_E_Range=x*d+d-1;
    }else if(this->C_T==2)
    {
     this->C[i].In_S_Range=i*d;
     this->C[i].In_E_Range=i*d+d-1;
     this->C[i].Out_S_Range=0;
     this->C[i].Out_E_Range=this->Out_W-1;
   }else if (this->C_T==3)
   {
     this->C[i].In_S_Range=i*d;
     this->C[i].In_E_Range=i*d+d-1;
     int x=(i+1);
     if (x>(C-1))
      x=0;
     this->C[i].Out_S_Range=i*d;
     this->C[i].Out_E_Range=i*d+d-1;
   }
  }
}
```

```
/*---------------------------------------------------------
               packet.h file
This header file has one class. this class is packet
class. This class is called by the simulator
whenver a new packet evnet is created. this
class has the following fileds:
Packet_Type: is used to indicate the event typ
             either data event or resources event
Packet_ID: is used to keep track of the number of
           event that are generated.
Source: is used to identify the source of the packet
       in case of the network simulation
Lambda: is used to identify the wavelength utilized by
        the packet
F: is used to indicate which fiber is used by this packet
C_Flag: is used to identify if this packet get converted
        to a new wavelength
Initial_Time: this field is used to hold the initial time
             this packet get injected to the network
Departure_Time: this field is used to indicate the packet
                time to be precess
Arrival_Time: this field is used to hold the arrival time of
              the current packet to the next node
Packet_Length: is used to hold the packet length;
Prev_Packet: is used to indicate the prvious event in
             the even list
Next_Packet: is used to indicate the next event in the
             even list
packet (): this function is called when the packet is
           generated to intialize all the above field
-------------------------------------------------------------*/
class packet
{
     public:
     char Packet_Type;
     __int64 Packet_ID;
     int Source;
     int Lambda;
     int F;
     int C_Flag;
       __int64 Initial_Time;
       __int64 Departure_Time;
       __int64 Arrival_Time;
       signed __int64 Packet_Length;
```

```
       packet *Prev_Packet;
       packet *Next_Packet;
       packet ();
       ~packet(void);
};
packet::~packet()
{
}
packet::packet()
{
  Packet_Type='\0';
  Source=-1;
  Lambda=-1;
  F=-1;
  C_Flag=-1;
  Initial_Time=0;
  Departure_Time=0;
  Arrival_Time=0;
  Packet_Length=0;
  Prev_Packet=NULL;
  Next_Packet=NULL;
}
```

Appendix D

SIMULATOR SOURCE CODE: WAVELENGTH-ROUTED NETWORK

The simulator sources code for wavelength-routed network is shown below. This simulator is explained in detail in Chapter 7. The simulator is used to evaluate end-to-end blocking probability of the wavelength-routed network. The fixed routing algorithm is considered in this work. The random wavelength assignment was implemented in this simulator.

```
/*----------------------------------------------------------------
--             Wavelnegth-routed network Simulator
--                                By
--                      Fahad A. AL-zahrani
--  This simulator was build as a part of my Ph.D dissertation.
--  This simulator is used to evaluate end-to-end blocking
--  probability of the wavelength-routed network. The fixed
--  routing algorithm is considered in this work. The random
--  wavelength assignment was implemented in this simulator.
--  File Name: Sim_WRN.cpp
--  Inputs:
--       Topology.dat
--  Outputs:
--        Pb: Blcoking probability
-----------------------------------------------------------------*/
/*----------------------------------------------------------------
--                      Global variables
-- In this Part of the simulator all global variables that are
-- used in the different functions of the simulator are defined
-- These variables are:
-- Simulator_end_Time: Holds the simulator end time
-- Simulator_start_Time: Holds the simulator starting time
-- alpha:is used to define a single source-destination
-- traffic ratio
-- Reservation_Method: is used to identify the resources
-- resrvation protocol.
-- Assignment_Algorithm: is used to define which wavelength
```

```
-- assignment will be used in this simulation run
-- Stop_Flag: is used for a simulator debuging.
-- Stteing this to 1 during the simulation will
-- end the simulator
-- Cflag: Flag indicator to identify all packets that get
-- converted
-- Blocked: Holds the number of the blocked packets
-- Processed: Holds the number of the precessed packets
-- Seed1: The Initial seed that used to generate the packet
-- arrival time
-- Seed2: The Initial seed that used to generate the
-- packet length
---------------------------------------------------------------*/
#include "list.h"
unsigned __int64 Simulator_start_Time=0;
unsigned __int64 Simulator_end_Time=0;
double alpha=0.0;
int Reservation_Method;
int Assignment_Algorithm;
int Stop_Flag=1;
int Cflag=0;
int Destination_Flag;
long Blocked=0;
long Processed=0;
__int64 seed=1;
__int64 seed1=11;
__int64 spacket=0;
__int64 packetL=0;
/*-------------------------------------------------------------
--                      Random generator module
-- There are two functions in this module. The First one is
-- used to generate the seed that will be use to generate the
-- next random variable.The second function is the actual
-- implementation of the LCG random generator.
----------------------------------------------------------------*/
/*--------------------------------------------------------------
--                          GenerateSeed function
-- Function return type: 64 bit integer
-- Calld in variables funtion: Seed- 64 bit integer
-- Returned funtion variable : a - 64 bit integer
----------------------------------------------------------------*/
__int64 GenerateSeed(__int64 Seed)
{
   __int64 a;
```

```
   a =(__int64)(abs( (16807 * Seed) % (2147483647)));
   return a;
}
/*-------------------------------------------------------------------
--                          Generateu function
-- Function return type: bouble
-- Calld in variables funtion: Seed- 64 bit integer
-- Returned funtion variable : u - double
---------------------------------------------------------------------*/
double GenerateU(__int64 Seed)
{
   double u=0.0;
   u = ((double)(Seed) / (double)2147483647);
   return u;
}
/*--------------------------------------------------------------
--                          Destination_Generation function
-- Function return type: int
-- Calld in variables funtion: s:integer, No_of_Node:integer,
-- n:node1 class objet
-- Returned funtion variable : d: integer
-- This function is used to generate the lightpathrequest
-- destination.
----------------------------------------------------------------*/
int Destination_Generation(int s,int No_of_Node,node1 n[])
{
   int d=0;
   int c=-1;
   int count=-1;
   double u=0.0;
   if ((n[s].Node_Type=='s')&&(n[s].Node_No==0))
   {
     for (int i=0;i<No_of_Node;i++)
     {
       if(n[i].Node_Type=='d')
       {
         d=n[i].Node_No;
         return d;
       }
     }
    }else if(n[s].Node_Type!='d')
    {
      if (Destination_Flag==1)
      {
```

```
        int *dm=new int[No_of_Node];
        for (int i=0;i<No_of_Node;i++)
         dm[i]=-1;
        c=n[s].Next_Node;
        if(n[s].Node_Type=='s')
  c=n[c].Next_Node;
        while (c!=-1)
        {
          count++;
          dm[count]=c;
          c=n[c].Next_Node;
        }
        u=GenerateU(n[s].Node_Seed[2]);
        n[s].Node_Seed[2]=GenerateSeed(n[s].Node_Seed[2]);
        d=dm[(int)(count*u)];
      }else
      {
        d=n[s].Next_Node;
      }
   }
   return d;
 }
/*----------------------------------------------------------------
--                      Packet generation module
-- There are two functions in this module. The First function is
-- used to generate the release resources packets. The second
-- function is the used to generate the data packet events.
-------------------------------------------------------------------*/
/*-------------------------------------------------------------
--                           C_packet function
-- Function return type: packet class type
-- Called in variables function: c:packet class type
-- Returned function variable : Current_Packet packet class type
-- The called function passes the c packet pointer to
-- this function. The C_packet function extract the
-- necessary information from the packet such as Sourcs,
-- wavelength, Fiber, conversion flag, Packet ID,
-- Departure_Time, then it generate a new packet event
-- and set the departure time for this new packet as
-- departure time of the original packet plus the original
-- packet length
-------------------------------------------------------------------*/
packet *C_packet(packet *c)
{
```

