

AN INVESTIGATION OF FAST RESTORATION AND
EFFICIENT SPARE CAPACITY ASSIGNMENT
TECHNOLOGIES BASED ON
PREPLANNED RESTORATION
IN OPTICAL NETWORKS

By

BYOUNG-YONG LEE

Bachelor of Engineering

Yosu National University

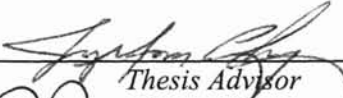
Yosu, Korea

1998

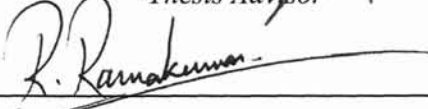
Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
December, 2002

AN INVESTIGATION OF FAST RESTORATION AND
EFFICIENT SPARE CAPACITY ASSIGNMENT
TECHNOLOGIES BASED ON
PREPLANNED RESTORATION
IN OPTICAL NETWORKS

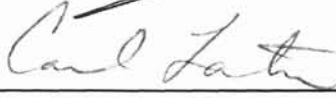
Thesis Approved:



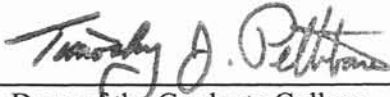
Thesis Advisor



R. Ramakrishnan



Carl Zahr



Dean of the Graduate College

PREFACE

The networking field has changed significantly over the last decade. The most important advancement has been the rapid enhancements in optical fiber switching and wavelength division multiplexing (WDM) technology. The optical networks provide broad bandwidth in a single fiber by making use of the WDM technology. Due to this, the failure of a single fiber may cause a large amount of data to be lost. Therefore, fast recovery in optical networks becomes essential to the network's Quality of Service (QoS) and reliability.

Survivability of optical networks is one of the principal issues in designing a robust and reliable optical network. In current optical networks, the Self-Healing Ring (SHR) architecture is mostly used due to its simple protection mechanism. A link failure in SHR can be recovered by using the Automatic Protect Switching (APS) protocol. The APS protocol is a simple switching method and requires relatively stable recovery time (< 50 ms). However, SHR is not efficient with regards to the optimum usage of channel capacity and cannot deal with multi-link failure within a ring network. Thus, the mesh architecture has been developed to reduce the amount of back-up capacity and also to enhance the reliability. This is accomplished by utilizing the fact that mesh networks have multiple routing paths to select from based on a source to destination node connection.

This thesis analyzes several restoration techniques used in optical networking and compares the performance/requirements in terms of back-up capacity and restoration time. There are two basic survivability methods in optical networks: path restoration and line restoration. These methods can perform failure recovery either by dynamic or preplanned restoration scheme. In addition, this thesis proposes two new sub-optimal line restoration efficient spare capacity assignment (SCA) algorithms, namely the Equally Shared Traffic (EST) and Proportionally Shared Traffic (PST) algorithms. We show the benefits of the proposed algorithms and compare the algorithms with the optimized spare capacity (OSC) solution, which is based on a linear programming (LP) topology.

ACKNOWLEDGMENTS

I wish to express my deep and sincere thanks and gratitude to my advisor Dr. Jong-Moon Chung for his supervision, support, critical suggestions, and inspiration without whom; this thesis would not have been possible. My appreciation and thanks are also due to my committee members, Dr. R. G. Ramakumar and Dr. Carl D. Latino for their invaluable support, assistance, encouragement and guidance throughout my Master's program here at the Oklahoma State University.

I would like to thank the Advanced Communication Systems Engineering Laboratory (ACSEL) and the Oklahoma Communication Laboratory for Networking and Bioengineering (OCLNB) at the Oklahoma State University for supporting resources. I would like to thank my group members for their contribution. I would also like to thank other members of the ACSEL and OCLNB laboratories for their recommendations and support and my friends who made my thesis work an enjoyable and pleasant one.

Finally, I would like to thank my wife, Joung-Hee Park for her encouragement and thank one of research team member, Vizayakumar Kotikalapudi for his helping completing the programming of the simulation experiments.

TABLE OF CONTENTS

Chapter	Page
1. INTRODUCTION	1
2. REVIEW OF RESTORATION TECHNOLOGIES.....	4
2.1 Differences between Restoration and Protection.....	4
2.2 SONET APS Protocol.....	5
2.2.1 Automatic Protection Switching (APS) Architectures	7
2.2.1.1 1+1 APS Architecture.....	7
2.2.1.2 1:1 APS Architecture	8
2.2.1.3 1:N APS Architecture	8
2.3 SONET Self-Healing Ring (SHR) Network.....	9
2.3.1 Unidirectional SHR (U-SHR).....	10
2.3.2 Bi-directional SHR (B-SHR)	11
2.4 Restoration Methods in Mesh Network	12
2.4.1 Line Restoration vs. Path Restoration.....	14
2.4.2 Centralized vs. Distributed Control	15
3. ANALYSIS OF RESTORATION ALGORITHMS.....	17
3.1 Considerations of Preplanned Line Restoration in WDM network	17
3.1.1 Preplanned Line Restoration Procedure.....	20
3.1.2 Preplanned Line Restoration Time	23
3.2 Efficiency Spare Capacity Assignment Algorithms	24
3.3.1 Linear Program (LP) with Min-Max Function [13], [20].....	26
3.3.2 The Equally-Share Traffic (EST) Algorithm.....	29
3.3.3 The Proportionally-Share Traffic (PST) algorithm.....	31
3.4 Restoration Time and Control Packets between the OSC and RSC Reservation... 33	
4. OBSERVATION OF THE RESULTS	40
4.1 Analysis of Performance.....	40
4.2 Actual Traffic Assignment on the Network for Realistic Simulation.....	41
5. CONCLUSIONS.....	47
REFERENCES	48

APPENDIXES	51
Appendix A - The Traffic Demand computation in Matrix Form	51
Appendix B – MATLAB Program	52
Appendix C – C++ Program	54

LIST OF FIGURES

Figure	Page
Figure 2-1 Restoration or protection methods.....	4
Figure 2-2 SONET APS protocol [4].	7
Figure 2-3 types of APS [3].....	9
Figure 2-4 U-SHR operation normal and failure scenarios.....	11
Figure 2-5 B-SHR/2 operation normal and failure scenarios.....	12
Figure 2-6 Line restoration vs. Path restoration [5].....	15
Figure 3-1 Preplanned line restorations processing in OXCs.....	19
Figure 3-2 Preplanned line restoration procedures.....	23
Figure 3-3 Flow chart for LP [20]	29
Figure 3-4 Example of the EST algorithm.	30
Figure 3-5 example of PST algorithm.....	31
Figure 3-6 Flow chart of EST and PST	33
Figure 3-7. Restoration Procedure.....	35
Figure 3-8 Requirement stages for preplanned restoration	36
Figure 4-1 An example of a backbone optical network connection.	43
Figure 4-2. The total spare capacity using ETS and PTS algorithm in the actual network.	44
Figure 4-3 Restoration time.....	44
Figure 4-4 the average number of control packets.	45
Figure 4-5 Test network Ref. [20]......	45
Figure 4-6 Total spare capacity between LP, EST, and PST algorithm.	46

LIST OF TABLES

Table	Page
Table 3-1 WFT and RFT at each OXC.	20
Table 3-2 Optical switching time [6].....	37
Table 4-1 populations of the cities	43

CHAPTER I

1. INTRODUCTION

Internet applications such as video, audio and data transmission require extensive bandwidth in present. Therefore, optical networking has become an essential transportation technology due to its reliability and capacity of supporting support high data rate services. The Synchronous Optical Network (SONET) and Synchronous Digital Hierarchy (SDH) is a commonly used standard in optical networks. Meanwhile, the pure Optical Transport Network (OTN) technology applying wavelength division multiplexing (WDM) is rapidly emerging as a new industrial standard for the future optical networks, which is capable of increasing the network capacity. The idea behind this technique is to multiplex several wavelengths with giga-bit data rates into a higher transmission rate up to tera-bits through a single fiber link [1], [2], [6], [18]. The optical network can be divided into three layers, namely the control layer, optical transport layer and WDM layer. The control layer controls and manages the optical signals transmitted in the optical transport layer through the Digital Cross connect System (DCS) and Optical Cross Connector (OXC). The functionalities of the controller include monitoring, network management, and fault management. The control mechanism can be either centralized or distributed [4]. In the optical transport layer, OXCs have a simple switching function and can effectively establish optical channels from sources to destinations. However, in this layer potential channel failures may be caused by weak optical transmitter power, receiver problems, or especially a link cut. Both SONET/SDH and OTN networks multiplex numerous optical channels into a single optical fiber using wavelength-

multiplexing technology, and therefore, a single fiber link failure in a WDM layer will result in a significant loss in terms of disconnecting these optical channels. Due to this reason, the survivability of the optical network is one of the most important issues in network reliability. In existing optical ring networks, the Self-Healing Ring (SHR) architecture is most commonly used to deal with optical network failures, because of its simple protection procedure and relatively fast recovery time (commonly less than 50 ms) [4]. However, SHR is not efficient in bandwidth allocation and cannot deal with multi-link failures within a ring network.

There are two main considerations in the restoration of optical networks. One is the fast recovery of network failure and the other is the cost efficient network design including reserved spare capacity. The main advantages of a mesh network topology are its capacities of dealing with multi-link failures and efficiently use of network resources. But it requires a more complex recovery mechanism than SHR architecture. Therefore, reducing restoration time in optical mesh networks has become one of the most important areas in networking research.

The purpose of this thesis is to study several restoration techniques in optical networks, and to figure out the best recovery method in terms of restoration time and backup capacity. This method needs to assign the spare capacity over the network. In order to efficiently perform Spare Capacity Assignment (SCA), this thesis proposes two new sub-optimal methods to find efficient SCA, which are Equally Shared Traffic (EST) and Proportionally Shared Traffic (PST) algorithms. We refer call these as Reduced Spare Capacity (RSC) algorithms throughout this thesis. Since the Linear Program (LP) [13], [20] algorithm is one of the most important approaches for Optimized Spare

Capacity (OSC) reservation. The comparison of the proposed RSC algorithm with LP algorithm will be presented in Chapter III. There are two methods to send control packets (i.e., unidirectional and bidirectional methods) for the failure recovery. The benefits of RSC algorithms are shown to be obvious when efficiently performing SCA in the bidirectional case.

The following parts of this thesis are organized as follows. Chapter 2 introduces different protection and restoration techniques for optical network failures and chapter 3 presents new proposed RSC algorithms and compares it with the LP algorithm. Simulation results are reported in chapter 4. Chapter 5 concludes this thesis.

CHAPTER II

2. REVIEW OF RESTORATION TECHNOLOGIES

In this chapter, a summary of restoration technologies are presented based on the references [3], [4], [5], [16]. Fast restoration in optical networks is a critical issue because of each links extraordinary data transmission capability. In general, non-hierarchical restoration can be divided into two categories, i.e., dynamic and preplanned. These two classes of restoration will be discussed in detail in this chapter. Figure 2-1 shows a classification of techniques used in the existing optical networks.

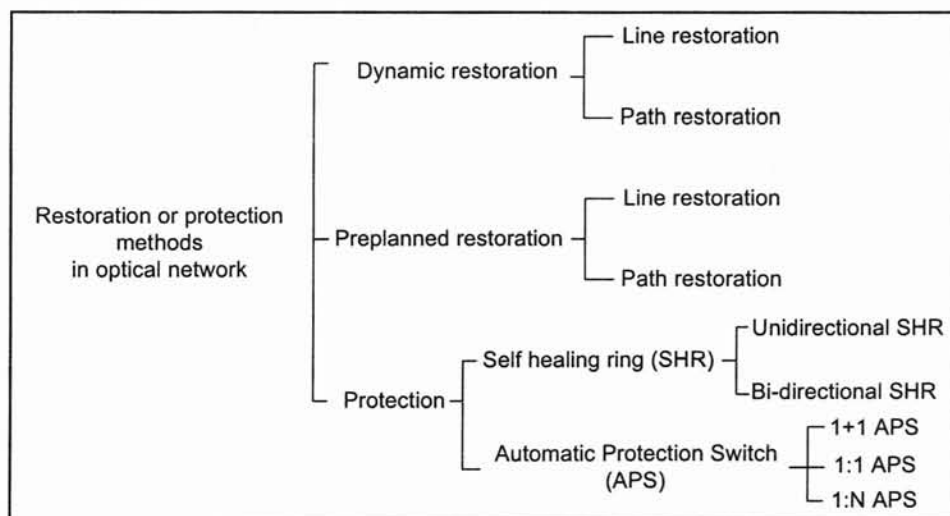


Figure 2-1 Restoration or protection methods.

2.1 Differences between Restoration and Protection

The main purpose of restoration and protection is to recover any type of network failure especially link failures. The differences between restoration and protection are the

way to find restoration paths and reserve spare capacity for network failures. The protection scheme uses an Automatic Protection Switching (APS) protocol. Typically, APS can handle an optical link failure, which occurs due to a single fiber or some single fibers cut in an optical network, which results in a break of optical channels or result in weak optical or no signal power arrival in WDM networks. Originally, there are three kinds of APS method, which are 1+1, 1:1, and 1:N APS. Usually, these APS schemes are used in ring architecture but can be also used in mesh networks as well. The difference between the three schemes is the amount of protection resources. The protection technique uses simple switching functions to reroute automatically from normal working mode to the protection mode directly after a link failure is detected. These APS topologies are explained below.

The other hand, restoration techniques are more complicated techniques, because restoration techniques need algorithms to find alternative paths and reserve backup capacity for the restoration routes in case of link failures. This method need more recover time than protection scheme, but the restoration method is more reliable and is an efficient to deal with network failures. The restoration methods were originally developed for mesh network architectures.

2.2 SONET APS Protocol

SONET network failure can be recovered by an APS protocol. For using APS protocol, physical layer must prepare dedicated protection link (i.e. 1:1, 1+1, and M:N), which is an empty channel in normal situation (non-failure state).

The SONET APS protocol uses in-band signaling for protection switching through K1 and K2 bytes within the SONET line overhead. The K1 byte requests a channel for the switch action; the K2 byte confirms that the channel is bridged onto the protection line.

The SONET 1:1 APS protocol uses a three-phase protocol for bi-directional protection switching operation, which is summarized as follows. When a failure is detected or a switch command is received at the receiving node, the protection logic compares the priority of this new condition with the request priority of the working channel that requests the use of the protection channel. This comparison includes the priority of any bridge order. If the new request is of higher priority, the K1 byte is loaded with the request and the ID number of the channel requesting the use of the protection line. The receiving node then sends out the K1 byte on the protection line. Figure 2-2 depicts the APS protocol.

When this new K1 byte has been verified and evaluated by the priority order at the transmitting node, the K1 byte is sent back to the receiving node with a reverse request to confirm the channel requesting the use of the protection channel. A bridge is also ordered at the receiving node for that channel. This action initiates a bi-directional switch. At the transmitting node, the indicated channel is bridged to protection. When the channel is switched, the K2 byte is set to indicate the number of the channel being protected.

At the receiving node, when the channel number on the received K2 byte matches the number of the channel requesting the switch, that channel is selected for protection. This completes the switching to the protection channel for one direction. The receiving

node also performs the bridges, as ordered by the K1 byte and indicates the bridged channel on the K2 byte. The transmitting node completes the bi-directional switch by selecting the channel from protection when it receives a matching K2 byte [4], [5].

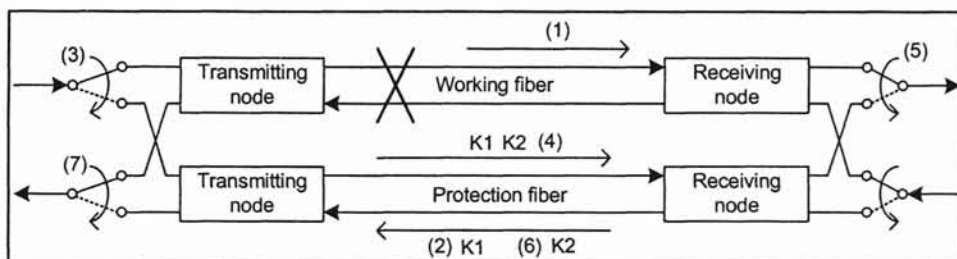


Figure 2-2 SONET APS protocol [4].

2.2.1 Automatic Protection Switching (APS) Architectures

This chapter briefly explains the different kinds of APS architecture in Figures 2-3. The three different types of APS architectures are explained in the following subsections [4].

2.2.1.1 1+1 APS Architecture

In normal situation, the 1+1 APS architecture, source node transmits the information signal to destination node through both working and protection link, which have the same volume of capacity. The receiving node monitors received signal quality on both working and protection link, and the selector selects signals from protection line if the received signals from working link are not qualified. It is a simple method in that

the nodes do not need any switching function. The recovery time need not be considered in 1+1 APS architecture.

2.2.1.2 1:1 APS Architecture

The 1:1 APS uses switch function in end terminal instead of using signal selector such as 1+1 APS. The 1:1 APS is normally works with working link and deal protection, but the failure detected of the end node switch to change path from working to protection link after failure is occurred. This protection must consider switching time as the failure protection time. 1:1 APS and 1+1 APS require 100 % redundancy protection link capacity necessary. Under normal conditions, the protection link is either ideal or used to carry low-priority traffic [3].

2.2.1.3 1:N APS Architecture

In the 1:N architecture, N working fiber links share one protection link, where working and protection fiber systems are placed in the same physical route. The reason is that, only one of the working fiber links can be protected by one protection link. The limitation of that can be approved by using M:N, which has M protection links and N working links. Basically, The 1:N comes from 1:1 APS. The 1:N APS system includes protection switching modules, a 1:N bus, and a Protection-Switching Controller (PSC) [4].

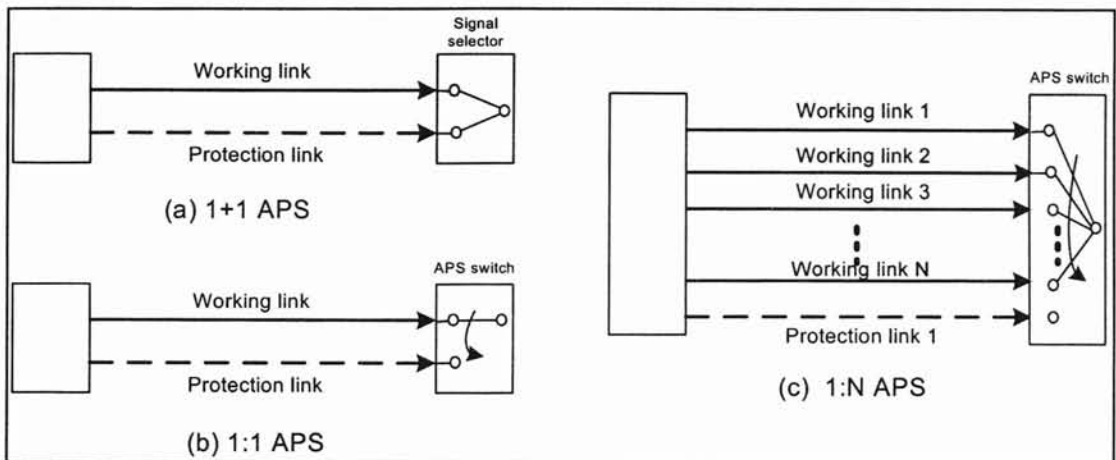


Figure 2-3 types of APS [3]

2.3 SONET Self-Healing Ring (SHR) Network

A SHR consists of Add Drop Multiplexes (ADMs), and it guards against fiber cuts without using the more expensive technique of 1:1 APS and 1+1 APS with diverse routing. It also reduces the number of nodes required for the same traffic by allowing fiber capacity to be shared. The self-healing ring can generally be divided into two categories: bi-directional SHR (B-SHR) and unidirectional SHR (U-SHR). Application roles between U-SHR and B-SHR are suggested that a U-SHR may be more economically attractive in areas where demands are homed to one central location, but B-SHR may be attractive in area where demands are more uniformly distributed. Thus, the U-SHR may be more appropriate in feeder networks and peripheral networks, and the B-SHR may be more economically attractive in interoffice networks with a nonhubbing structure. In a realistic network environment, a SONET network may use different types of survivable network architecture due to economics and demand distribution [5].

The type of ring depends upon the path traveled by a duplex communication channel between each office pairs. The SHR is called a bi-directional SHR (B-SHR) if both directions of a duplex channel travel over the same path; a unidirectional SHR (U-SHR) if the direction of a duplex channel travels the opposite paths.

2.3.1 Unidirectional SHR (U-SHR)

In a unidirectional SHR (U-SHR), two fibers are needed between adjacent nodes: one for working and the other for protection. Each ring node is equipped with one ADM, which adds/drops local channels and passes through transit channels. Figure 2-4 shows an example of U-SHR operation under the normal and failure scenarios. The U-SHR can be implemented according to the concept of 1+1 or 1:1 protection. The 1+1 U-SHR splits the signals onto both the working and protection ring at the transmitting node, and the receiving node selects the best of two identical receiving signal based on protection switching criteria. In contrast, The 1:1 U-SHR uses a separate ring as the protection ring, which does not carry the service demand in the normal situation, and loops the disrupted channels onto the protection ring from the working ring when the network occurs failures [4]. In Figure 2-4, the detail of an example of 1:1 U-SHR explains as follows.

The U-SHR can protect any failure using 1:1 APS and 1+1 APS. Traffic from node A to node C is routed through node B (i.e. path A-B-C), and traffic from node C to node A is traveled through node D (i.e. path C-D-A). Therefore, traffic arrives at node A and C by different paths. Because the transmission of normal working traffic on the U-SHR is in only one direction. U-SHRs are sometimes called “counter-rotating rings” because the second communication ring (for protection only) transmits in the opposite

direction of the working ring [5]. In the failure scenario as refer to Figure 2-4 (b), when fiber link between node A and D is cut or malfunction, node A detects an alarm signal or poor Bit Error Rate (BER), then node A and D switch over from normal working path to protection fiber link.

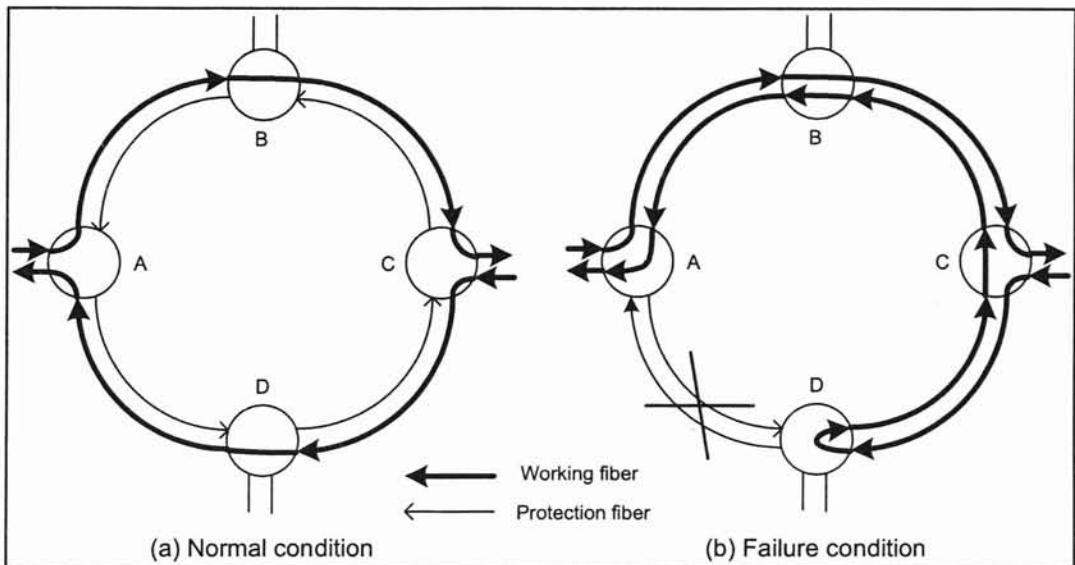


Figure 2-4 U-SHR operation normal and failure scenarios

2.3.2 Bi-directional SHR (B-SHR)

There are two types of bi-directional SHR (B-SHR), which are two fibers B-SHR (B-SHR/2) and four fibers B-SHR (B-SHR/4). For a B-SHR/4, two fibers are used to carry normal services and the other two fibers are used for protection. Unlike the B-SHR/4, working and protection traffic for B-SHR/2 is routed on the same fiber with a portion of bandwidth reserved for protection. 1:1 APS uses complicated on B-SHR/2, because exactly half of the fiber bandwidth is reserved for the protection [5]. In a B-SHR/2, as refer to Figure 2-5, working traffic transmits in both directions over a single

path that uses the two parallel working paths using 1:1 APS. Note that protection fibers or capacities in B-SHR are allowed to carry lower-priority traffic that will be dropped during the network restoration process. In Figure 2-5 (a), normal working traffic between node A and node C is routed the same path with opposite direction. In Figure 2-5 (b), when failure occurs between node A and node D, which detect failure, and then traffic from node A to node C changes path A-B-C instead of A-D-C, and the other traffic from node C to node A converts path from C-D-A to C-D-C-B-A [5].

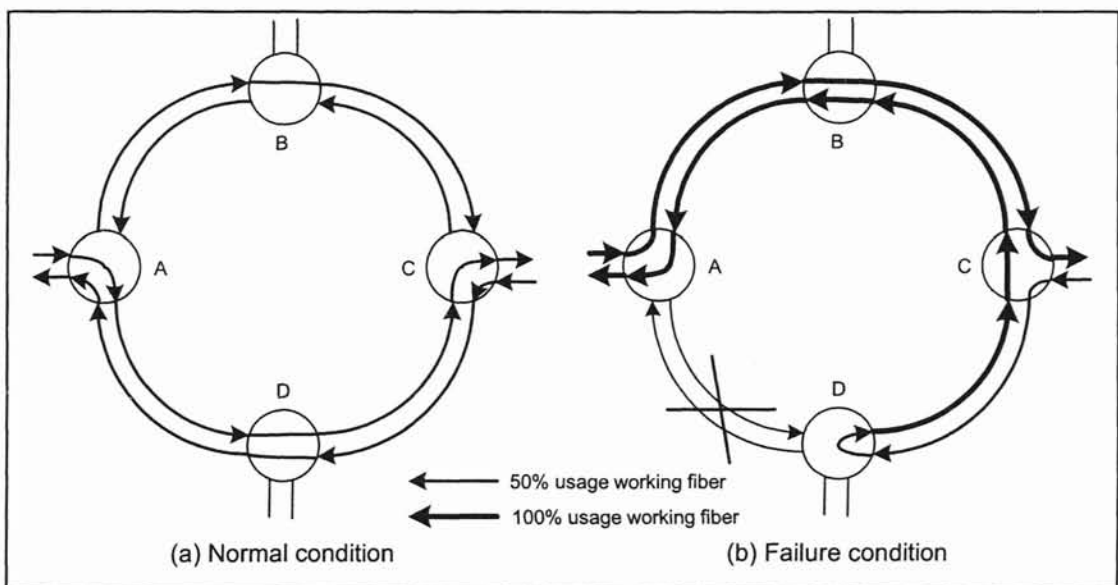


Figure 2-5 B-SHR/2 operation normal and failure scenarios

2.4 Restoration Methods in Mesh Network

The meaning of restoration is different from protection. The protection mechanism uses switch function to convert traffic path from the original working line to

the protection line, which are in the same fiber cable (i.e. the same route) or in the reverse path. On the other hand, restoration mechanism recovers a failure to reroute the working traffic through alternative paths. Normally, protection method is used in a ring network and restoration method is implemented in mesh architecture.

There are two kinds of restoration mechanism can be implement in mesh architecture. One of them is dynamic restoration, which implies the discovery of spare capacity dynamically in the network to restore the affected services; that is, the resources used for recovery are not reserved at the time of connection establishment, but are chosen from available resources. This is more efficient than preplanned restoration from the viewpoint of resource utilization. However, the disadvantage of dynamic restoration is more restoration time needed and has complexity to control. Sometimes, 100 percent recovery is not guaranteed because, sufficient spare capacity may not be available at the time of failure [3].

The other method is preplanned restoration. When the network designer designs network with using preplanned method, there is the requirement of redundancy spare capacity assignment in the network for any possible network failure considered. This method is not efficient compare with dynamic restoration but it has fast recovery capability for the failure. Because, it has already defined rerouting table before the failure happened.

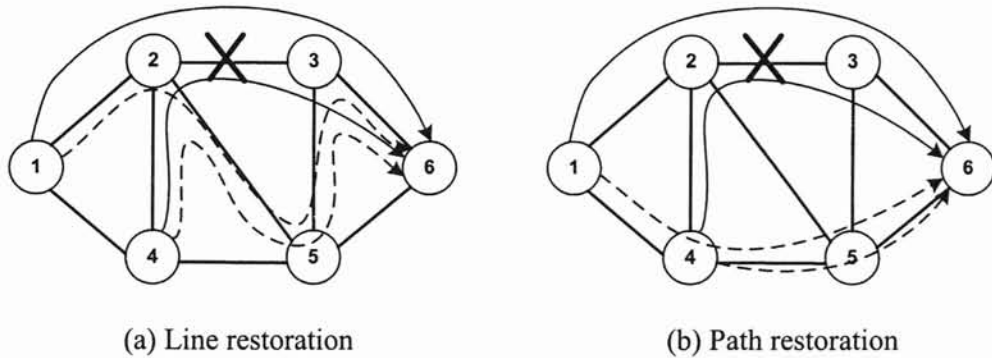
In optical network, which uses DWDM technology or not, still fast restoration time must be required. Due to the large traffic transmitted in optical fiber.