```
   int Intialize_Time=0;  //necessary time to intailize laer
   packet *Current_Packet=NULL;
   Current_Packet=new packet();
   Current_Packet->Source=c->Source;
   Current_Packet->Destination=c->Destination;
   Current_Packet->Current_Node=c->Current_Node;
   Current_Packet->Departure_Time=c->Departure_Time+c->Packet_Length;
   Current_Packet->Lambda=c->Lambda;
   Current_Packet->F=c->F;
   Current_Packet->C_Flag=c->C_Flag;
   Current_Packet->Packet_Type='R';
   Current_Packet->Packet_Tag=2;
   Current_Packet->Packet_ID=c->Packet_ID;
   Current_Packet->Packet_Status=0;
   Current_Packet->Prev_Packet=NULL;
   Current_Packet->Next_Packet=NULL;
   return Current_Packet;
}
/*---------------------------------------------------------------
--                          Generate_packet function
-- Function return type: packet class type
-- Called in variables function: n:node class object,
--                               s:integer,
--                               d:integer,
--                               flag:integer
-- Returned function variable : Current_Packet packet class type
-- The called function passes the node information as variable n
-- of type class node1, this variable contains all node infromation
-- such as conversion type number of inputs, number of outputs,
-- wavelengths, number of fibers, Next-node and Previous noed etc.
-- called functions:
-- GenerateSeed: To generate a new interarrival time seed and packet
-- length seed
-- Generateu: To genetrate a uniform random number
-----------------------------------------------------------------*/
packet *Generate_packet(int s, int d, node1 n[],int flag)
{
   double u=0.0;
   unsigned __int64 temp_time=0;
   unsigned __int64 temp_length=0;
   packet *Current_Packet=NULL;
   Current_Packet=new packet();
   Current_Packet->Source=s;
   Current_Packet->Destination=d;
```

```
   Current_Packet->Packet_Type='D';
   Current_Packet->C_Flag=0;
   Current_Packet->Packet_Status=0;
   Current_Packet->Light_Path=NULL;
   if (n[s].Node_Type!='s')
      Current_Packet->Current_Node=s;
   else
      Current_Packet->Current_Node=n[s].Next_Node;
   Current_Packet->Packet_ID=n[s].Last_Packet+1;
   n[s].Last_Packet=n[s].Last_Packet+1;
   u=GenerateU(n[s].Node_Seed[0]);
   n[s].Node_Seed[0]=GenerateSeed(n[s].Node_Seed[0]);
   temp_time=(unsigned __int64)((-1/n[s].Lambda)*log(1-u));
   if (n[s].Node_Type!='s')
   {
      Current_Packet->Initial_Time=n[s].Last_Time+temp_time;
      Current_Packet->Arrival_Time=n[s].Last_Time+temp_time;
      Current_Packet->Departure_Time=n[s].Last_Time+temp_time;
      Current_Packet->End_Time=Current_Packet->Departure_Time;
      n[s].Last_Time=Current_Packet->Initial_Time;
   }else
   {
      unsigned __int64 x=n[s].Last_D+n[s].Last_Length;
    unsigned __int64 x1=n[s].Last_Time+temp_time;
      if ((x1<=x)&&(flag==1))
      {
         Current_Packet->Initial_Time=x1;
         Current_Packet->Arrival_Time=x1;
         Current_Packet->Departure_Time=x+1;
         Current_Packet->End_Time=Current_Packet->Departure_Time;
         n[s].Last_Time=x1;
     n[s].Last_D=x+1;
      }else
  {
  Current_Packet->Initial_Time=x1;
  Current_Packet->Arrival_Time=x1;
  Current_Packet->Departure_Time=x1;
  Current_Packet->End_Time=Current_Packet->Departure_Time;
  n[s].Last_Time=x1;
  n[s].Last_D=x1;
}
   }
   u=GenerateU(n[s].Node_Seed[1]);
   n[s].Node_Seed[1]=GenerateSeed(n[s].Node_Seed[1]);
```

```
   temp_length=(unsigned __int64)((-1/n[s].Mu)*log(1-u));
   Current_Packet->Packet_Length=temp_length;
   n[s].Last_Length=temp_length;
   Current_Packet->Hop_Count=0;
   Current_Packet->Prev_Packet=NULL;
   Current_Packet->Next_Packet=NULL;
   if(n[s].Node_Type=='s')
   {
Current_Packet->Lambda=-1;
Current_Packet->F=-1;
 }
   if(s!=0)
   {
    double r = GenerateU(n[s].Node_Seed[3]);
    n[s].Node_Seed[3]=GenerateSeed(n[s].Node_Seed[3]);
    int y = (int) (r*n[Current_Packet->Current_Node].Out_W);
    Current_Packet->Lambda=y;
    r = GenerateU(n[s].Node_Seed[4]);
    n[s].Node_Seed[4]=GenerateSeed(n[s].Node_Seed[4]);
    y = (int) (r*n[Current_Packet->Current_Node].Out_F);
    Current_Packet->F=y;
   }
   if (n[s].Node_Type=='s')
   {
     Current_Packet->Packet_Tag=1;
   Current_Packet->Packet_Status=0;
   }
   else
   {
      Current_Packet->Packet_Tag=0;
   }
   spacket++;
   packetL+=Current_Packet->Packet_Length;
   return Current_Packet;
}
/*-------------------------------------------------------------
--                           Get_Path2 function
-- This function is implemented to find a route between
-- source-destination pair.
-- Function return type: Path class object
-- Called in variables function: n1:node class object,
-- No_of_Nodes: integer,current_packet: packet class object
-- Returned function variable : head: path class object
-- The called function passes the node information as
```

```
-- variable n1 of type class node1, this variable contains
-- all node infromation such as conversion type number of
-- inputs, number of outputs, wavelengths, number of fibers,
-- Next-node and Previous noed etc. also the Current_packet
-- that wants to find a path to its destination.
------------------------------------------------------------*/
path *Get_Path2(node1 n1[], int No_of_Nodes, packet *current_packet)
{
 path *head=new path(n1[current_packet->Current_Node].Out_F);
   head->Next=NULL;
   head->Prev=NULL;
   head->Same=NULL;
   path *temp=NULL;
 path *temp1=NULL;
   temp=head;
 temp1=head;
 //+++++++++++++++++++++++++++++++++++++++++++++++++++++++
 // extrac the current packet current node and destination
 //+++++++++++++++++++++++++++++++++++++++++++++++++++++++
 int c= current_packet->Current_Node;
 int d=current_packet->Destination;
   int i=0;
   //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
   //check the outgoing wavelngths of the output of the current
   // to consrtuct the grid data structure
   //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
   while (i<n1[c].Out_W)
   {
      temp1->Node_No=n1[c].Node_No;
      temp1->w=i;
      for (int j=0;j<n1[c].Out_F;j++)
      {
     temp1->F[j][0]=j;
          if (n1[c].Out_Port[i][j]==1)
              temp1->F[j][1]=1;
          else if (n1[c].Out_Port[i][j]==0)
              temp1->F[j][1]=0;
      }
  i++;
  if(i<n1[c].Out_W)
  {
    temp1=new path(n1[c].Out_F);
    temp->Next=temp1;
    temp1->Prev=temp;
```

```
    temp=temp1;
    temp->Next=NULL;
    temp->Same=NULL;
 }
   }
   temp=head;
   path *temp2=NULL;
  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
//set the c to the next node of the current node
// copmare the next node of the current node of
//the packet object with desitnation of the pakcet
// if the next nod is qual to the destination that means
// the grid structure will have just one level. if the next
// node is the desitnation enter the loop and cheack the outgoing
// wavlengths of the next node and so on till the packet
//destination reached.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 c=n1[c].Next_Node;
   temp2=NULL;
   while(c!=d)
   {
     if (n1[c].Node_Type!='d')
     {
       while (temp!=NULL)
     {
     temp1=new path(n1[c].Out_F);
    temp1->Next=NULL;
    temp1->Same=NULL;
    temp1->Prev=NULL;
          temp1->Node_No=n1[c].Node_No;
          temp1->w=temp->w;
          for (int j=0;j<n1[c].Out_F;j++)
          {
            temp1->F[j][0]=j;
            if (n1[c].Out_Port[temp->w][j]==1)
              temp1->F[j][1]=1;
            else
              temp1->F[j][1]=0;
          }
          temp->Same=temp1;
      if (temp->Prev!=NULL)
      {
       temp2->Next=temp1;
       temp1->Prev=temp2;
```