2.4.1 Line Restoration vs. Path Restoration

There are two restoration techniques namely, line restoration and path restoration. These techniques can be used either by line layer or by path layer. Line restoration uses the line layer information to trigger the restoration process and restores all affected paths in the affected facility regardless of the sources and destinations of these affected paths. In contrast, path restoration restores affected STS paths on an end-to-end basis [5]. Figure 2-6 (a) depicts Line restoration, which is perhaps simple to implement in the optical network. A line restoration routes the failed optical path across an alternate link between sites, providing protection in case of a fiber or equipment failure. Although providers can dedicate fibers for a protection link, it is usually not cost effective to do so. Therefore, shared risk fiber can reduce the cost of capacity. Line restoration can reduce restoration time more than path restoration, but line restoration may require more redundancy spare capacity. From the point of view optical network, the fast restoration time is the most important parameter, so line restoration is preferred to use.

Path restoration is more complex to control and more restoration time is needed, because when the node detects one of the link failures, the node broadcast failure message to the other node gathering information about restoration paths, which has enough bandwidth for specific failure link. Each node has its source traffic to find alternative route for avoiding the failure link as shown Figure 2-6 (b). Path restoration requires time for the all node of the recognition time, finding alternative path time, and configuration time, etc. Sometimes, path restoration is considered as the most economical restoration technique when it is working with dynamic method. Because of its complexity

and slower restoration time, path restoration is not likely to be considered in the optical network.



(Source, Destination)	Original path	Line restoration path	Path restoration path
(1,6)	1-2-3-6	1-2-5-3-6	1-4-5-6
(4,6)	4-2-3-6	4-2-5-3-6	4-5-6

Figure 2-6 Line restoration vs. Path restoration [5]

2.4.2 Centralized vs. Distributed Control

In the centralized restoration, a node detects a failure and sends failure information to the central controller, and then the central controller computes the rerouting path based on network topology and resource data. Centralized control schemes can be implemented in ATM, SONET/SDH, as well as WDM layer

In the distributed control environment, the alternative route is computed at the local node. These functions are performed to exchange the restoration message in a distributed manner.

There are several disadvantages of the centralized control scheme, which are higher administration overhead, higher system vulnerability, and slower restoration speed (on the order of minutes or longer). On the other hand, the distributed control scheme has faster restoration time, lower system vulnerability, and lower administration costs. However, it does so at the expense of system unpredictability, a complex control system, greater standardization efforts, and, possibly, less efficient use of spare capacity [4], [16].

CHAPTER III

3. ANALYSIS OF RESTORATION ALGORITHMS

The first section of this chapter shows the preplanned line restoration process of WDM networks in the context of the wavelength non-blocking case. Since WDM is conceptually similar to frequency division multiplexing (FDM), if there are some specific wavelength channels used in a part of the switching path, then the original working wavelength must be converted to a different wavelength to avoid wavelength blocking. The second section explains the details of the preplanned line restoration processes and time. The second section discusses about the LP algorithm. The last section presents the proposed RSC algorithms and compares them with the performance of the LP algorithm in terms of restoration time and average number of control packets.

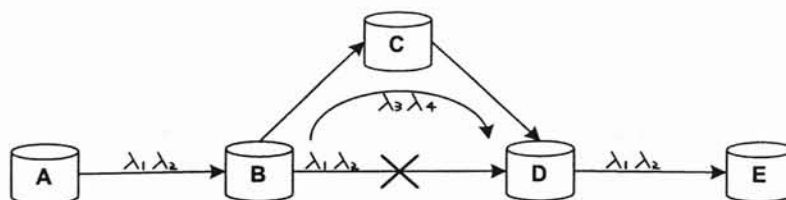
3.1 Considerations of Preplanned Line Restoration in WDM network

An optical network consists of three layers namely the control layer, the optical transport layer, and the WDM layer. Optical multiplexing technology is used to transmit bundles of optical signals at different wavelengths through the WDM layer. Failure of a WDM network means that the multiplexed group of optical channels is lost at a given time. Thus, fast restoration time is very important in WDM networks. As explained before, there are two main kinds of restoration mechanisms, dynamic restoration and preplanned restoration. WDM networks prefer preplanned restoration methods due to the fast restoration time. However, due to possible wavelength blocking problems when the

original wavelength channels need to switch over to a new restoration path, wavelength recalculations will be required. Recently wavelength converters are under development developed to solve the wavelength-blocking problem.

This section explains how preplanned line restoration processes can assist to recover link failure in WDM mesh networks without blocking problems. The details are shown in Figure 3-1 and Table 3-1. When OXC-D detects a failure, the original wavelengths through path OXC-B to OXC-D will be routed through OXC-B → OXC-C → OXC-D. This eligible alternative path is pre-calculated in OXC-B's restoration-forwarding table (RFT) in case of the link failure. OXC-C and B will immediately detect the link failure. So, OXC-D needs to notify this link failure to the other OXCs, which are included within the pre-calculated rerouting path (D-C-B). When OXC-C and OXC-B receive a Failure Message (FM), they change the forwarding table immediately from the Working Forwarding Table (WFT) to the Restoration Forwarding Table (RFT). The details of WFT and RFT show in Table 3-1.

In addition, in figure 1 (b), the OXC-C has been using wavelengths λ_1 and λ_2 for the data traffic. In addition, OXC-B and OXC-D may need to convert the original wavelength λ_1 to λ_3 , and also convert λ_2 to λ_4 through the alternate route to avoid blocking problems in WDM network.



(a) Wavelength channel routing

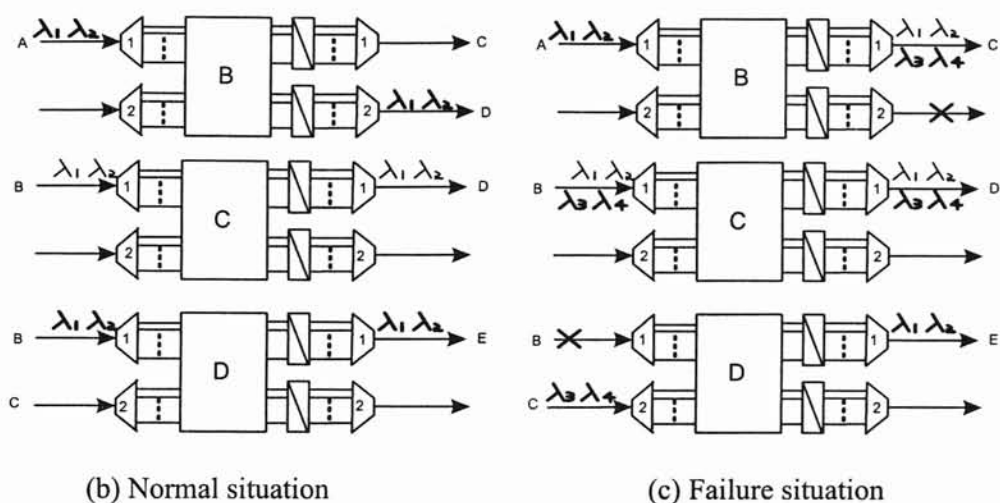


Figure 3-1 Preplanned line restorations processing in OXCs

Input WDM ID #	Input Wavelength # (Status)	Convert status	Output Wavelength # (Status)	Output WDM ID #
N/A	λ	Non	$\lambda 1(\text{using})$	1
N/A	λ	Non	$\lambda 2(\text{using})$	1
1	$\lambda 1(\text{using})$	Non	$\lambda 1(\text{using})$	2
1	$\lambda 2(\text{using})$	Non	$\lambda 2(\text{using})$	2

(a) WFT at OXC B

Input WDM ID #	Input Wavelength # (Status)	Convert status	Output Wavelength # (Status)	Output WDM ID #
N/A	λ	Non	$\lambda 1(\text{using})$	1
N/A	λ	Non	$\lambda 2(\text{using})$	1
1	$\lambda 1(\text{using})$	$\lambda 1 \rightarrow \lambda 3$	$\lambda 3(\text{ready to use})$	1
1	$\lambda 2(\text{using})$	$\lambda 2 \rightarrow \lambda 4$	$\lambda 4(\text{ready to use})$	1

(b) REF at OXC B for the link failure situation between OXC B and OXC D

Input WDM ID #	Input Wavelength (#) (Status)	Convert status	Output Wavelength # (Status)	Output WDM ID #
1	$\lambda 1(\text{using})$	Non	$\lambda 1(\text{using})$	1
1	$\lambda 2(\text{using})$	Non	$\lambda 2(\text{using})$	1
N/A	λ	Non	λ	N/A
N/A	λ	Non	λ	N/A

(c) WFT at OXC C

Input WDM ID #	Input Wavelength # (Status)	Convert status	Output Wavelength # (Status)	Output WDM ID #
1	$\lambda 1$ (using)	Non	$\lambda 1$ (using)	1
1	$\lambda 2$ (using)	Non	$\lambda 2$ (using)	1
1	$\lambda 3$ (ready to use)	Non	$\lambda 3$ (ready to use)	1
1	$\lambda 4$ (ready to use)	Non	$\lambda 4$ (ready to use)	1

(d) REF at OXC C for the link failure situation between OXC B and OXC D

Input WDM ID #	Input Wavelength # (Status)	Convert status	Output Wavelength # (Status)	Output WDM ID #
1	$\lambda 1$ (using)	Non	$\lambda 1$ (using)	1
1	$\lambda 2$ (using)	Non	$\lambda 2$ (using)	1
N/A	λ	Non	λ	N/A
N/A	λ	Non	λ	N/A

(e) WFT at OXC D

Input WDM ID #	Input Wavelength # (Status)	Convert status	Output Wavelength # (Status)	Output WDM ID #
1	λ	Non	λ	1
1	λ	Non	λ	1
1	$\lambda 3$ (ready to use)	$\lambda 3 \rightarrow \lambda 1$	$\lambda 1$ (ready to use)	1
1	$\lambda 4$ (ready to use)	$\lambda 4 \rightarrow \lambda 2$	$\lambda 2$ (ready to use)	1

(f) REF at OXC D for the link failure situation between OXC B and OXC D

Table 3-1 WFT and RFT at each OXC.

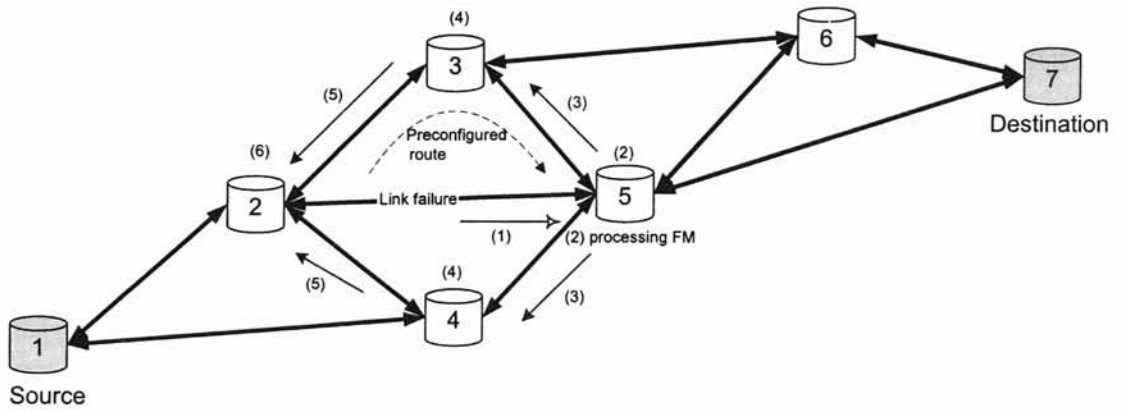
3.1.1 Preplanned Line Restoration Procedure

The preplanned restoration method does not require additional time to search for the eligible alternative paths and therefore the restoration time can be reduced.

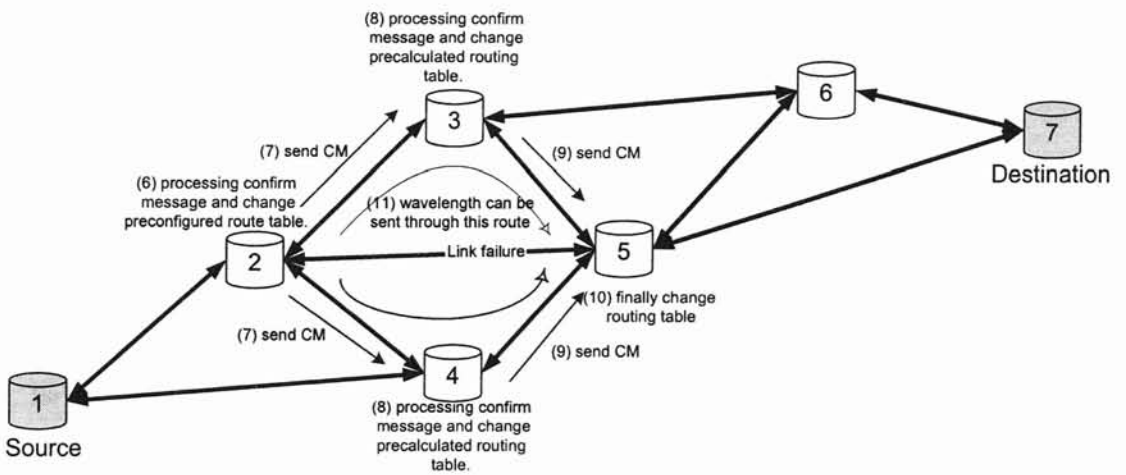
Preplanned restoration systems need to set up pre-calculated restoration paths and resource backup capacity at every node in case of adjacent link failures. These pre-calculated recovery paths are stored in the OXC forwarding table (i.e. RFT). Eligible recovery paths are measured based on the limited hops count (i.e. $h = 2, 3, 4, 5 \dots$) [13], [20], which gives the hop count limit for restoration part of the alternate routes. If the limited hop count increases, the preplanned line restoration time will increase, but the pre-allocated spare channel capacity can be reduced using efficient SCA algorithms. Preplanned line restoration means that nodes have their own RFT, in which restoration paths are pre-calculated for the case of adjacent link failures. The preplanned line restoration procedure is explained step by step below and is shown in Figure 3-2:

1. If bi-directional link failure occurs between OXC-2 and OXC-5, these two OXCs can detect the link failure by Loss Of Frames (LOF) at each OXC. LOF is commonly defined as the state where no signal has been received for $625\mu s$ or more. In this step-by-step procedure, we show only one direction in which the loss of the data link (wavelength λ) is detected by OXC-5.
2. OXC-5 can be the sender node for the received optical signal after link failure. OXC-5 processes the FM and prepares itself to reconfigure its working table based on the pre-calculated table. FM contains the following data.
 - a. Failure link ID,
 - b. Sender OXC-ID and chooser OXC-ID
 - c. Information of request channels for the specific restoration paths.
 - d. Update information of intermediate node IDs.
 - e. Line restoration command message

- f. Other data. (Network topology, priority of the channels...)
3. OXC-5 sends the FM to the adjacent nodes, which are in the pre-calculated recovery paths.
4. OXC-3 and 4, after they receives the FM from OXC-5, process the FM and prepare themselves to reconfigure to their working table based on the pre-calculated table.
5. OXC-3 and 4 send FM to the OXC-2, which receives the FM from OXC-3 and 4 at nearly the same time.
6. At this stage, OXC-2 knows the specific link failure and has information about the optical channels that are not being used. OXC-2 processes confirm message (CM) and changes the WFT to the RFT. The CM contains the following data.
 - a. Chooser node ID
 - b. Acknowledgement of reserved channels
 - c. Update information of intermediate node ID
 - d. Other functionalities.
7. OXC-2 sends the CM to OXC-3 and OXC-4.
8. OXC-3 and OXC-4 reconfigure their RFT after getting CM from OXC-2.
9. OXC-3 and OXC-4 send the CM to OXC-2.
10. OXC-2 finally reconfigures to it's preplanned RFT.
11. The failure wavelength channels can go through both OXC-5 → 3 → 2 and OXC-5 → 4 → 2.



(a) Preplanned link restoration procedure 1st step



(b) Preplanned link restoration procedure 2nd step

Figure 3-2 Preplanned line restoration procedures

3.1.2 Preplanned Line Restoration Time

The preplanned line restoration time can be calculated using the following equation (4.1). The equation is derived from the preplanned line restoration procedures.

The equation consists of three variables namely the failure detection time, the propagation delay, and optical switching configuration time.

$$T_{line} = F + 2h \cdot P_{i,j} + 2h \cdot M + (h + 1) \cdot C \quad (4.1)$$

- T_{line} : The preplanned line restoration time.
- $P_{i,j}$: Propagation delay between node i and j (ms).

$$P_{i,j} = \frac{D_{i,j} (km)}{0.6 \times 3 \times 10^5 (km/sec)}$$

- $D_{i,j}$: Link distance between node i and j (km).
- M : Message processing time at the OXC.
- C : Time to configure the optical cross-connection routing table.
- p : The number of alternative paths between the end nodes of the failure link.
- h^p : The number of hop counts in the p^{th} alternative path.
- h : The maximum number of hops in the backup route between the ends of the failure link, (i.e., $h = \max \{h^p\}$).

3.2 Efficiency Spare Capacity Assignment Algorithms

Spare capacity assignment (SCA) is an important part of a fault tolerant network design. The optimized spare capacity (OSC) assignment on a mesh network has been investigated in [8], [13], [19]. Most of these papers consider OSC solutions for the single link failure. The main papers [13] and [20] have already presented OSC base solutions using LP algorithm, which uses Min-Max functions. With respect to the optimized

solution, LP gives the most efficient SCA. Basically, the LP algorithm sets the minimized initial value of restoration flow ($f_{i,n}$) and then finds out the optimized spare capacity (S_j) by reducing the reducible value S_j one unit at a time until the non-reducible value is found. The above algorithm will be explained in detail in following subsection. When the OSC solution assigns redundant spare capacity to an optical network, the network can use preplanned line restoration with unidirectional procedures. Due to traffic fluctuation, unidirectional procedures require preplanned line restoration with OSC assignments. There are two ways of performing bidirectional procedures with OSC assignments. One is to calculate traffic assignment of the restoration paths from the maximum working traffic of each link. The fixed RFT is set up on the nodes, but this is an inefficient way to restore link failure, because of the fluctuation of working traffic. The other method is to update the nodes' RFT frequently by using MPLS as the control plane in the optical network.

The thesis proposes two simple algorithms. One is the Equally-Shared Traffic (EST) algorithm and the other is the Proportionally-Shared Traffic (PST) algorithm. These algorithms are simple to obtain efficient spare capacity for preplanned line restoration. The reserved spare capacity solutions of these algorithms are not optimal as the LP algorithm. Therefore, in this thesis these solutions are called Reduced Spare Capacity (RSC). However, the EST and PST algorithms are simpler than the LP algorithm. Also, preplanned line restoration can use bidirectional procedures when EST and PST solution assigns redundant spare capacity to an optical network. Efficient Spare Capacity Assignment (SCA) algorithms such as LP, EST, and PST algorithm can obtain

different solutions depending upon the network connection and traffic allocation. These algorithms are explained in detail in the following subsection.

3.3.1 Linear Program (LP) with Min-Max Function [13], [20]

The LP iterative procedure is aimed at minimizing the total spare capacity. It is based on finding reduced cut sets for each link failure. The following notation was used to formulate the mathematical model in this section:

h : The limited hop count for the restoration path.

$h_{i,n}$: The number of hop counts in the n^{th} restoration path for the failure line i ($i = 1, 2, 3, \dots, L$).

h_i^T : The sum of hop counts in all restoration paths for the failure line i .

τ : This variable is in inverse proportion to the number of hop count ($h_{i,n}$).

L : The number of lines in the network, indexed ($i = 1, 2, 3, \dots, L$).

T : The total number of assigned channel capacity in the network.

W : The total number of working channel capacity.

S : The total number of spare channel capacity.

W_i : The working channel capacity in the line i .

S_j : The spare channel capacity in the line j .

S_i^j : The total spare channel capacity on the line j for the failure line i .

P_i : Total number of eligible restoration paths for failure link i ($n = 1, 2, 3, \dots, P_i$).

$f_{i,n}$: The restoration flow through the n^{th} restoration path for the failure line i

($n = 1, 2, 3, \dots, P_i$).

F_i^j : Total “restoration flow” through link j .

q_i : The minimal restoration level ($0 \leq q_i \leq 1$).

$\alpha_{i,n}^j$: The value is 1 if the line j is used within n^{th} restoration path for the failure line i .

From the above notation, one can formulate the SCA task in the following form.

$$M_1: \quad \text{Min} \left\{ \sum_{j=1}^L C_j \cdot \max_{i \neq j} \{F_i^j\} \right\} \quad (4.2)$$

$$\sum_{n=1}^{P_i} f_{i,n} \geq q_i \cdot W_i \quad 0 \leq q_i \leq 1, \quad \text{Usually set } q_i = 1, \forall i \quad (4.3)$$

$$F_i^j = \sum_{n=1}^{P_i} \delta_{i,n}^j \cdot f_{i,n} \quad i, j = 1, 2, 3 \dots L, \quad i \neq j \quad (4.5)$$

The equation (4.3) requires ensuring the restoration level, which is normally set to one.

The flow F_i^j represents the expected amount of spare capacity through link j . The above

Min-Max model can be represented as a LP formulation in the following way:

$$Z_j - \sum_{n=1}^{P_i} \delta_{i,n}^j \cdot f_{i,n} \geq 0 \quad (4.6)$$

$$M_2: \quad \text{Min} \left\{ \sum_{j=1}^L C_j \cdot Z_j \right\} \quad (4.7)$$

It is worth nothing that some plausible initial values can be added to the model M_2 . The next two models, M_3 and $M_4(l)$ involve finding the OSC solution by reducing value S_j , which is $S_j = \{Z_j\}$ (the lowest integer which is greater or equal to the value Z_j). It is rounded up to a complete integer solution for spare channel assignment. In

Figure 1, the flow chart shows how the M_3 and $M_4(l)$ models works and the equations are represented as given below. Model M_3 finds R_i (a maximum restoration flow value) to associate each link i to one of two sets, s_1 and s_2 .

$$M_3: \quad R_i = \text{Max} \left\{ \sum_{n=1}^{P_i} f_{i,n} \right\} \quad (4.8)$$

$$i \in s_1, \text{ if } R_i - q_i \cdot W_i \geq \sum_{j=1}^L (S_j - Z_j)$$

$$i \in s_2, \text{ if } R_i - q_i \cdot W_i \geq 0$$

The set s_1 represents all links for which their restoration level is ensured through the tightening phase. The set s_2 represents links for which reducing certain spare capacity value S_j might decrease their restoration level. The feasibility of $M_4(l)$ means that the spare capacity of link l can be set to the value $S_l - 1$ and the solution is still feasible.

Model $M_4(l)$ is considered as follows:

$$M_4(l): \quad \text{Max} \left\{ \sum_{i \in S_2} \sum_{n=1}^{P_i} f_{i,n} \right\} \quad (4.9)$$

$$S_{i,j} = S_j \text{ if } l \neq j$$

$$S_{i,j} = S_j - 1 \text{ if } l = j$$

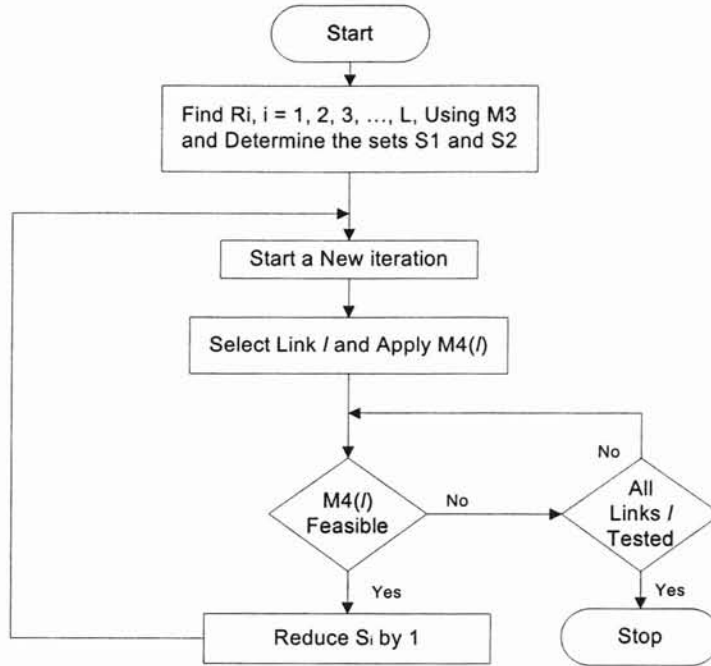


Figure 3-3 Flow chart for LP [20]

3.3.2 The Equally-Share Traffic (EST) Algorithm

The basic idea of the EST algorithm is to distribute the working traffic equally among the eligible restoration paths. As an example, if the value of working traffic (W_i) between node A and node D is 12 as shown in Figure 3-4, it can be divided by 3 the number of the eligible restoration path for failure link i (P_i), to get an integer solution 4. Each restoration flow ($f_{i,n}$) will be assigned a flow amount of 4. The sum of $f_{i,n}$ in a specific link j is the pre-assignment spare capacity (S_i^j) for the link j . By considering every case of single link failure, S_i^j can be calculated. The value of S_i^j is rounded up if it is not an integer. Then the maximum value of S_i^j will be the specific link (j) of spare

capacity (S_j) in the network after all cases of the single link failure are tested. However, if the value of W_i is not a divisible value by P_i , S_i^j must be rounded up for the integer solution.

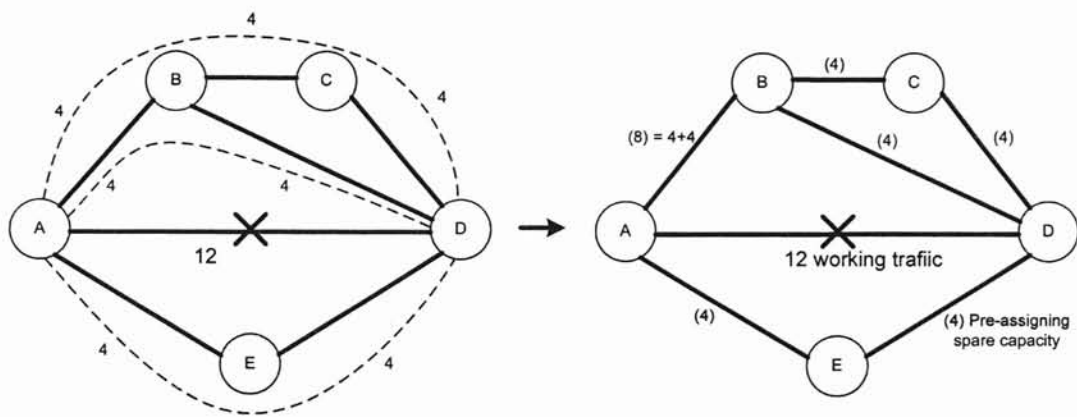


Figure 3-4 Example of the EST algorithm.

The target of all SCA algorithms is to reduce the total spare capacity (S) in the optical network for the design of economic Network Elements (NE).

$$T = \min\{S + W\}$$

$$S = \sum_{j=1}^L S_j, W = \sum_{i=1}^L W_i \quad (4.10)$$

The EST algorithm is expressed by the following equations:

$$f_{i,n} = \frac{W_i}{P_i}, \quad (n = 1, 2, 3, \dots, P_i), \quad i = 1, 2, 3, \dots, L \quad (4.11)$$

$$S_i^j = \sum_{n=1}^{P_i} f_{i,n} \cdot \delta_{i,n}^j, \quad j = 1, 2, 3, \dots, L \quad (i \neq j) \quad (4.12)$$

$$S_j = \left\lceil \max_j \{S_i^j\} \right\rceil \quad (4.13)$$

3.3.3 The Proportionally-Share Traffic (PST) algorithm

The other RSC algorithm is the PST algorithm. This algorithm uses inverse proportional variables (τ, η) where the working traffic (W_i) is divided into a number that is inversely proportional to the number of hop count ($h_{i,n}$) to obtain the value of S_i^j . The value S_i^j must be an integer, and so S_i^j is rounded up if it is not an integer. The maximum value of S_i^j will be the spare capacity for link j (S_j). In other words, some of the alternative paths have a smaller number of hop counts, which can be assigned more spare capacity than the other alternative paths that have larger number of hop counts. For instance, see the Figure 3-5; the working traffic is 16 between node A and D. If the link is cut between node A and D, the working traffic must reroute through three possible paths, which are A-B-D, A-B-C-D, and A-E-D. The alternative paths, A-B-D and A-E-D have hop count of two (i.e., $h = 2$), which means there is a need to assign more spare capacity than the other alternative path, A-B-C-D (i.e., $h = 3$).