```
      }
          temp=temp->Next;
          temp2=temp1;
   }
}
    temp=head;
    while(temp->Same!=NULL)
{
      temp=temp->Same;
c=n1[temp->Node_No].Next_Node;
}
  }
  return head;
}
/*----------------------------------------------------------
--                          cleanm function
-- This function is implemented to free the memory
-- locations that are reserved in the Get_path to
-- construct the grid data structure
-- Function return type: Void
-- Called in variables function: temp:ptah class object
-- Returned function variable : NULL
-----------------------------------------------------------*/
void cleanm (path *temp)
{
    path *t=temp;
    path *temp1=temp->Next;
    path *temp2=temp->Same;
    while(temp!=NULL)
    {
       temp2=temp->Same;
       while (temp->Same!=NULL)
       {
        temp->Same=temp2->Same;
        delete temp2;
        temp2=temp->Same;
        t=temp;
       }
       delete temp;
       temp=temp1;
       if (temp1!=NULL)
         temp1=temp1->Next;
       t=temp;
    }
```

```
    delete t;
}
/*-----------------------------------------------------------
--                          Random_Assign function
-- This function is implemented to check grid data structure
-- and find a continous wavelength between the source and
-- destination. if more the one wavelength is found one of
-- them will be picked randomly.
-- Function return type: packet class object
-- Called in variables function: n1 node class object,
--                              No_of_Nodes integer,
--                              c packet class object,
--                              h path class object
-- Returned function variable : c: packet class object
-- The called function passes the node information in the
-- variable n1 of type class node1, the current packet
-- infromation in the variable c, and the path
-- information from the packet's current node to the
-- packet's destinatio.
----------------------------------------------------------*/
packet *Random_Assign(packet *c, node1 n[], int No_of_Nodes, path *h)
{
  path *temp=h;
  int count=0;
  double r=-1;
  int y=-1;
//+++++++++++++++++++++++++++++++++++++++++++++
// first the number of wavlengths in grid data
// structure is found
//+++++++++++++++++++++++++++++++++++++++++++++
  while (temp!=NULL)
  {
    count++;
    temp=temp->Next;
  }
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// randomly pick one of the wavelengths that are in grid
// path to strat from
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  temp=h;
  r = GenerateU(n[temp->Node_No ].Node_Seed[3]);
  n[temp->Node_No].Node_Seed[3]=GenerateSeed(n[
       temp->Node_No].Node_Seed[3]);
  y = (int) ((r*count));
```

```
  count=0;
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// Move the header of the path grid to the selected wavelength
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  while(temp!=NULL)
  {
    if (count==y)
    break;
    temp=temp->Next;
    count++;
   }
   light_path *head=new light_path();
   head->Next=NULL;
   light_path *l=head;
   light_path *l1=head;
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// check all level in the path grid for the selected wavelength
// to find out if this wavelength is available on all hops.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  while (temp!=NULL)
  {
    count=0;
    int index=n[temp->Node_No].Out_F;
    int *Free=new int[index];
    for (int i=0;i<n[temp->Node_No].Out_F;i++)
    Free[i]=-1;
    for( i=0;i<n[temp->Node_No].Out_F;i++)
    {
      if(temp->F[i][1]==1)
      {
        Free[count]=i;
        count++;
      }
     }
     r=GenerateU(n[temp->Node_No].Node_Seed[4]);
     n[temp->Node_No].Node_Seed[4]=GenerateSeed(n[
           temp->Node_No].Node_Seed[4]);
     y = (int) ((r*count));
     l->Node_No=temp->Node_No;
     l->w=temp->w;
     l->C=0;
     l->F=Free[y];
     delete[] Free;
     if(temp->Same!=NULL)
```

```
      {
        l=new light_path();
        l1->Next=l;
        l->Next=NULL;
        l1=l;
      }
      temp=temp->Same;
    }
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// if a contious light path is found the head will point to that
// wavelength otherwise it will point to NULL
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
   c->Light_Path=head;
//+++++++++++++++++++++++++++++++++++++++++++++++++++++
// call clean memory fuction to clean up all memory
// loactions researved for this packet to be
// avialable for next event.
//+++++++++++++++++++++++++++++++++++++++++++++++++++++
   cleanm(h);
   return c;
}
/*-----------------------------------------------------------------
--                           Lest_Loaded function
-- This function is implemented to check grid data structure
-- and assign the least loaded wavelength to the current packet
-- if a countious wavelength is found.
-- wavelength that exhusted on at least one hop.
-- Function return type: packet class object
-- Called in variables function: n1:node class object,
-- No_of_Nodes: integer,c: packet class object,
-- h: path class object
-- Returned function variable : c: packet class object
-----------------------------------------------------------------*/
packet *Lest_Loaded(packet *c, node1 n[], int No_of_Nodes, path *h)
{
  Least_Loaded_Path *L=NULL;
  Least_Loaded_Path *L1=NULL;
  int flag=0;
  path *temp=h;
  path *head=h;
  while (temp!=NULL)
  {
   L=new Least_Loaded_Path();
   L->head=temp;
```

```
   h=temp;
//+++++++++++++++++++++++++++++++++++++++++++++
// check all the first level wavelengths to
// idntify the least loaded
//+++++++++++++++++++++++++++++++++++++++++++++
   while (h!=NULL)
   {
     L->Node_No=h->Node_No ;
     L->W=h->w;
     int index=0;
     for (int i=0;i<n[h->Node_No].Out_F;i++)
     {
       if (h->F[i][1]==1)
       {
         index=i;
         break;
       }
      }
//++++++++++++++++++++++++++++++++++++++++++++++++++++
// check the least picked wavelength from the first
// level if it avilable on all hops
//++++++++++++++++++++++++++++++++++++++++++++++++++++
   for(i=index+1;i<n[h->Node_No].Out_F ;i++)
   {
     if((n[h->Node_No].weight[h->w][i]<n[
                     h->Node_No].weight[h->w][index])
        &&(h->F[i][1]==1))
     {
        h->F[index][1]=0;
        index=i;
     }
     else
     {
       h->F[i][1]=0;
     }
    }
    L->F=index;
    L->sum=L->sum+n[h->Node_No].weight[h->w][index];
    h=h->Same;
   }
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// if flag=0 means if the picked wavelngth is not available
// on all hops delete it from the grid data structure.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
      if (flag==0)
      {
        L1=L;
        flag=1;
      }else if(L->sum<L1->sum)
      {
        delete L1;
        L1=L;
      }else
        delete L;
      temp=temp->Next;
     }
     h=L1->head;
     light_path *l_path=new light_path();
     light_path *h_path=l_path;
     light_path *l1_path=l_path;
  //++++++++++++++++++++++++++++++++++++++
  // search for the next loaded wavelength
  //++++++++++++++++++++++++++++++++++++++
     while (h!=NULL)
     {
       l_path->Node_No=h->Node_No;
       l_path->w=h->w;
       l_path->C=0;
       l_path->Next=NULL;
       for (int i=0;i<n[h->Node_No].Out_F;i++)
       {
         if (h->F[i][1]==1)
         {
          l_path->F=h->F[i][0];
          break;
         }
        }
        if(h->Same!=NULL)
        {
          l_path=new light_path();
          l1_path->Next=l_path;
          l1_path=l_path;
        }
        h=h->Same;
       }
       l_path=h_path;
       while(l_path!=NULL)
       {
```

```
      n[l_path->Node_No].weight[l_path->w][l_path->F]++;
      l_path=l_path->Next;
    }
    c->Light_Path=h_path;
    cleanm(head);
    return c;
}
/*------------------------------------------------------
--                          C_Random_Assign function
-- This function is used to assign a light path request
-- a wavelength with minimum nuber of conversiion
-- Function return type: packet class object
-- Called in variables function: n1:node class object,
--                                 No_of_Nodes: integer
--                                 c: packet class object,
--                                 h: path class object
-- Returned function variable : c: packet class object
-----------------------------------------------------------*/
packet *C_Random_Assign(packet *c,node1 n[],int No_of_Node,path *h)
{
  path *temp=h;
  path *temp1=h;
  path *temp2=NULL;
  light_path *head=NULL;
  light_path *l=NULL;
  light_path *l1=NULL;
  int flag=0;
  double r=0;
  int y=-1;
  while (temp->Next!=NULL)
  {
   temp=temp->Next;
  }
  while(temp!=NULL)
  {
   temp->Next=temp1;
   temp1->Prev=temp;
   temp=temp->Same;
   temp1=temp1->Same;
  }
  temp=h;
  if((c->Lambda!=-1)&&(c->F!=-1))
  {
    while( temp!=NULL)
```