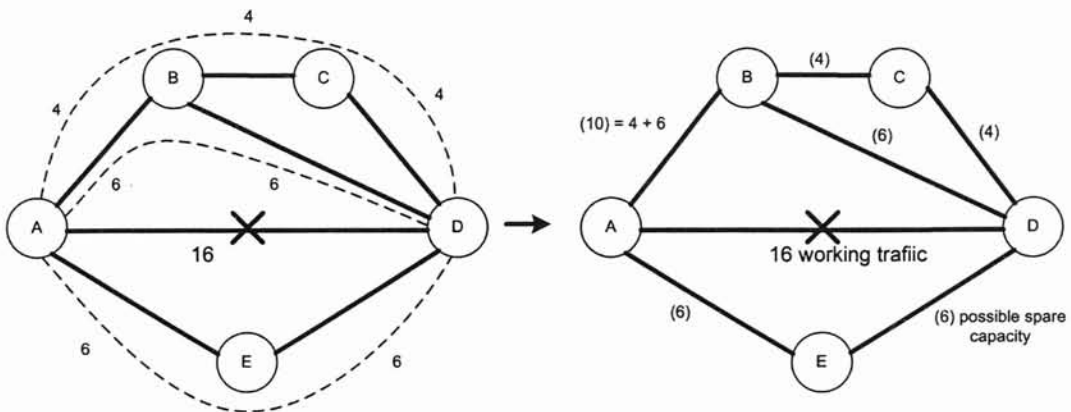


Figure 3-5 example of PST algorithm

The PST algorithm is expressed by the following equations:

τ, η : Inverse proportional variables. There is a flow chart in Figure 3-7 to show how EST and PST compute the spare capacity required (S_j) of link j .

$$h^T = \sum_{n=1}^{P_i} h_{i,n} \quad (4.14)$$

$$\tau = \prod_{n=1}^{P_i} h_{i,n}, \quad \eta = \sum_{n=1}^{P_i} \frac{\tau}{h_{i,n}} \quad (4.15)$$

$$f_{i,n} = \frac{W_i(\tau/h_{i,n})}{\sum_{n=1}^{P_i} \frac{\tau}{h_{i,n}}} = \frac{W_i}{h_{i,n} \cdot \sum_{n=1}^{P_i} \frac{1}{h_{i,n}}} \quad (4.16)$$

$$S_i^j = \sum_{n=1}^{P_i} f_{i,n} \cdot \delta_{i,n}^j, \quad j = 1, 2, 3, \dots, L \quad (i \neq j) \quad (4.17)$$

$$S_j = \left\lceil \max_j \{S_i^j\} \right\rceil, \quad (h \geq 2) \quad (4.18)$$

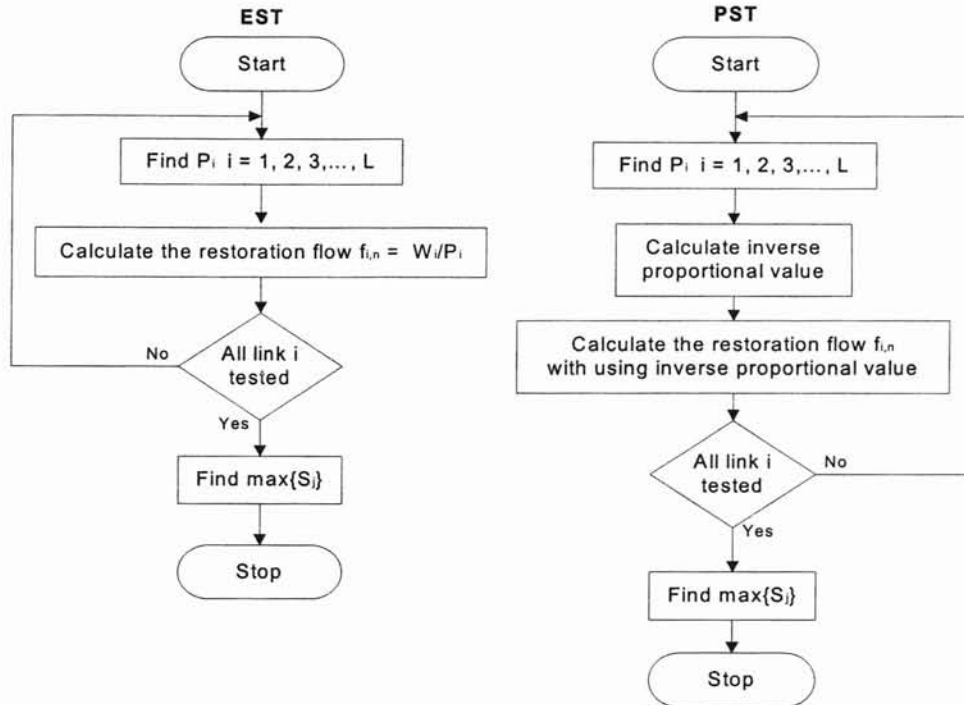


Figure 3-6 Flow chart of EST and PST

3.4 Restoration Time and Control Packets between the OSC and RSC Reservation

Preplanned line restoration means that every node has its pre-calculated restoration paths for the adjacent links. But when the OSC solution is carried out on the network, nodes do not know how much current working traffic can be rerouted through the restoration paths, because the amount of working traffic changes frequently. Therefore, preplanned restoration progress exchanges control packets in a unidirectional method, as shown in Figure 3-8 (b), after the end node detects the failure link. There is a way to use bidirectional procedure OSC solutions. For example, optical networks can use a control plane such as MP λ S or GMPLS, which can be used to update RFT frequently such that all nodes contain updated information about the capacity of each restoration

path when the working traffic changes. However, without MPLS or GMPLS, the OSC solution does not support individual restoration path of traffic assignment. On the other hand, the RSC solution, which is obtained by the EST or PST algorithm, makes each node know automatically the traffic assignment in each restoration path. This is because, the EST solution enables the failure traffic to be distributed equally through the restoration paths, and the PST solution enables failure traffic to be shared proportionally between the restoration paths. With these SCA solutions, FMs can contain the information about the traffic assignment of each restoration path. Therefore, we can use bidirectional method with either EST or PST algorithm.

This section describes two different restoration processes and derives the equations to calculate the restoration time and the average number of control packets. At the second stage of the preplanned restoration, OSC and RSC require different operation procedures, as shown in Figure 3-8. The procedure of exchanging restoration packets is shown in Figure 3-7, and the restoration procedure process is explained step-by-step below.

Bidirectional Procedure (Figure 3-7 (a))

1. Node A and D detect link failure.
2. Node A sends FM to node B, E; Node D sends FM simultaneously to B, C, E.
3. Nodes B and E send a CM back to nodes A and D through each restoration path.

Receiving the information of the same amount capacity requested, nodes B and C send an FM to each other.

4. Finally, node B and C send back a CM to D and A through each restoration path. Afterwards, all eligible restoration routes can begin to work replacing the failure link A-D.

Unidirectional procedures are shown in Figure 3-7 (b). In this case, Node A and D simultaneously send out a FM with the same procedure after they detect link failure. Therefore, we represent only the procedure performed by A.

1. Node A detects link failure.
2. Node A sends a FM to node B and E.
3. Node B sends a FM to C, D; E sends FM to D. The FM contains the updated node information. At that time, nodes B and E offer the information of spare link capacity A-B and A-E to nodes C and D.
4. Node C still needs to send FM to node D.
5. Finally, Node D gathers all the information of the spare link capacity, and then sends back a CM through the restoration routes.

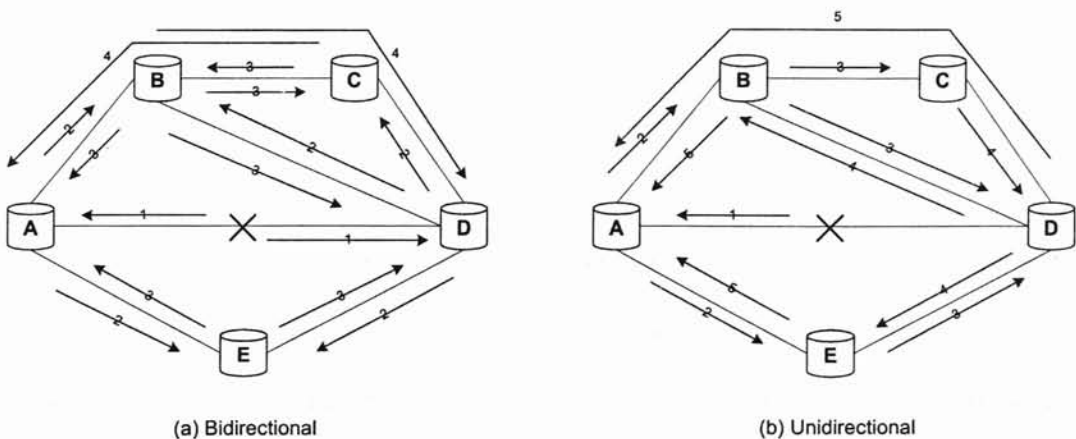


Figure 3-7. Restoration Procedure.

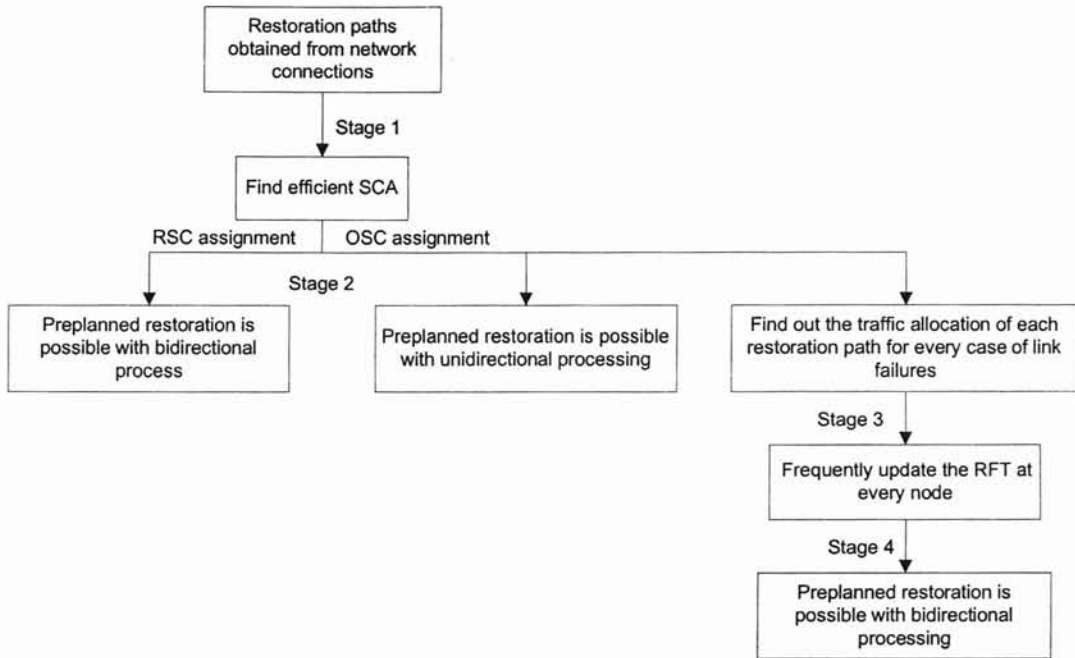


Figure 3-8 Requirement stages for preplanned restoration

Restoration times are usually fast as fewer network elements are involved and fewer control packets are sent and acknowledged during restoration procedure [20]. The above procedures can derive the equations given below for restoration time, which considers several time factors such as failure detection time (F), propagation delay ($P_{i,j}$), message processing time (M), and reconfiguration time (C).

The switching speed varies from nanoseconds to seconds depending on the materials used to make the switch. However, the selection of optical switch does not depend on the switch speed, even though the switch speed is one of the important parameter. There are many considerations to select a switch, such as optical dispersion, loss, reliability, switching/matrix size, temperature dependence, and cost. Some devices

and their switching time are shown in Table 3-2. In the following research, we consider reconfiguration time (C) to be $50 \mu s$ based on the reference optical switching time. The maximum time of LOF is $625 \mu s$ in SONET. Therefore, the assumed failure detection time in this thesis is $625 \mu s$. Finally, the propagation delay time ($P_{i,j}$) between nodes i and j can be obtained from (4.1). Each link has a propagation delay depending on the distance of the optical link. We assume that $P = 3 ms$ as all link distances are the same.

$$\text{Propagation delay time } P = \frac{DISTANCE(m)}{0.6 \times 3 \times 10^8 (m/sec)} \quad (4.19)$$

Optical switch	Switching speed
Thermo-optic switching	Few milliseconds ($\sim 2 ms$)
Acousto-optic switching	Few microseconds (μs)
Electro-optic ceramic component switching	Few microseconds (μs)
MEN switching	Few microseconds (μs)
SiO_2 -on- Si planar	Milliseconds or microseconds
$LiNbO_3$ Switching	nanoseconds (ηs)

Table 3-2 Optical switching time [6]

The variables applied in the following derivations are summarized below.

T_{UN} : Total restoration time for unidirectional restoration procedure.

T_{BI} : Total restoration time for bidirectional restoration procedure.

F : Failure detection time ($F = 625 \mu s$).

$P_{i,j}$: Propagation delay time between node i and j ($P_{i,j} = 3 ms$).

M : Message processing time at a node. ($M = 10 \mu s$).

C : Switch configuration time ($C = 50 \mu s$).

$L_{i,h}$: The total number of eligible restoration links within limited hop count ($h \geq 2$) for the failure link i .

$$\begin{aligned}
\text{For } h = \text{even}, \quad B_h &= B_{h-1} + \frac{h \cdot \sum_{i=1}^L (L_{i,h} - L_{i,h-1})}{L} \quad (h \geq 2), (B_1 = 0, L_{i,1} = 0) \\
\text{For } h = \text{odd}, \quad B_h &= B_{h-1} + \frac{(h+1) \cdot \sum_{i=1}^L (L_{i,h} - L_{i,h-1})}{L} \quad (4.23)
\end{aligned}$$

$$h \geq 2, \quad U_h = \frac{4 \cdot \sum_{i=1}^L L_{i,h}}{L} \quad (4.24)$$

The derivations of this section are based on the assumption that the acknowledgement messages are not piggy-back transmitted as shown in Figure 3-7. In other words, the control packet amounts and reconfiguration time procedures will always require a separate acknowledgement packet rather than being looped back on a reverse direction control message (acknowledgement field). Thus, the final results will provision a larger number of control packets and restoration time compared to when piggy-back acknowledgements are used. The gain in number of generated control packets and restoration time is common to these two restoration procedures and therefore does not obscure the analysis results.

In addition, to fully consider piggy-back acknowledgement packets the time issues of all packet arrivals must be considered to accurately estimate which packets could possibly include a piggy-back acknowledgement. This consideration significantly complicates the regulation derivations and is beyond the focus of this analysis.

CHAPTER IV

4. OBSERVATION OF THE RESULTS

4.1 Analysis of Performance

In order to compare the results from the LP, EST, and PST algorithms, a computer simulation was conducted based on the reference network shown in Figure 4-5 [20]. The results in Figure 4-6 show how the RSC algorithms (i.e., EST and PST) compare to the OSC solution obtained from the LP algorithm [20]. We can see that the RSC algorithms do not provide a good enough solution compared to the OSC solution in regard of spare capacity although the simple procedures enable benefits in computation time and complexity in dynamic operations.

In summary, there are three advantages of the proposed RSC algorithms. First, the complexity of algorithms should be seriously considered. From the flow charts in Figure 3-3 and 4-6, it is easily found that the LP algorithm is much more complex compared to EST and PST. Based on the observation of the computer simulation the LP requires much more computation complexity when the input data such as the number of node and the number of possible restoration path (P_i) are the same as those in the EST and PST algorithm. This is due to the LP based min-max iterative procedures applied is search of the minimum spare capacity. Secondly, when deploying the RSC algorithms, the network automatically knows the restoration flows ($f_{i,n}$) going through each of possible restoration paths (P_i). But in the case of the LP, the network first has to find the OSC solution and then the restoration paths can be computed separately. The third advantage

of the RSC algorithms is that the algorithms can be easily expanded for multiple cut cases without much modification to the algorithms. Compared to this the LP algorithm will have to be modified to incorporate the operations for multiple link failures or cuts.

There are four simulation results presented in this thesis. First, the results of the EST and PST applied over an example network using the actual population of the nodes (cities in the U.S.A.) are shown in Figure 4-1. Figure 4-2 shows how efficient the EST and PST algorithms allocate redundant spare capacity under the limited hop count restriction. The EST and PST algorithm were implemented in C++ are provided in appendix C. In the second and third experiments, we compare the restoration time and average numbers of control packets required for OSC and RSC solutions at equivalent stages (shown in Figure 3-8) and the results are provide in Figure 4-3 and 5-4. The result in Figure 4-6 compares the amount of spare capacity reserved for the EST, PST and the LP algorithm for the example network in Figure 4-5, how close RSC solutions can approximate OSC solution by using EST and PST.

4.2 Actual Traffic Assignment on the Network for Realistic Simulation

This section explains how an estimated practical network is designed using real population of the cities. The traffic assignment result is obtained using MATLAB in appendix B. The purpose of the estimated actual network test is to show how much the EST and PST algorithm reduce the redundant spare capacity depending on the limited hop count (h).

The actual population of each city is shown in Table 4-1. Original working traffic is assigned by using the shortest path routing algorithm. For instance, one of the city populations divides proportionally based on the other city's population. This is assumed to be the demand traffic from one city to the others. In Figure 4-1, the original network is assigned as an actual working traffic using the shortest path algorithm. The number on the link indicates the number of working wavelength channels (i.e., OC-192). We assume that the average individual user data uses 50 Kbps. In this case, one wavelength channel carries data for 196,000 (i.e. $\frac{9.8 \times 10^9}{50 \times 10^3}$) at once. For example, Boston's (C1) population is 5,667,225 that require 29 OC-192 wavelength channels to be divided proportionally based on the other city's population.

1. The estimated traffic between two cities C_i and C_j can be computed as,

$$= \frac{\text{number_of_population}(C_i) \times \text{number_of_population}(C_j)}{\text{Total_population}(= \sum (\text{individual_city_population}(C_i))}, (i \neq j) \quad (4-25)$$

2. We assume that the average individual user data is 50 Kbps.
3. We use OC-192, wavelength channel that is 9.8 Mbps in the every WDM link.
4. The actual traffic is assigned by shortest path algorithm.

In appendix A, the population (Table 4-1) of the example network in Figure 4-1 is provided in a matrix form.

Name of city	Population	Symbol	Name of city	Population	Symbol
Boston	5,667,225	C1	Kansas City	1,755,899	C10
Albany	869,474	C2	Dallas	4,909,523	C11
New York	20,196,649	C3	Houston	4,493,741	C12
Cleveland	2,910,616	C4	Denver	2,417,908	C13
Washington D.C.	7,359,044	C5	Salt lake city	1,275,076	C14
Chicago	8,885,919	C6	Phoenix	3,013,696	C15
Cincinnati	1,960,995	C7	San Francisco	6,873,645	C16
Atlanta	3,857,097	C8	Los Angeles	16,036,587	C17
Miami	3,711,102	C9	Total Population	96,194,196	

Table 4-1 populations of the cities

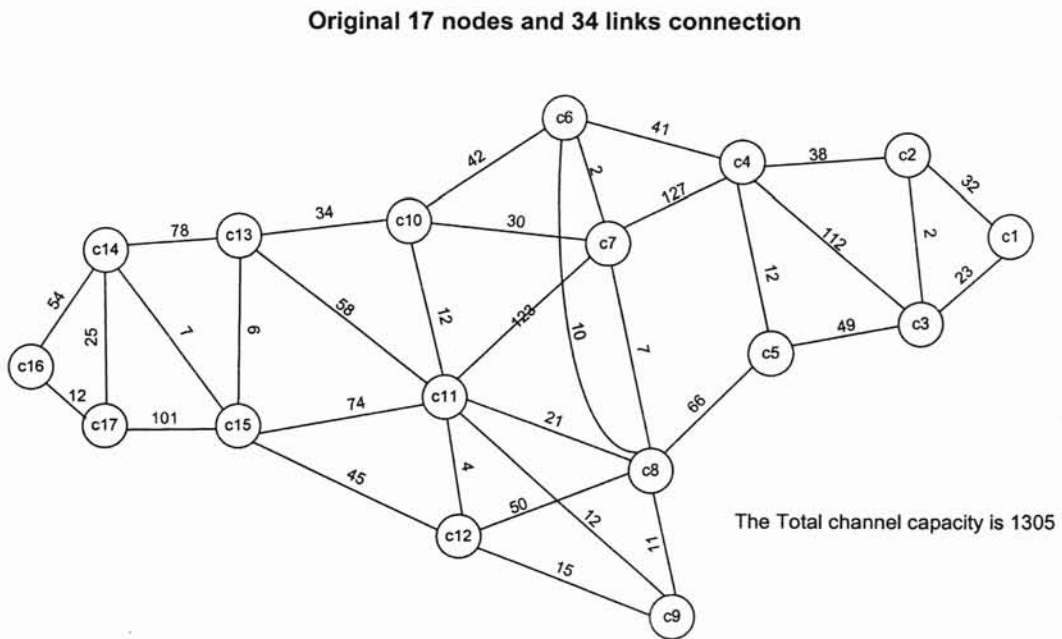


Figure 4-1 An example of a backbone optical network connection.

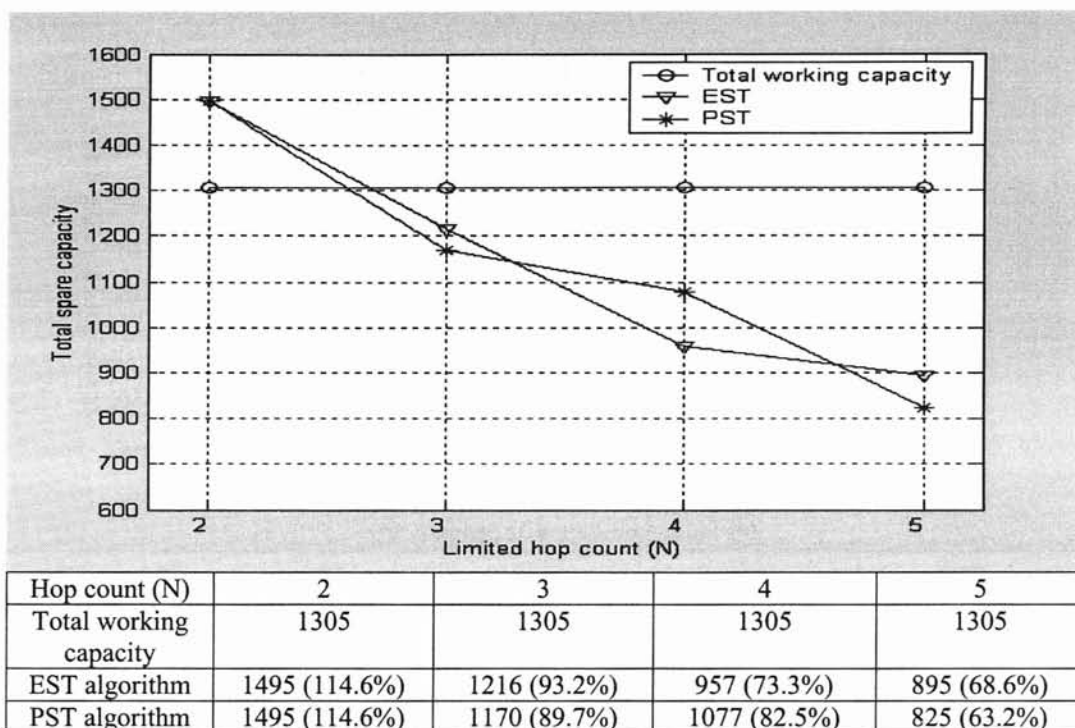


Figure 4-2. The total spare capacity using ETS and PTS algorithm in the actual network.

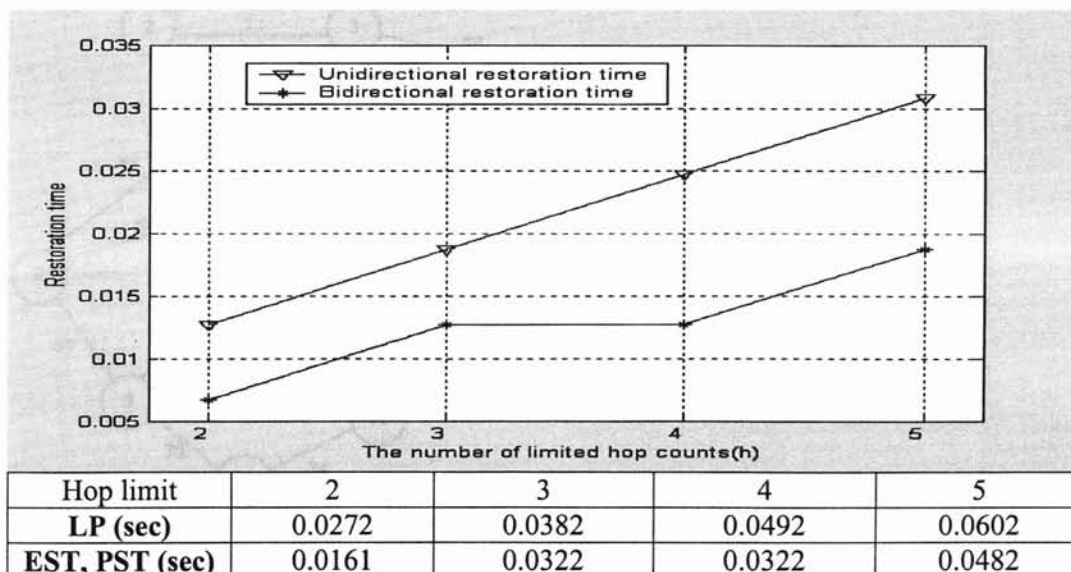


Figure 4-3 Restoration time.

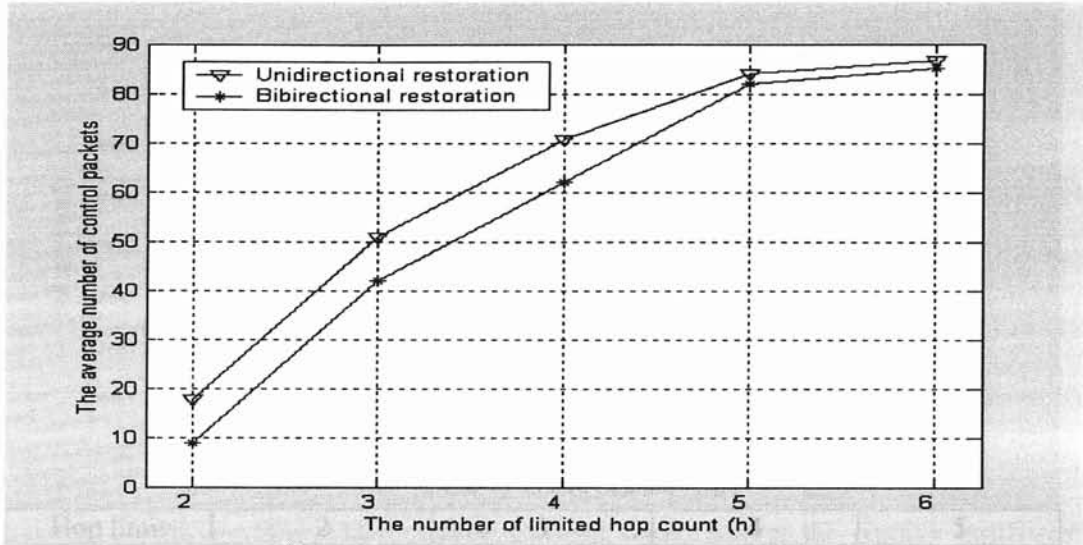


Figure 4-4 the average number of control packets.

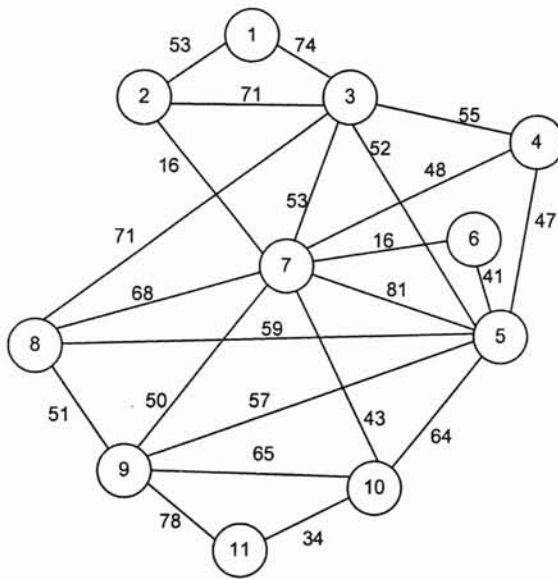


Figure 4-5 Test network Ref. [20].