```
  {
   if(temp->w==c->Lambda)
   {
     temp1=temp;
     break;
   }
   temp=temp->Next;
  }
}
else
{
  r = GenerateU(n[temp->Node_No].Node_Seed[4]);
  n[temp->Node_No].Node_Seed[4]=GenerateSeed(n[
        temp->Node_No].Node_Seed[4]);
  y = (int) (r*n[c->Current_Node].Out_W);
  while( temp!=NULL)
  {
   if(temp->w==y)
   {
     temp1=temp;
     break;
   }
   temp=temp->Next;
 }
}
temp1=temp;
temp2=temp;
do
{
  l=Countious_Wavelength(c,n,temp);
  if(l!=NULL)
  {
   if(c->Source==0)
   {
     c->Light_Path=l;
     break;
   }
   if(l->w!=c->Lambda)
   {
    if(n[c->Current_Node].C_T==1)
    {
      if(n[c->Current_Node].C[l->F]>0)
      {
       l->C=1;
```

```
 c->Light_Path=l;
 break;
}
else
{
  c->Light_Path=NULL;
  head=l;
  while(l!=NULL)
  {
    head=l->Next;
    delete l;
    l=head;
  }
 }
}
else if(n[c->Current_Node].C_T==2)
{
 if(n[c->Current_Node].C[0]>0)
 {
   l->C=1;
   c->Light_Path=l;
   break;
 }
 else
 {
   c->Light_Path=NULL;
   head=l;
   while(l!=NULL)
   {
     head=l->Next;
     delete l;
     l=head;
   }
  }
 }
 else if (n[c->Current_Node].C_T==3)
 {
   if (n[c->Current_Node].C[0]>0)
   {
     l->C=1;
     c->Light_Path=l;
     break;
   }
   else
```

```
                {
                  c->Light_Path=NULL;
                  head=l;
while(l!=NULL)
                  {
                    head=l->Next;
                    delete l;
                    l=head;
                  }
                }
              }
              else
                c->Light_Path=NULL;
             }
             else
             {
               c->Light_Path=l;
               break;
             }
           }
           temp=temp->Next;
         } while (temp!=temp1);
         if(c->Light_Path==NULL)
         {
           temp1=temp2;
           do
           {
             l=N_Conversion(c,n,temp1);
             if((head==NULL)&&(l!=NULL))
             {
               head=l;
               l1=head;
               temp2=temp1;
             }
             if(l==NULL)
             {
               temp1=temp1->Next;
               if(temp1==temp2)
               break;
             }
             else
             {
               temp1=temp1->Same;
               temp2=temp1;//temp2->Same;
```

```
  }
  if(l!=NULL)
  {
    l1->Next=l;
    l1=l;
  }
  l=NULL;
}while (temp1!=NULL);
if (head!=NULL)
{
  l=head;
  while(l!=NULL)
  {
    if(l->Next!=NULL)
    {
      if(l->Node_No==l->Next->Node_No)
       l->Next=NULL;
    }
    l=l->Next;
   }
   l=head;
   while(l->Next!=NULL)
   l=l->Next;
   if(n[l->Node_No].Next_Node==c->Destination)
   {
     light_path *l2=new light_path();
     l2->Next=NULL;
     l2->Node_No=c->Current_Node;
     l2->w=c->Lambda;
     l2->F=c->F;
     l=head;
     if(l2->w!=-1)
       l1=l2;
     else
       l1=l;
     do
     {
       if(l->w!=l1->w)
       {
         if(n[l->Node_No].C_T==1)
         {
           if(n[l->Node_No].C[c->F]>0)
           {
            l->C=1;
```

```
}
else
{
  c->Light_Path=NULL;
  l=head;
  while(l!=NULL)
  {
    head=l->Next;
    delete l;
    l=head;
  }
 }
}
else if(n[l->Node_No].C_T==2)
{
  if(n[l->Node_No].C[0]>0)
  {
   l->C=1;
  }
  else
  {
   c->Light_Path=NULL;
   l=head;
   while(l!=NULL)
   {
    head=l->Next;
    delete l;
    l=head;
   }
  }
 }
 else if (n[l->Node_No].C_T==3)
 {
   if (n[l->Node_No].C[0]>0)
   {
    l->C=1;
   }
   else
   {
     c->Light_Path=NULL;
     l=head;
     while(l!=NULL)
     {
      head=l->Next;
```

```
                delete l;
                l=head;
               }
             }
            }
            else
            {
             c->Light_Path=NULL;
             l=head;
             while(l!=NULL)
             {
               head=l->Next;
               delete l;
               l=head;
             }
           }
         }
         l1=l;
         if(l!=NULL)
          l=l->Next;
      }while (l!=NULL);
      delete l2;
    }
    else
    {
     c->Light_Path=NULL;
     l=head;
     while(l!=NULL)
     {
      head=l->Next;
      delete l;
      l=head;
     }
   }
  }
  if (head!=NULL)
    c->Light_Path=head;
}
temp=h;
temp1=h->Prev;
while(temp!=NULL)
{
 temp->Prev=NULL;
 temp=temp->Same;
```

```
        temp1->Next=NULL;
        temp1=temp1->Same;
       }
       cleanm(h);
       return c;
}
/*----------------------------------------------------------
--                          Least_Used function
-- This function is used as main wavelength assignment
-- which means from this function one of the above
-- function is caaled depned on the wavelength assignment
-- algorithm that selected.
-- Function return type: packet class object
-- Called in variables function: n1:node class object,
--                                No_of_Nodes: integer
--                                c: packet class object,
--                                h: path class object
-- Returned function variable : c: packet class object
-------------------------------------------------------------*/
packet *Least_Used(packet *c, node1 n[], int No_of_Nodes)
{
  path *temp=NULL;
  temp=Get_Path2(n,No_of_Nodes,c);
  path *temp1=temp;
  path *temp2=temp;
  path *temp3=temp;
  path *h=temp;
  int flag=0;
  if(Cflag==0)
  {
   while (temp1!=NULL)
   {
    temp2=temp1;
    while (temp2!=NULL)
    {
     flag=0;
     for (int i=0;i<n[temp1->Node_No].Out_F;i++)
     {
      if (temp2->F[i][1]==1)
      {
       temp2=temp2->Same;
       flag=1;
       break;
      }
```

```
     }
     if (flag==0)
     {
      temp2=temp1;
      temp1=temp1->Next;
      if(temp2->Prev==NULL)
        h=temp2->Next;
      while(temp2!=NULL)
      {
       temp3=temp2;
       if(temp2->Prev!=NULL)
         temp2->Prev->Next=temp2->Next;
       if (temp2->Next!=NULL)
         temp2->Next->Prev=temp2->Prev;
       temp2=temp2->Same;
       delete (temp3);
     }
     break;
    }
   }
   if(flag!=0)
     temp1=temp1->Next;
  }
  if((h!=NULL)&&(Assignment_Algorithm==1))
    c=Lest_Loaded(c,n, No_of_Nodes,h);
  else if((h!=NULL)&&(Assignment_Algorithm==2))
    c=Random_Assign(c,n,No_of_Nodes,h);
  else
    c->Light_Path=NULL;
 }
 else
   c=C_Random_Assign(c,n,No_of_Nodes,h);
 return c;
}
packet *Reach_Destination(packet *c, node1 n[], int NO_of_Node);
/*---------------------------------------------------------------
--                          Tag1_Packet_Process function
-- in this function the current packet under perocess
-- information is updated. the pakcet passed to this
-- function after its ligt path resources are researved
-- on all hops.
-- Function return type: packet class object
-- Called in variables function: n1:node class object,
-- No_of_Nodes: integer,h: packet class object
```

```
-- Returned function variable : c: packet class object
-------------------------------------------------------------*/
packet *Tag1_Packet_Process(packet *h, node1 n[], int NO_of_Node)
{
  light_path *l=NULL;
  int Processing_Time=0;
  unsigned long Propagation_Delay=0;
  packet *temp=NULL;
  packet *temp1=NULL;
  list LIST;
  temp=h;
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// Check if the packet will utilize a wavelength converter in
// the current node. Also it delete the current node information
// for the light path list.
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
if (temp->Light_Path->C==1)
      temp->C_Flag=1;
else
     temp->C_Flag=0;
  temp->Lambda=temp->Light_Path->w;
  temp->F=temp->Light_Path->F;
//---------------- Update Light Path ----------------
  l= temp->Light_Path;
  temp->Light_Path=temp->Light_Path->Next;
  l->Next=NULL;
  delete l;
//--------------------------------------------------
  temp1=C_packet(temp);
temp->Current_Node=n[temp->Current_Node].Next_Node;
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// Check for the destination. if the packet reach its destinatio
// will be deleted from the event list
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
if (temp->Current_Node==temp->Destination)
{
h=Reach_Destination(temp,n,NO_of_Node);
}
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// in this section the packet's time get updated also the event list
// updated according to the current new infromation
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
else
{
```

```
  Propagation_Delay=( unsigned long) ((n[temp->Current_Node].Distance
  /200000.0)/(1.0/1000000000.0));
  temp->Arrival_Time=temp->Departure_Time+ Propagation_Delay;
    temp->Departure_Time=temp->Arrival_Time+Processing_Time;
    temp->End_Time=temp->Departure_Time;
  temp->Hop_Count++;
    temp->Packet_Status=1;
    h=temp->Next_Packet;
  h->Prev_Packet=NULL;
    temp->Next_Packet=NULL;
    h=LIST.Insert_To_List(h,temp);
}
  h=LIST.Insert_To_List(h,temp1);
  return h;
}

/*--------------------------------------------------------------------
--                            Releas_Resources function
-- The node reources infromation will be upadted in this function.
-- This function will be called only with resources event
-- Function return type: packet class object
-- Called in variables function: c: packet class object,
-- n1:node class object
-- Returned function variable : c: packet class object
---------------------------------------------------------------------*/
packet *Releas_Resources(packet *c, node1 n[])
{
list LIST;
packet *temp=NULL;
temp=c;
n[c->Current_Node].Out_Port[c->Lambda][c->F]=1;
    if (c->C_Flag>0)
    {
       if (n[c->Current_Node].C_T==1)
          n[c->Current_Node].C[temp->F]++;
       else
          n[c->Current_Node].C[0]++;
    }
 c=LIST.Delete_From_List(temp);
    return c;
}
/*--------------------------------------------------------------------
--                            Reach_Destination function
-- This function is invoked when the pakcet arrive to its
```

```
-- destination to update the arrived packet counter and generate
-- a new data packet from the same source and wavelength
-- Function return type: packet class object
-- Called in variables function: c: packet class object,
-- n1:node class object
-- Returned function variable : c: packet class object
---------------------------------------------------------------------*/
packet *Reach_Destination(packet *c, node1 n[], int No_of_Nodes)
{
  list LIST;
  packet *temp=NULL;
  n[c->Source].Arrived_Packet++;
  int source1=c->Source;
  int d=Destination_Generation(source1,No_of_Nodes,n);
  c=LIST.Delete_From_List(c);
  temp=Generate_packet(source1,d,n,1);
  c=LIST.Insert_To_List (c,temp);
  return c;
}
/*--------------------------------------------------------------------
--                           Blocked_Packet function
-- This function is invoked when a pakcet drop event occurs.
-- So, the dropped packets counter will upadted and a new pakcet
-- will be generated. Also, the event list will be updated by
-- deleting the dropped packet and insert the new generated packet.
-- Function return type: packet class object
-- Called in variables function: c: packet class object,
-- n1:node class object
-- Returned function variable : c: packet class object
---------------------------------------------------------------------*/
packet *Blocked_Packet(packet *c, node1 n[],int No_of_Nodes)
{
  Blocked++;
  list LIST;
  int source1=c->Source;
  packet *temp;
  temp=NULL;
  n[c->Source ].blocked++;
  c->Packet_Status=3;
  c=LIST.Delete_From_List(c);
  int d=Destination_Generation(source1,No_of_Nodes,n);
  return c;
}
/*---------------------------------------------------------------
```

```
--                            Instanince_Reservation function
-- This function was implmwnted to do the resources reservation
-- for the current packet after these resources identified.
-- Function return type: packet class object
-- Called in variables function: c: packet class object,
-- n1:node class object
-- Returned function variable : c: packet class object
-----------------------------------------------------------------*/
packet *Instanince_Reservation(packet  *c, node1 n[],int No_of_Nodes)
{
  n[1].RT[0][0]+=1;
  if(n[c->Source].Node_Type=='s')
    n[c->Source ].RT[0][0]+=c->Packet_Length ;
  if((Assignment_Algorithm==1)||(Assignment_Algorithm==2))
    c=Least_Used(c,n, No_of_Nodes);
  if (c->Light_Path==NULL)
  {
    c->Packet_Status=3;
    c->Light_Path=NULL;
    return c;
  }else
  {
   c->Packet_Status=1;
   light_path *l;
   l=NULL;
   l=c->Light_Path;
   if(c->Source==0)
     n[0].Th[0][0]+=c->Packet_Length;
   while(l!=NULL)
   {
     if(c->Source!=0)
       n[l->Node_No].Th[l->w][l->F]+=c->Packet_Length;
     n[l->Node_No].Out_Port[l->w][l->F]=0;
     if (l->C>0)
     {
       if (n[l->Node_No].C_T==1)
        n[l->Node_No].C[l->F]--;
       else
        n[l->Node_No].C[0]--;
     }
     l=l->Next;
    }
  }
  return c;
```

```
}
packet *Data_Packet_Process(packet *c, node1 n[],int No_of_Nodes)
{
  if(c->Light_Path==NULL)
  {
   n[c->Current_Node].Pro_Packet++;
   if (c->Source==0)
    n[c->Source].Pro_Packet++;
   if (Reservation_Method==0)
   {
     c=Instanince_Reservation(c, n,No_of_Nodes);
     if (c->Light_Path!=NULL)
     {
       c=Tag1_Packet_Process(c, n, No_of_Nodes);
     }
     else if(c->Light_Path==NULL)
       c=Blocked_Packet(c,n, No_of_Nodes);
    }
   }else
   {
     n[c->Current_Node].Pro_Packet++;
     c=Tag1_Packet_Process(c,n,No_of_Nodes);
   }
   return c;
}
/*-----------------------------------------------------------------
                        Packet_Processing function
-- Function return type: packet class type
-- Called in variables function: c - packet class type,
-- node -node class type
-- Returned function variable : c - packet class type
-- depend of the event type if data or release this
-- function call either Conversion1 function or Releas_Resources
-----------------------------------------------------------------*/
packet *Packet_Processing(packet *c, node1 n[], int No_of_Nodes)
{
  if(c->Packet_Type=='D')
  {
    c=Data_Packet_Process(c,n,No_of_Nodes);
  }
  else if(c->Packet_Type=='R')
  {
   c=Releas_Resources(c,n);
  }
```

```
  return c;
}
/*----------------------------------------------------------
                      List_Processing function
-- Function return type: Void
-- Called in variables function: head-packet class type,
-- node-node class type
-- Returned function variable : NULL
-- This function get the header of the event list and pass
-- it to the packet processing function to be processed.
-- Also this function keep track of how many packets get
-- processed and calculate the blocking probability
-- if this number reach 1000000 packet event. if the
-- tolerance between the current blocking probability
-- and the pervious from the pervious iteration less
-- than 0.0001 the simulator will be ended after the
-- computer memory get cleaned by deleting all
-- the event list.
----------------------------------------------------------*/
void List_Processing(packet *head, packet *tail, node1
 n[], int No_of_Nodes)
{
   packet *temp;
   temp=NULL;
   temp=head;
   long Packet_Count=0;
   double pb=0;
   double pb1=0;
   int exit=0;
   double epslon=0.0;
   long Num=1;
   while (temp!=NULL)
   {
     if ((temp->Packet_Type=='D')&&(temp->Packet_Status==0))
     {
      Packet_Count++;
     }
     if(Packet_Count==(Num*1000))
     {
      Num++;
     }
     Simulator_end_Time=temp->Departure_Time;
     if (temp->Packet_Type=='D')
      Processed++;
```