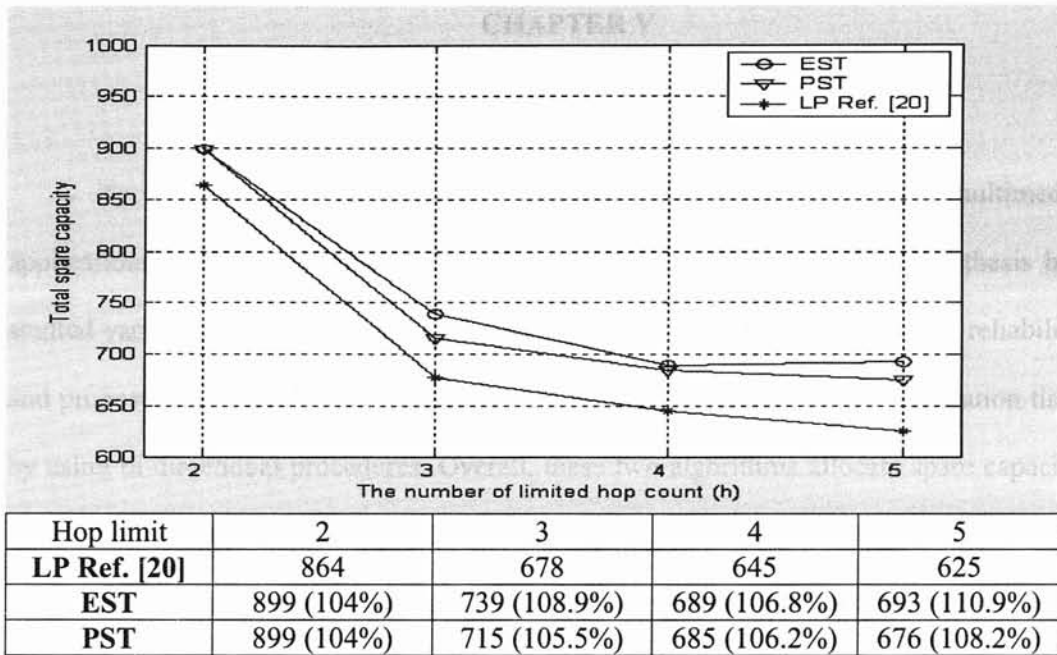


Figure 4-6 Total spare capacity between LP, EST, and PST algorithm.

CHAPTER V

5. CONCLUSIONS

To support high-speed data communication and real-time multimedia applications, a fast restoration process is essential in optical networks. This thesis has studied various failure recovery techniques for achieving high optical network reliability and proposes new simple algorithms (i.e., EST and PST) to achieve fast restoration time by using bi-directional procedures. Overall, these two algorithms allocate spare capacity over a network by distributing original working traffic among the eligible restoration routes either equally or proportionally. Due to this mechanism, EST and PST can deal easily with fluctuation in the working traffic. In other words, the two end nodes of the failure link are responsible for assigning the original working traffic among the restoration paths in a bi-directional way. However, the OSC solution cannot deal with such a case without frequently updating the RFT. Our theoretical analysis and simulation results have shown that although the proposed EST and PST algorithms in this thesis are sub-optimal in restoration capacity reservation, they can provide quick restoration due to the specific mechanisms, and they can be effectively and efficiently implemented due to the simplicity of the applied algorithms, thus making them more effective for restoration operation.

REFERENCES

- [1] S. Ramamurthy and B. Mukherjee, "Survivable WDM Mesh Networks, Part I- Protection," *IEEE Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies Proceedings*, vol. 2, pp. 744 –751, 1999.
- [2] S. Ramamurthy and B. Mukherjee, "Survivable WDM Mesh Networks Part II- Restoration," *Proc. IEEE International Conference on Communications*, pp. 2023 –2030, 1999.
- [3] D. Zhou and S. Subramaniam, "Survivability in Optical Networks," vol. 14, *IEEE Network*, pp.16-23, 2000.
- [4] T. Wu, *Fiber Network Service Survivability*. Norwood, MA: Artech House, 1992.
- [5] T. Wu and N. Yoshikai, *ATM Transport and Network Integrity*. MA: Academic Press, 1997.
- [6] S. V. Kartalopoulos, *Introduction to DWDM Technology*. NJ: IEEE Press, 2000.
- [7] B.A. Coan, W. E. Leland, M. P. Vecchi, A. Weinrib, and L.T. Wu, "Using Distributed Topology and Preplanned Configurations to Achieve Trunk Network Survivability," *IEEE Transaction on Reliability*, vol. 40, no. 4, pp. 404-414, 1999.
- [8] R. R. Iraschko, M. H. MacGregor, and W. D. Grover, "Optimal Capacity Placement for Path Restoration in Mesh Survivable Networks," *Proc. IEEE International Conference on Communications*, vol. 3, pp. 1568 –1574, 1996.
- [9] Z. Zhang, J. Fu, D. Guo, and L. Zhang, "Lightpath Routing for Intelligent Optical Networks," *IEEE Network*, vol. 15, no.4, pp. 28-35, 2000.

- [10] H. Zang, J. P. Jue, L. Sahasrabudde, R. Ramamurthy, and B. Mukherjee, "Dynamic Lightpath Establishment in Wavelength-Routed WDM Networks," *IEEE Communications Magazine*, vol. 39, no. 9, pp. 100-108, 2001.
- [11] G. Li, J. Yates, R. Doverspike, and D. Wang, "Experiments in Fast Restoration using GMPLS in Optical/Electronic Mesh Networks," *Proc. OFC'01*, vol. 4, pp. PD34-P1-3, 2001.
- [12] T. H. Oh, T. M. Chen, and J. L. Kennington, "Fault Restoration and Spare Capacity Allocation with QoS Constrains for MPLS Networks," *Proc. IEEE GLOBECOM'00*, vol. 3, pp. 1731-1735, 2000.
- [13] M. Herzberg, and S. J. Bye, "An Optimal Spare-capacity Assignment Model for Survivable Networks with Hop Limits," *Proc. IEEE GLOBECOM*, vol. 3, pp. 1601-1606, 1994.
- [14] J. Strand, A. L. Chiu, and R. Tkach, "Issues for Routing in The Optical Layer," *IEEE Communications Magazine*, vol. 39, no. 2, pp. 81-87, 1994.
- [15] G. Mohan, C. Siva Ram Muethy, and A. K. Somani, "Efficient Algorithms for Routing Dependable Connections in WDM Optical Networks," *IEEE/ACM Transactions on Networking*, vol. 9, no. 5, pp. 553-566, 2001.
- [16] J. Sosnosky, "Service Applications for SONET DCS Distributed Restoration," *IEEE Journal on selected Areas in Communications*, vol. 12, no. 1, pp. 59-68, 1994.
- [17] C. E. Chow, J. Bicknell, S. McCaughey, and S. Sced, "A Fast Distributed Network Restoration Algorithm," *Proc. Twelfth Annual International Phoenix Conference on*, pp. 261-267, 1993.

- [18] L. Sahasrabudde, S. Ramamurthy, and B. Mukherjee, "Fault Management in IP-Over WDM Networks: WDM Protection Versus IP Restoration," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 1, pp. 21-33, 2002.
- [19] "Multi-protocol Label Switching (MPLS)," International Engineering Consortium, <http://www.iec.org/online/tutorials/mpls>, 2002.
- [20] M. Herzberg, S. J. Bye, and A. Utano, "The Hop-limit Approach for Spare-Capacity Assignment in Survivable Networks," *IEEE/ACM Transaction on Networking*, vol. 3, no. 6, pp. 775-784, 1995.
- [21] Y. Liu, D. Tipper, and P. Siripongwutikon, "Approximating Optimal Spare Capacity Allocation by Successive Survivable Routing," *Proc. IEEE INFOCOM'01. and Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 2, pp. 699-708, 2001.
- [22] C. Mauz, "Allocation of Spare Capacity for Shared Protection of Optical Paths in Transport Networks," *Proc. Third International Workshop on Design of Reliable Communicatio Networks, 2001*.
- [23] B. Zhou and H. T. Mouftah, "Spare Capacity Planning Using Survivable Alternate Routing for Long-Haul WDM Networks," *Proc. of IEEE ISCC'02*, pp. 732-738, 2002.

APPENDIXES

Appendix A - The Traffic Demand computation in Matrix Form

In appendix A, the traffic demands between cities are calculated based on equation (4-25) as show following.

$$\text{Traffic demand} = \frac{\text{number_of_population}(C_i) \times \text{number_of_population}(C_j)}{\text{Total_population}(= \sum (\text{individual_city_population}(C_i))}, (i \neq j) \quad (4-25)$$

	C1	C2	C2	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	sum row
C1	333881	51225	1189874	171477	433554	523509	115531	227239	218637	103448	289242	264746	142450	75120	177550	404957	944786	5,667,225
C2	51225	7859	182552	26308	66516	80317	17725	34863	33544	15871	44376	40618	21855	11525	27240	62129	144950	869,474
C3	1189874	182552	4240429	611104	1545083	1865661	411725	809825	779172	368663	1030789	943493	507657	267711	632747	1443170	3366994	20,196,649
C4	171477	26308	611104	88069	222668	268868	59335	116707	112289	53129	148551	135970	73160	38581	91188	207981	485230	2,910,616
C5	433554	66516	1545083	222668	562981	679790	150020	295075	283907	134330	375588	343780	184975	97546	230554	525847	1226830	7,359,044
C6	523509	80317	1865661	268868	679790	820835	181147	356299	342812	162201	453516	415108	223354	117785	278390	634952	1481376	8,885,919
C7	115531	17725	411725	59335	150020	181147	39976	78630	75654	35795	100085	91608	49291	25993	61437	140125	326919	1,960,995
C8	227239	34863	809825	116707	295075	356299	78630	154658	148804	70406	196857	180185	96951	51127	120840	275612	643019	3,857,097
C9	218637	33544	779172	112289	283907	342812	75654	148804	143172	67741	189406	173365	93281	49192	116266	265180	618680	3,711,102
C10	103448	15871	368663	53129	134330	162201	35795	70406	67741	32052	89617	82027	44136	23275	55011	125469	292727	1,755,899
C11	289242	44376	1030789	148551	375588	453516	100085	196857	189406	89617	250570	229350	123404	65077	153812	350814	818469	4,909,523
C12	264746	40618	943493	135970	343780	415108	91608	180185	173365	82027	229350	209926	112953	59566	140786	321104	749154	4,493,741
C13	142450	21855	507657	73160	184975	223354	49291	96951	93281	44136	123404	112953	60776	32050	75751	172774	403091	2,417,908
C14	75120	11525	267711	38581	97546	117785	25993	51127	49192	23275	65077	59566	32050	16901	39947	91112	212569	1,275,076
C15	177550	27240	632747	91188	230554	278390	61437	120840	116266	55011	153812	140786	75751	39947	94417	215346	502415	3,013,696
C16	404957	62129	1443170	207981	525847	634952	140125	275612	265180	125469	350814	321104	172774	91112	215346	491163	1145909	6,873,645
C17	944786	144950	3366994	485230	1226830	1481376	326919	643019	618680	292727	818469	749154	403091	212569	502415	1145909	2673468	16,036,587
column sum	5,667,225	869,474	20,196,649	2,910,616	7,359,044	8,885,919	1,960,995	3,857,097	3,711,102	1,755,899	4,909,523	4,493,741	2,417,908	1,275,076	3,013,696	6,873,645	16,036,587	96,194,196

Appendix B – MATLAB Program

In Figure 4-1, the numbers on the connection represent the amount of data traffic, which is obtained by using the program code as below.

```
function [netpath, index] = nettree2 (index0, from, to, n, connection, traffic, netpath)
%
% Function file: nettree.m
%
% Purpose
% To find out the cell matrix of netpath by using Open Shortest Path Fist (OSPF),
% and then use this result to get the total traffic of the network map.
%
% Define variables:
% n = the number of nodes
% index = index variable.
% index0 = index variable for command isempty.
% netpath = the cell matrix to show from source to destination path in the each cell.
% traffic = the matrix for traffic when consider connection.
% connection = the matrix to show the connection in the map.
%
% Record of revision:
% Date      Programmer      Description of change
% -----
% 04/25/01  Lee, Byoung-yong  Connection path traffic

for k = 1:n
    if isempty(index0), index=from;
    else, index=index0;
    end

    if ~any(k==index)
        lastindex=index(end);
        if connection ( lastindex, k) ==1
            index = [index, k];
            if k == to,
                if isempty(netpath{from, to}), netpath{from, to}=index;
                else
                    if length(netpath {from, to}) > length(index),
                        netpath{from, to} = index;
                    elseif length(netpath {from, to})==length(index),
                        trafficpath1 = 0;
                        trafficpath2 = 0;
                        for kk=1: length(index)-1,
                            trafficpath1 = trafficpath1 + traffic(netpath{from, to}(kk), netpath{from, to}(kk+1));
                            trafficpath2 = trafficpath2 + traffic(index(kk),index(kk+1));
                        end
                        if trafficpath1 > trafficpath2,
                            netpath{from, to} = index;
                        end
                    end
                end
            else
                [netpath, index]=nettree2( index, from, to, n, connection, traffic, netpath);
            end
        end
    end
end

% Script file: pathfind.m
%
% purpose:
% To figure out the netpath with using the function file, which is using
% tree algorithm to find out shortest path.
%
% Define variables:
% netpath = the cell matrix to show from source to destination path in the each cell.
% traffic0 = the matrix for traffic.
% connection = the matrix to show the connection in the map.

% input data of the number of nodes.
n = input('Number of nodes = ');
connection = [0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0;
             1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0;
             1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0;
             0 1 1 0 1 1 1 0 0 0 0 0 0 0 0 0;
             0 0 1 1 0 0 0 1 0 0 0 0 0 0 0 0; %original mesh connection
             0 0 0 1 0 0 1 1 0 0 0 0 0 0 0 0;
             0 0 0 1 0 1 0 1 0 0 1 0 0 0 0 0;
             0 0 0 0 1 1 1 0 1 0 1 0 0 0 0 0;
             0 0 0 0 0 0 0 1 0 0 1 1 0 0 0 0;
             0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0;
             0 0 0 0 0 0 1 1 1 0 1 1 0 1 0 0;];
```

```

0 0 0 0 0 0 0 0 1 0 1 0 0 0 1 0 0;
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 0;
0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 1;
0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 1;
0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1;
0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0;

% traffic0 matrix is calculated by
% this equation
% traffic0 = ( Source population (from) * destination population (to) ) / Total population of city.
traffic0=[333881 51225 1189874 171477 433554 523509 115531 227239 218637 103448 289242 264746 142450 75120 177550 404957 944786;
51225 7859 182552 26308 66516 80317 17725 34863 33544 15871 44376 40618 21855 11525 27240 62129 144950;
1189874 182552 4240429 611104 1545083 1865661 411725 809825 779172 368663 1030789 943493 507657 267711 632747 1443170 3366994;
171477 26308 611104 88069 222668 268868 59335 116707 112289 53129 148551 135970 73160 38581 91188 207981 485230;
433554 66516 1545083 222668 562981 679790 150020 295075 283907 134330 375588 343780 184975 97546 230554 525847 1226830;
523509 80317 1865661 268868 679790 820835 181147 356299 342812 162201 453516 415108 223354 117785 278390 634952 1481376;
115531 17725 411725 59335 150020 181147 39976 78630 75654 35795 100085 91608 49291 25993 61437 140125 326919;
227239 34863 809825 116707 295075 356299 78630 154658 148804 70406 196857 180185 96951 51127 120840 275612 643019;
218637 33544 779172 112289 283907 342812 75654 148804 143172 67741 189406 173365 93281 49192 116266 265180 618680;
103448 15871 368663 53129 134330 162201 35795 70406 67741 32052 89617 82027 44136 23275 55011 125469 292727;
289242 44376 1030789 148551 375588 453516 100085 196857 189406 89617 250570 229350 123404 65077 153812 350814 818469;
264746 40618 943493 135970 343780 415108 91608 180185 173365 82027 229350 209926 112953 59566 140786 321104 749154;
142450 21855 507657 73160 184975 223354 49291 96951 93281 44136 123404 112953 60776 32050 75751 172774 403091;
75120 11525 267711 38581 97546 117785 25993 51127 49192 23275 65077 59566 32050 16901 39947 91112 212569;
177550 27240 632747 91188 230554 278390 61437 120840 116266 55011 153812 140786 75751 39947 94417 215346 502415;
404957 62129 1443170 207981 525847 634952 140125 275612 265180 125469 350814 321104 172774 91112 215346 491163 1145909;
944786 144950 3366994 485230 1226830 1481376 326919 643019 618680 292727 818469 749154 403091 212569 502415 1145909 2673468 ];

% traffic1 is the matrix of traffic, which is considered connection matrix.
traffic1 = (traffic0*50000)/(9.8*10^9); % individual people use 50 Kbps
%traffic1 = ones(17);
traffic = connection.*traffic1;
netpath = cell(n,n);
for from = 1:n % 'from' is the source 1 through n.
    from=from
    for to = 1:n % 'to' is the destination 1 through n.
        if from~=to % if 'from' is not same 'to' start to calculate the path.
            index0=[];
            [netpath, index] = nettree ( index0, from, to, n, connection, traffic, netpath);
        end
    end
end

% net traffic calculation
tmp=zeros(n,n,n);
for i3=1:n
    for i1=1:n,
        tmp1=netpath(i3,i1);
        for i2=1:length(tmp1)-1,
            tmp(tmp1(i2),tmp1(i2+1),i3)=tmp(tmp1(i2),tmp1(i2+1),i3)+traffic1(i3,tmp1(end));
        end
    end
end

total0=zeros(n);
for i1=1:n
    for i2=1:n
        total0(i1,i2)=sum(tmp(i1,i2,:));
    end
end

% The total traffic in the map with using OSPF
total = total0 + total0';
T3 = ceil(total); % total traffic devided by oc-192 (= 9.8 Mbps)
A3 = sum(sum(T3))/2
% The graph of total traffic
bar3(T3,0.3);