```
temp=Packet_Processing(temp,n,No_of_Nodes);
if((Packet_Count==100000)&&(exit!=1))
{
  Num=0;
  double summp=0;
  for (int i=0;i<No_of_Nodes;i++)
   summp+=n[i].blocked;
  pb=(double)Blocked/(double)Processed;
  epslon=fabs(pb-pb1);
  if(floor(epslon*1000)<=1)
  {
    exit=1;
    for(int i=0;i<No_of_Nodes;i++)
    {
      n[i].blocked=0;
      n[i].Last_Packet=0;
      n[i].Arrived_Packet=0;
    }
    Packet_Count=0;
   }else
   {
     for(int i=0;i<No_of_Nodes;i++)
     {
       n[i].blocked=0;
       n[i].Last_Packet=0;
       n[i].Arrived_Packet=0;
     }
     Packet_Count=0;
    }
    pb1=pb;
   }else if ((exit==1)&&(Packet_Count==100000))
   {
     pb=(double)Blocked/(double)Processed;
     for (int i=0;i<No_of_Nodes;i++)
     {
       for( int k=0;k<n[i].Out_F;k++)
       {
         for (int j=0;j<n[i].Out_W;j++)
         {
           if (n[i].Th[j][k]<0)
             n[i].Th[j][k]+=Simulator_end_Time;
           if (n[i].RT[0][0]<0)
             n[i].RT[0][0]+=Simulator_end_Time;
          }
```

```
              }
            }
            break;
           }
      }
      while(temp!=NULL)
      {
        head=temp->Next_Packet;
        delete temp;
        temp=head;
      }
}
/*------------------------------------------------------------
                       Event_Handeler function
-- Function return type: Void
-- Called in variables function: node -node class type
-- Returned function variable : NULL
-- This function generate the initial event list and
-- set up the simulator starting time to the time of the
-- fisrt event will be processed
-------------------------------------------------------------*/
void Event_Handeler(node1 n1[],int NO_of_Node)
{
   int d;
   packet *Event_Head=NULL;
   packet *Event_Tail=NULL;
   list LIST;
   packet *c1=NULL;
   for (int i=0;i<NO_of_Node;i++)
   {
     if ((n1[i].Node_Type!='d')&&(n1[i].Lambda>0))
     {
       for( int j=0;j<3;j++)
       {
        d=Destination_Generation(i,NO_of_Node,n1);
        c1=Generate_packet(n1[i].Node_No,d,n1,1);
          Event_Head=LIST.Insert_To_List(Event_Head,c1);
       }
     }
   }
 Simulator_start_Time=Event_Head->Departure_Time;
 Event_Tail=LIST.Get_Tail(Event_Head);
   List_Processing(Event_Head,Event_Tail,n1,NO_of_Node);
 signed __int64 diff=Simulator_end_Time-Simulator_start_Time;
```

```
}

void Find_Lambda (node1 n[], int No_of_Nodes, double load[])
{
   n[0].Lambda=load[0]*n[0].Mu
      *(n[n[0].Next_Node].Out_W*n[n[0].Next_Node].Out_F);
   for (int i=1;i<No_of_Nodes;i++)
      n[i].Lambda=(double)(load[i]*n[i].Mu)*(n[i].Out_W*n[i].Out_F);
}
/*-----------------------------------------------------------
                       Construct_Toplogy function
-- in this function the toplogy file is read and the
-- nodes paramters are set accoriding the values of these
-- paramters in the toplogy file
-- Function return type: Void
-- Called in variables function: n1:node class object
-- Returned function variable : Null
------------------------------------------------------------*/
void Construct_Toplogy(node1 n1[])
{
  int F=0;
  int w=0;
  int C=-1;
  double mu=0;
  int length=0;
  __int64 Temp_Seed=0;
  double Mean_packet_Length=0;
  int NO_Of_Node=0;
  Mean_packet_Length=500000;
  mu=1.0/Mean_packet_Length;
  ifstream out1 ("Toplogy.dat");
  out1>>NO_Of_Node;
  double *load=new double [NO_Of_Node];
  srand(7);
  for (int i=0;i<NO_Of_Node;i++)
  {
    length=0;
    F=0;
    w=0;
    C=0;
    out1>>n1[i].Node_Type;;
    out1>>n1[i].Node_No;
    out1>>n1[i].Next_Node;
    out1>>n1[i].Distance ;
```

```
  out1>>n1[i].Out_F;
  out1>>n1[i].Out_W;
  out1>>n1[i].Prev_Node;
  out1>>length;
  out1>>F;
  out1>>w;
  out1>>n1[i].C_T;
  out1>>n1[i].DC;
  out1>>load[i];
  n1[i].Mu=mu;
  n1[i].alpha=alpha;
  if (n1[i].C_T==1)
  {
    for (int j=0;j<n1[i].Out_F;j++)
    {
       C=(int)ceil(n1[i].DC*n1[i].Out_W);
       n1[i].C[j]=C;
    }
  } else if (n1[i].C_T==2)
  {
     C=(int)ceil(n1[i].DC*n1[i].Out_F*n1[i].Out_W);
     n1[i].C[0]=C;
  }else if(n1[i].C_T==3)
  {
    n1[i].C[0]=n1[i].Out_W*n1[i].Out_F;
  }
  double r= ((double)rand() / (double)(RAND_MAX+1) );
Temp_Seed=(int)(r*10000);
while ((Temp_Seed%2)==0)
{
  r= ((double)rand() / (double)(RAND_MAX+1) );
     Temp_Seed=(int)(r*10000);
  }
  n1[i].Out_Port_Initialization(Temp_Seed);
 }
 Find_Lambda(n1,NO_Of_Node,load);
 cout<<endl<<"0 Instanince reservation";
 cout<<endl<<"1 Backward reservation";
 cout<<endl<<"2 Forward reservation";
 cout<<endl<<"3 Hop-by-Hop Reservation ";
 cout<<endl<<"4 Future Reservation ";
 cout.flush();
 cin>> Reservation_Method;
 cout<<endl<<"0 First Fit Algorithm";
```

```
   cout<<endl<<"1 Least loaded Algorithm";
   cout<<endl<< "2 Random Assignment algorithm ";
   cin>>Assignment_Algorithm;
   cout<<endl<<"Conversion Flag";
   cin>>Cflag;
   cout<<endl<<"1. Random Destination ";
   cout<<endl<<"2. Next Node Destination ";
   cin>>Destination_Flag;
   out1.close();
}
int main()
{
 int NO_Of_Node;
 ifstream out1 ("Toplogy.dat");
 out1>>NO_Of_Node;
 out1.close();
 node1 *n=new node1[NO_Of_Node];
 Construct_Toplogy(n);
 for (int i=0;i<NO_Of_Node;i++)
 n[i].node_info();
 Event_Handeler(n,NO_Of_Node);
 delete [] n;
 return 0;
}
```

Appendix E

ANALYTICAL MODEL SOURCE CODE: BLOCKING PROBABILITY OF OPS

The source code of the analytical model, which is written in Maple, to evaluate the blocking probability of the optical packet switch with share-per-link conversion is shown below. This analytical model is explained in detail in Chapter 5.

```
#######################################################
## File name: OPS.mws
##
## Purpose: To identify the blocking probabilty of
## the OPS employing share-per-link conversion schme.
##
##  Inputs:
##    N : Number of inputs/outputd of the switch
##  rho: Load per wavelengt
##  w  : Number of wavelengths per fiber
##  F  : Number of fibers
##  C  : Number of converters
##
## Outputs:
## PB1 : Switch blocking probability
#######################################################
###########################
# Switch paramters settings
###########################
N:=4:rho:=1.0:w:=32:C:=4:F:=3:
#######################################################
# Probability that there are x arrival on a wavelength
#######################################################
X  := (x) -> binomial(N*F,x)*(((rho)/N)^x)*((1-((rho)/N))^((N*F)-x)):
#############################################################
# The probability that there is n Y's that will constitute T
#############################################################
NY := (n) -> binomial(w,n)*((sum('X(i)','i'=(F+1)..N*F))^n)
*(sum('X(i)','i'=0..F))^(w-n):
```

```
#############################################################
#Probability there are y packets forwarded to the conversion
#bank from a wavelength
##############################################################
Y  := (y) -> X(y+F)/(sum('X(i)','i'=(F+1)..N*F)):
########################################################
#Expected number of forwarded packets to the conversion
########################################################
EY:= sum('j*Y(j)','j'=1..N*F-F):
##############################################################
#Probability that there are g copies of a wavelength are free
##############################################################
G := (g) ->  X(F-g)/(sum('X(i)','i'=0..F)):
#################################################
#G random number probability generating function
#################################################
tmp1:=(sum('G(g)*z^g','g'=0..F))^tt:
################################################################
# Probability that there are beta free wavelengths on the output
################################################################
B := proc(ny,beta)
  if (ny=w) then 0;
  elif (beta=0) then (G(0))^(w-ny);
  else eval(diff(eval(tmp1,tt=(w-ny)),z$beta)/beta!,z=0):
  end if:
end proc:
##############################################################
# Correspondong to those wavelengths that don't contribute to
#Y's considered in formulating T
##############################################################
tmp2:=(sum('Y(y)*z^y','y'=1..(N*F-F)))^tt:
#####################################################
# Probability that there are alpha packets forwarded
# to the conversion
#####################################################
T := (t,k) -> eval(diff(eval(tmp2,tt=t),z$k)/k!,z=0):
######################################################
# Expected number of the total number of packets that
# are forwarded to the conversion
######################################################
ET := sum('NY(ny)*(ny*EY)','ny'=0..w);
#######################################################
#Expected number of accepted packets by the conversion
#######################################################
```

```
EA:=sum('NY(ny)*sum('B(ny,beta)*sum('min(j,min(beta,C))*
T(ny,j)','j'=ny..ny*(N*F-F))','beta'=0...((w-ny)*F))','ny'=1..w);
ED1:=ET-EA;
######################
#Blocking probability
######################
PB1:= ED1/(w*F*rho);
```

Appendix F

SIMULATOR SOURCE CODE: OPTICAL PACKET SWITCH

The simulator sources code for OPS is shown below. This simulator is explained in detail in Chapter 7. The simulator is used to evaluate blocking probability of the multi-fiber OPS with share-per-link conversion option. The results generated using this simulator is presented in Chapter 5.

```
/*-----------------------------------------------------------------
--              OPS Simulator
--                                   By
--                           Fahad A. AL-zahrani
--  This simulator was build as a part of my Ph.D dissertation.
--  This simulator is used to evaluate the blocking
--  probability of the multi-fiber OPS.
--  File Name: Slotted.cpp
--  Inputs:
-- Input: Number of inputs
--    Output: Number of outputs
--    Wi: Number of input wavelengths
--     Wo: Number of output wavelengths
--      F: Number of fibers
-- Load: Load per wavelength
--        C: Number of converters
--  Outputs:
--         Pb: Blcoking probability
--         Uc: Conversion bnak utilization
-----------------------------------------------------------------*/
/*-----------------------------------------------------------------
--                       Global variables
-- In this Part of the simulator all global variables that are
-- used in the different functions of the simulator are defined
-- These variables are:
-- Slot_Time: Is used to keep track of the numebr of the slots
-- P_Seed: Is used as an initial seed for selecting input
-- W_Seed: Is used as an initial seed for selecting wavelength
-- F_Seed: Is used as an initial seed for selecting fiber
```

```
-- C_Seed: Is used as an initial sedd for selecting converters
---------------------------------------------------------------*/
#include "list.h"
double Load=0;
int C=0;
long Slot_Time=1;
long P_Seed=7;
long W_Seed=13;
long F_Seed=31;
long C_Seed=21;
/*--------------------------------------------------------------
--                      Random generator module
-- There are two functions in this module. The First one is
-- used to generate the seed that will be use to generate the
-- next random variable.The second function is the actual
-- implementation of the LCG random generator.
------------------------------------------------------------------*/
/*----------------------------------------------------------------
--                          GenerateSeed function
-- Function return type: long integer
-- Calld in variables funtion: Seed- long integer
-- Returned funtion variable : a - long integer
------------------------------------------------------------------*/
long GenerateSeed(long Seed)
{
   long a;
   a =(long)(abs( (16807 * Seed) % (2147483647)));
   return a;
}
/*-----------------------------------------------------------------
--                          Generateu function
-- Function return type: bouble
-- Calld in variables funtion: Seed- long integer
-- Returned funtion variable : u - double
--------------------------------------------------------------------*/
double GenerateU(long Seed)
{
   double u=0.0;
   u = ((double)(Seed) / (double)2147483647);
   return u;
}
/*------------------------------------------------------------
--                          Any_To_Any function
-- Function return type: packet class type
```

```
-- Called in variables: c packet class type,
--                      node node class type
-- Returned function variable : c - packet class type
-- This function is called when a contention between packets
-- occur. A free wavelength and free converter in the conversion
-- bank that associated to the requested output port are sought,
-- if a free wavelength and free converter is found then they
-- assigned to the passed packet. And change the status of
-- these resources to busy. Otherwise this packet as well as
-- all other packets that destined to the same output port in
-- the current time slot dropped.
---------------------------------------------------------------*/
packet *Any_To_Any(packet *c, node1 *node)
{
  int Choosed_Packet=-1;
  int Choosed_W=-1;
  int Choosed_F=-1;
  int count=0;
  int C_Counter=0;
  double u=-1.0;
  packet *temp=NULL;
  packet *temp1=NULL;
  temp=c;
  while (c!=NULL)
  {
    Choosed_Packet=-1;
    Choosed_W=-1;
    count=0;
    temp=c;
    u=GenerateU(P_Seed);
    P_Seed=GenerateSeed(P_Seed);
    while(temp!=NULL)
    {
      count++;
      temp=temp->After;
    }
    Choosed_Packet=(int)(u*count);
    count=0;
    temp=c;
    while(count!=Choosed_Packet)
    {
      count++;
      if (count!=Choosed_Packet)
        temp=temp->After;
```

```
}
Free_Out_C *Free1=new Free_Out_C[node->Wo*node->F];
count=0;
for (int i=0;i<node->Wo;i++)
{
  for (int k=0;k<node->F;k++)
  {
   if (node->Out_Port[i][temp->Output][k]==1)
   {
    Free1[count].W=i;
    Free1[count].F=k;
    count++;
   }
  }
 }
 if (count>0)
 {
   int Temp_Choose=-1;
   u=GenerateU(W_Seed);
   W_Seed=GenerateSeed(W_Seed);
   Temp_Choose=(int)(u*count);
   Choosed_W=Free1[Temp_Choose].W;
   Choosed_F=Free1[Temp_Choose].F;
   delete [] Free1;
 }
 if ((Choosed_W!=-1)&&(node->C[temp->Output]>0))
 {
  if(temp->Output==0)
    C_Counter++;
  node->Out_Port[Choosed_W][temp->Output][Choosed_F]=0;
  node->C[temp->Output]--;
  temp->C_Flag=1;
  temp->Out_Lambda=Choosed_W;
  temp->F=Choosed_F;
  node->Forwarded_Packet++;
  if (temp==c)
  {
    c=c->After;
    if (c!=NULL)
      c->Before=NULL;
      temp->After=NULL;
  }else
  {
    temp1=temp->Before;
```

```
          temp1->After=temp->After;
          temp1=temp->After;
          temp1->Before=temp->Before;
          temp->Before=NULL;
          temp->After=NULL;
        }
        delete temp;
      }
      else
      {
        if (c->Before!=NULL)
          c=c->Before;
        while (c!=NULL)
        {
          temp=c;
          c=c->After;
          ++node->blocked;
          delete temp;
        }
      }
  }
  node->U_C[C_Counter]++;
  return c;
}
/*----------------------------------------------------------------------
--                            Slot_Initilaze function
-- Function return type: Void
-- Called in variables: n1-node class type
-- Returned function variable : NULL
-- This function is called at the end of every time slot to reset the
-- node resources to the free state.
----------------------------------------------------------------------*/
void Slot_Initilaze(node1 *n)
{
  for (int i=0;i<n->Output;i++)
   n->C[i]=C;
  for (i=0;i<n->Output;i++)
  {
    for(int j=0;j<n->Wo;j++)
    {
      for (int k=0;k<n->F;k++)
        n->Out_Port[j][i][k]=1;
    }
  }
```