```

Appendix C – C++ Program

This program can obtain the result of RSC assignment (i.e., EST and PST) on the example of network, Figure 4-1. This program code can obtain the result Figure 4-6 changing the connection and traffic matrix for the test network, Figure 4-5.

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <iostream.h>

int connect[17][17] = {
{ 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
{ 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
{ 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
{ 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
{ 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
{ 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0 },
{ 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0 },
{ 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0 },
{ 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0 },
{ 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0 },
{ 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0 },
{ 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0 },
{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0 },
{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0 },
{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0 }};

int traffic[17][17] = {
{ 0, 32, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
{ 32, 0, 2, 38, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
{ 23, 2, 0, 112, 49, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
{ 0, 38, 112, 0, 12, 41, 127, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
{ 0, 0, 49, 12, 0, 0, 0, 66, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
{ 0, 0, 0, 41, 0, 0, 2, 10, 0, 42, 0, 0, 0, 0, 0, 0, 0, 0 },
{ 0, 0, 0, 127, 0, 2, 0, 7, 0, 0, 123, 0, 0, 0, 0, 0, 0, 0 },
{ 0, 0, 0, 66, 10, 7, 0, 11, 0, 21, 50, 0, 0, 0, 0, 0, 0, 0 },
{ 0, 0, 0, 0, 0, 0, 0, 11, 0, 0, 15, 12, 0, 0, 0, 0, 0, 0 },
{ 0, 0, 0, 0, 0, 42, 0, 0, 0, 0, 12, 0, 34, 0, 0, 0, 0, 0 },
{ 0, 0, 0, 0, 0, 0, 0, 123, 21, 15, 12, 0, 4, 58, 0, 74, 0, 0 },
{ 0, 0, 0, 0, 0, 0, 0, 50, 12, 0, 4, 0, 0, 0, 45, 0, 0, 0 },
{ 0, 0, 0, 0, 0, 0, 0, 0, 34, 58, 0, 0, 78, 6, 0, 0, 0, 0 },
{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 78, 0, 7, 54, 25, 0 },
{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 74, 45, 6, 7, 0, 0, 101, 0 },
{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 54, 0, 0, 12, 0 },
{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 25, 101, 12, 0, 0, 0, 0 }};

int HopCount;
int** path;
float** spare;
float** propSpare;
int** used;
int NoPaths = 50;
int MaxPaths = 50;

int NodeCount = 17;

int main(void)
```

```

{

int temp = 0;
int findpath(int , int );
int* FinalHopCount;
int status = 0;
int same = 0;

used = new int*[NoPaths];
for (int c=0; c<NoPaths; c++)
{
used[c] = new int[NoPaths];
}

path = new int*[NoPaths];
for (c=0; c<NoPaths; c++)
{
path[c] = new int[NoPaths];
}

spare = new float*[NodeCount];
for (c=0; c<NodeCount; c++)
{
spare[c] = new float[NodeCount];
}

for (int a=0;a<NodeCount;a++)
{
for (int z=0;z<=NodeCount;z++)
{
spare[a][z] = 0;
}
}

propSpare = new float*[NodeCount];
for (c=0; c<NodeCount; c++)
{
propSpare[c] = new float[NodeCount];
}

for (a=0;a<NodeCount;a++)
{
for (int z=0;z<=NodeCount;z++)
{
propSpare[a][z] = 0;
}
}

for (HopCount = 2; HopCount<6;HopCount++)
{
cout << "\n\n\t\t\t\t\tHOP COUNT = " << HopCount << endl << endl;

for (int a=0;a<NodeCount;a++)
{
for (int z=0;z<NodeCount;z++)
{
spare[a][z] = 0;
}
}

for (a=0;a<NodeCount;a++)
{
for (int z=0;z<=NodeCount;z++)
{
propSpare[a][z] = 0;
}
}
}

```



```

}
}
}
}
FinalHopCount[FHCount] = FHop - 1;
FHCount++;
cout << endl;
cout << flush;
}
int hT = 0;
double t = 1;
float deno = 0.0;
for (int g=0;g<FHCount;g++)
{
hT += FinalHopCount[g];
t *= FinalHopCount[g];
deno += (float) 1.0/(FinalHopCount[g]);
}

double delta = t * deno;

for (a=0;a<NodeCount;a++)
{
for (int z=0;z<=NodeCount;z++)
{
used[a][z] = 0;
}
}

for (a=0;a<status;a++)
{
for (int z=0;z<=HopCount;z++)
{
if (path[a][z] >= 0 )
{
if ((z < HopCount) && (traffic[i][k] > 0))
{
int temp1 = path[a][z];
int temp2 = path[a][z+1];
float tempProp = (float) ((traffic[i][k]/delta) * (t/FinalHopCount[a]));
if (temp2 >= 0)
{
if (used[temp1][temp2] == 1)
{
propSpare[temp1][temp2] += tempProp;
propSpare[temp2][temp1] += tempProp;
}

else
{
if (tempProp > propSpare[temp1][temp2])
{
propSpare[temp1][temp2] = tempProp;
propSpare[temp2][temp1] = tempProp;
used[temp1][temp2] = 1;
used[temp2][temp1] = 1;
}
}
}
}
}
}
}
}

}
}
}

cout << "\n\n\tEQUAL SHARED - SPARE CAPACITY ASSIGNEMENT FOR HOP COUNT " << HopCount<< endl;

```

```

int totalSpare = 0;
for (a=0;a<NodeCount;a++)
{
for (int z=0;z<NodeCount;z++)
{
cout << (int)spare[a][z] << "\t";
totalSpare += spare[a][z];
}
}
cout << endl;
}
cout << "Total Spare Capacity for Hop Count " << HopCount << " using equal shared spared capacity assignment = " << totalSpare/2;
cout << endl;
cout << flush;

cout << "\n\n\tPROPORTIONAL SHARED - SPARE CAPACITY ASSIGNEMENT FOR HOP COUNT " << HopCount<< endl;
totalSpare = 0;
for (a=0;a<NodeCount;a++)
{
for (int z=0;z<NodeCount;z++)
{
cout << ceil(propSpare[a][z]) << "\t";
totalSpare += propSpare[a][z];
}
}
cout << endl;
}
cout << "Total Spare Capacity for Hop Count " << HopCount << " using proportional shared spared capacity assignment = " <<
totalSpare/2;
cout << endl;
cout << flush;
}

FinalHopCount = NULL;
delete[] FinalHopCount;

for (c=0; c<NoPaths; c++)
{
path[c] = NULL;
delete[] path[c];
}
delete[] path;

for (c=0; c<NodeCount; c++)
{
spare[c] = NULL;
delete[] spare[c];
}
delete[] spare;

for (c=0; c<NodeCount; c++)
{
propSpare[c] = NULL;
delete[] propSpare[c];
}
delete[] propSpare;

for (c=0; c<NodeCount; c++)
{
used[c] = NULL;
delete[] used[c];
}
delete[] used;

cout << endl;
cout << flush;
return 0;
}

int findpath(int p, int q)
{

```

```

int pathCount = 0;
int pNodes[20], qNodes[20];
int p2Nodes[20], q2Nodes[20];
int pinterNode[20], qinterNode[20];
int tempPNode=0, tempQNode=0;
int p2NodeCount=0, q2NodeCount=0;
int pNodeCount=0, qNodeCount=0, pInter=0, qInter=0;
int Change = 0;
/*
int* pNodes;
int* qNodes;
int* p2Nodes;
int* q2Nodes;
int* pinterNode;
int* qinterNode;

pNodes = new int[10];
qNodes = new int[10];
p2Nodes = new int[10];
q2Nodes = new int[10];
pinterNode = new int[10];
qinterNode = new int[10];

for (int b=0;b<10;b++)
{
pNodes[b] = 0;
qNodes[b] = 0;
p2Nodes[b] = 0;
q2Nodes[b] = 0;
pinterNode[b] = 0;
qinterNode[b] = 0;
}
*/
for (int x=0;x<NoPaths;x++)
{
for (int y=0;y<NoPaths;y++)
{
path[x][y] = -1;
}
}

for (x=0;x<16;x++)
{
pNodes[x]= 0;
qNodes[x] = 0;
p2Nodes[x]= 0;
q2Nodes[x] = 0;
pinterNode[x]= 0;
qinterNode[x] = 0;
}

for (x=0;x<NodeCount;x++)
{
if (x != p)
{
if ((connect[p][x] == 1))
{
pNodes[pNodeCount] = x;
pNodeCount++;
}

if ((connect[x][q] == 1))
{
qNodes[qNodeCount] = x;
qNodeCount++;
}
}
}
}

```

```

for (x=0;x<pNodeCount;x++)
{
pInter = pNodes[x];
for (int z=0;z<qNodeCount;z++)
{
qInter = qNodes[z];
if (pInter == qInter)
{
if ((connect[p][pInter] == 1) && (connect[pInter][q] == 1))
{
path[pathCount][0] = p;
path[pathCount][1] = pInter;
path[pathCount][2] = q;
pathCount++;
}
}
else
{
if ((connect[pInter][qInter] == 1) && (HopCount > 2))
{
if ((connect[p][pInter] == 1) && (connect[pInter][qInter] == 1) && (connect[qInter][q]))
{
path[pathCount][0] = p;
path[pathCount][1] = pInter;
path[pathCount][2] = qInter;
path[pathCount][3] = q;
pathCount++;
}
}
}
}

int max;
if (pNodeCount > qNodeCount)
max = pNodeCount;
else
max = qNodeCount;

for (x=0;x<max;x++)
{
tempPNode = pNodes[x];
tempQNode = qNodes[x];

for (int y=0;y<NodeCount;y++)
{
if (tempPNode >= 0)
{
if ((connect[tempPNode][y] == 1) && (tempPNode != y))
{
p2Nodes[p2NodeCount] = y;
pinterNode[p2NodeCount] = tempPNode;
p2NodeCount++;
}
}

if (tempQNode >= 0)
{
if ((connect[tempQNode][y] == 1) && (tempQNode != y))
{
q2Nodes[q2NodeCount] = y;
qinterNode[q2NodeCount] = tempQNode;
q2NodeCount++;
}
}
}
}
}

```

```

if ((HopCount > 3) && (HopCount < 6))
{
for (x=0;x<p2NodeCount;x++)
{
pInter = p2Nodes[x];
for (int z=0;z<q2NodeCount;z++)
{
qInter = q2Nodes[z];
if ((p != pInter) && (p != qInter) && (qInter != q) && (pInter != q) && (pinterNode[x] != qInter) && (pinterNode[x] !=
qinterNode[z]) && (pInter != qinterNode[z]) && (pInter && qinterNode[z]))
{
if (pathCount < MaxPaths)
{
if (pInter == qInter)
{
int temp1 = pinterNode[x];
int temp2 = qinterNode[z];

if ((connect[p][temp1] == 1) && (connect[temp1][pInter] == 1) && (connect[pInter][temp2] == 1) && (connect[temp2][q] == 1))
{
path[pathCount][0] = p;
path[pathCount][1] = pinterNode[x];
path[pathCount][2] = pInter;
path[pathCount][3] = qinterNode[z];
path[pathCount][4] = q;
pathCount++;
}
}
else
{
if ((connect[pInter][qInter] == 1) && (HopCount > 4))
{
int temp1 = pinterNode[x];
int temp2 = qinterNode[z];

if ((connect[p][temp1] == 1) && (connect[temp1][pInter] == 1) && (connect[pInter][qInter] == 1) && (connect[qInter][temp2] == 1)
&& (connect[temp2][q] == 1))
{
path[pathCount][0] = p;
path[pathCount][1] = pinterNode[x];
path[pathCount][2] = pInter;
path[pathCount][3] = qInter;
path[pathCount][4] = qinterNode[z];
path[pathCount][5] = q;
pathCount++;
}
}
}
}
}
}
}
}
}
return pathCount;
}

```

VITA 2

Byoung-yong, Lee

Candidate for the Degree of

Master of Science

Thesis: AN INVESTIGATION OF FAST RESTORATION AND EFFICIENT SPARE
CAPACITY TECHNOLOGIES BASED ON PREPLANNED RESTORATION IN
OPTICAL NETWORKS

Major Field: Electrical Engineering

Biographical:

Personal Data: Born in Seoul, Korea, On April 30, 1974, the son of
Tae-soo, Lee and Kae-yim, Lee.

Education: Received a Bachelor of Engineering degree in Electronics and
Communication Engineering from Yosu National University, Yosu in March 1998.
Completed the requirements for the Master of Science degree with a major in
Electrical Engineering at the Oklahoma State University in May 2002.

Experience: Employed as a Research Assistant by the ACSEL Laboratories at the School
of Electrical and Computer Engineering, Oklahoma State University, Aug. 2000 to
present.

Professional Membership: Student member of the honorary Institute of Electrical and
Electronics Engineers, Inc. (IEEE) for the academic years 2000 and 2001.