```
}

/*-------------------------------------------------------------
--                          Eevnt_Handeler function
-- Function return type: Void
-- Called in variables: n-node class type
-- Returned function variable : NULL
-- In this function, all wavelengths are checked if there
-- is an arrival on the wavelength under consideration then
-- randomly select one of the switch outputs as a destination
-- for this arrival. Every time slot this process is repeated.
-- Every 100000 time slot runs the blocking probability is
-- calculated and compared with blocking probability previous
-- value, If the difference is less than 0.001 the simulators
-- will be ended and all statistics collected.
-------------------------------------------------------------*/
void Event_Handeler(node1 *n)
{
  packet *head=NULL;
  packet *Current_Packet=NULL;
  list LIST;
  double p0=0;
  double p1=0;
  double e=9999;
  double u=0.0;
  int In=-1;
  int Out=-1;
  int F=-1;
  int Break_Flag=0;
  In_V *ch=new In_V[n->Input*n->F];
  int Slot_C=1;
  long Max_T=0;
  double Expected_of_Loss=0;
  int Count_In=0;
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++
// Check if the blocking difference is less than 0.001
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++
  while (e>0.001)
  {
   for(int i=0;i<n->Wi;i++)
   {
    Count_In=0;
    Break_Flag=0;
    for (int k=0;k<n->Input;k++)
```

```
    {
     for (int m=0;m<n->F;m++)
     {
      ch[Count_In].In=k;
      ch[Count_In].F=m;
      Count_In++;
     }
    }
    while (Break_Flag==0)
    {
     Count_In=0;
     for (int k=0;k<n->Input*n->F;k++)
     {
      if (ch[k].F!=-1)
       Count_In++;
     }
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// Randomly select one of the switch inputs and check if there is
// an arrival on the current wavelength.
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
    if (Count_In>0)
    {
     int temp_in=-1;
     u=GenerateU(n->Input_Seed);
     n->Input_Seed=GenerateSeed(n->Input_Seed);
     In=(int)(u*Count_In);
     temp_in=ch[In].In;
     F=ch[In].F;
     ch[In].F=-1;
     ch[In].In=-1;
     In=temp_in;
     Break_Flag=0;
     Count_In=0;
     for (int j=0;j<n->Input*n->F;j++)
     {
      if(ch[j].F!=-1)
      {
       Break_Flag=0;
       ch[Count_In].In=ch[j].In;
       ch[Count_In].F=ch[j].F;
       if(Count_In!=j)
       {
        ch[j].In=-1;
        ch[j].F=-1;
```

```
       }
       Count_In++;
      }
     }
//+++++++++++++++++++++++++++++++++++++++++++++++++++
// Randomaly select one of the siwtch outputs
// and consider it as the current packet destination
//+++++++++++++++++++++++++++++++++++++++++++++++++++
     u=GenerateU(n->Output_Seed);
     n->Output_Seed=GenerateSeed(n->Output_Seed);
     Out=(long)(u*n->Output);
     if (Out==0)
     {
      u=GenerateU(n->Seeds[i][In][F]);
      n->Seeds[i][In][F]=GenerateSeed(n->Seeds[i][In][F]);
//+++++++++++++++++++++++++++++++
// Check if there is an arrival.
//+++++++++++++++++++++++++++++++
     if (u<Load)
     {
      n->Pro_Packet++;
      Current_Packet=new packet();
      Current_Packet->After=NULL;
      Current_Packet->Before=NULL;
      Current_Packet->Input=In;
      Current_Packet->F=F;
      Current_Packet->Lambda=i;
      Current_Packet->Slot_Time=Slot_Time;
      Current_Packet->Output=Out;
      Current_Packet->Out_Lambda=i;
      Current_Packet->C_Flag=-1;
      int F_flag=-1;
      for (int k=0;k<n->F;k++)
      {
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// Check if the requested wavelength is free on any fiber
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      if (n->Out_Port[Current_Packet->Out_Lambda]
          [Current_Packet->Output][k]==1)
      {
       n->Forwarded_Packet++;
       n->Out_Port[Current_Packet->Out_Lambda]
                  [Current_Packet->Output][k]=0;
       delete Current_Packet;
```

```
       F_flag=1;
       break;
      }
     }
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// if the requested wavelength is busy on all fibers. Add this
// packet to the requested output conversion list.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
     if ((F_flag==-1)&&(n->C>0))
     {
      n->h[Current_Packet->Output]=
        LIST.Insert_To_List(n->h[Current_Packet->Output]
                    ,Current_Packet);
     }
    }
   }
  }
  else
  {
    Break_Flag=1;
  }
 }
}
//+++++++++++++++++++++++++++++++++++++++++++++++++++++
// Check if there are packets in the conversion list.
// if so call the conversion function
//+++++++++++++++++++++++++++++++++++++++++++++++++++++
  if ((n->C_T==0)&&(n->h[0]!=NULL))
    n->h[0]=Any_To_Any(n->h[0],n);
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// check how long the simulator run. If the Slot counter
// is equal 100000 calculate the blocking probability and
// comaper it with the previous value.
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 if (Slot_C==100000)
 {
  cout<<endl<<Max_T;
  p0=(double)n->blocked/(double)n->Pro_Packet;
  e=fabs(p0-p1);
  p1=p0;
  Expected_of_Loss=Expected_of_Loss+n->blocked;
  n->blocked=0;
  n->Pro_Packet=0;
  n->Forwarded_Packet=0;
```

```
  Slot_C=0;
 }
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// Initialize the slot counter and call the node initialztion
// function explained above.
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  Slot_C++;
  Slot_Initilaze(n);
  Slot_Time++;
 }
//++++++++++++++++++++++++++++++++++++++++++
// Calculate the conversion bank utilization
//++++++++++++++++++++++++++++++++++++++++++
  u=0;
  for (i=0;i<=C;i++)
  {
   u+=i*n->U_C[i];
  }
  u=(double)u/(double)(C*(Slot_Time-1));
//++++++++++++++++++++++++++++++++++++++++++++++++++
// Print out the   blocking probability and conversion
// bank utilization
//+++++++++++++++++++++++++++++++++++++++++++++++++++
  cout<<endl<<"=============================";
  cout<<endl<<"Blocking= "<<p0<<endl;
  cout<<endl<<"Convter bank utilization = "<<u<<endl;
  cout<<endl<<Slot_Time;
  cout.flush();
}
/*---------------------------------------------
                      Node_Intiliaze function
-- Function return type: node class type
-- Called in variables function:
-- Returned function variable : n
-- This function create the node object and
-- initialize all the node's variables
-----------------------------------------------*/
node1 *Node_Intiliaze()
{
     node1 *n=new node1();
   int Input;
   int Output;
   int Wi;
   int Wo;
```

```
    int C_T;
    int d=0;
    int F=0;
    cout<<"Please Enter the load per wavelength per port ";
    cin>>Load;
    cout<<endl<<"Input = ";
    cin>>Input;
    cout<<endl<<"Ouput = ";
    cin>>Output;
    cout<<endl<<"# of fibers per link = ";
    cin>>F;
    cout<<endl<<"Number of wavelength per input per fiber ";
    cin>>Wi;
    cout<<endl<<"Number of wavelength per output per fiber ";
    cin>>Wo;
    cout<<endl<<"Conversion Type ";
    cin>>C_T;
    cout<<endl<<"No of Converters ";
    cin>>C;
    n->Input=Input;
    n->Output=Output;
    n->F=F;
    n->Wo=Wo;
    n->Wi=Wi;
    n->C_T=C_T;
    if ((C_T==1)||(C_T==2)||(C_T==3))
    {
        cout<<endl<<"Enter d";
        cin>>d;
      }else
        d=0;
      n->Out_Port_Initialization(C,d);
      return n;
}
void main()
{
  node1 *node=NULL;
  node=Node_Intiliaze();
  Event_Handeler(node);
}
```