SURVIVABILITY SCHEMES FOR METRO ETHERNET NETWORKS

QIU JIAN

(B.Eng. Xi'an Jiaotong University)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2010

To my parents...

who gave me their unconditional support, love, and wishes...

Acknowledgements

I am truly indebted to my supervisors, Assoc. Prof. Gurusamy Mohan and Prof. Chua Kee Chaing, for their continuous guidance and support during this work. Without their guidance, this work would not be possible.

I am deeply indebted to the National University of Singapore for the award of a research scholarship. I would like to give thanks to all the researchers in the Optical Networks Laboratory, who greatly enriched both my knowledge and life with their intelligence and optimism. I would also thank my family and all my friends for their love, encouragement and support.

Qiu Jian

May 2010

Contents

Summary

Ethernet is becoming a preferred technology to be extended to Metro Area Networks (MANs) due to its low cost, simplicity and ubiquity. However, the traditional spanning tree based Ethernet protocol does not meet the requirements for MANs in terms of fast recovery and guaranteed protection, despite the advancement of Ethernet standardization and commercialization. In the literature, some survivability schemes in Metro Ethernet networks have been proposed to solve the slow spanning tree convergence problem. Most of these schemes are either centralized or have high signaling overhead. In addition, few works have considered backup capacity reservation in Metro Ethernet networks to provide guaranteed protection. The aim of our study is to propose and analyze novel distributed survivability

schemes which can provide fast and guaranteed protection for Metro Ethernet networks.

In this study, we firstly propose and discuss a local restoration mechanism for Metro Ethernet networks using multiple spanning trees, which is distributed, fast, and does not need failure notification. Upon failure of a single link, the upstream switch locally restores traffic to pre-configured backup spanning trees. The mechanism can provide guaranteed protection with short recovery time and low signaling overhead, but requires high cost hardware implementation. A fast spanning tree reconnection (FSTR) mechanism which has much lower implementation cost to handle single link failure is then proposed. Under FSTR mechanism, a distributed failure recovery protocol is activated to reconnect the broken spanning tree using a *reconnect-link* not on the original spanning tree when a single link failure happens. To implement the protocol, failure notification and switching table reconfiguration are carefully designed. We further improve the FSTR mechanism by a novel approach to handle double-link failures. Finally, a two-layer architecture for survivability in Ethernet over WDM networks is presented.

The proposed mechanisms in this thesis require pre-configuration of spanning trees in the network and pre-allocation of backup capacity to provide guaranteed protection. We develop integer linear programming models (ILP)to formulate each proposed survivability schemes. We theoretically prove that the pre-configuration problem of local restoration and FSTR schemes are NP-Complete. We develop

several efficient algorithms, including two heuristics for local restoration mechanism and an augmentation based algorithm for FSTR mechanism. We demonstrate the effectiveness of the proposed survivability schemes through numerical results obtained from solving ILP models and simulation results on various network topologies and scenarios.

List of Symbols

(V, E) : a network with node set V and edge set E

K : number of established spanning trees

$\mathcal{D}$: traffic demand matrix

F : set of all failure scenarios

T^k : link set of spanning tree k

d : one connection in set $\mathcal{D}$

c_d : traffic amount of connection d

u_l : capacity on link l

o_d : origin of traffic demand d

t_d : terminal of traffic demand d

$o_{\vec{l}}$: origin of directional arc $\vec{l}$

$t_{\vec{l}}$: terminal of directional arc $\vec{l}$

P_d^k : path of connection d on spanning tree k

P_{ij}^k : path from node i to node j on spanning tree k

$P_{ij}^k(n)$: nth hop of path from node i to j on spanning tree k

$P_d(T, m)$: equals 1 if link m belongs to the path of connection d on spanning tree T, otherwise equals 0

$r_{\vec{l}}^{\vec{m}}$: reserved spare capacity on arc $\vec{l}$ for failure of directional arc $\vec{m}$

r_l : reserved backup capacity on bi-directional link l

r_l^m : reserved backup capacity on bi-directional link l for failure of bi-directional link m

$w_{\vec{l}}$: working traffic on directional arc $\vec{l}$

w_l^k : working traffic on bi-directional link l of spanning tree k

w_{lm}^k : working traffic traversing both bi-directional link m and l of spanning tree k

R_f^k : set of links which can be used to reconnect spanning tree k upon failure of link f

RP_{fl}^k : *reconnect-path* when link l is the *reconnect-link* to protect link f on spanning tree k

$T^k_{f_i,l}$: reconnected spanning tree with the reconnect link l upon failure of i

f_l : single link failure scenario of link l

f_{lm} : double-link failure scenario of link l and m

$\pi(f)$: stationary probability of a failure scenario f

A_d : protection grade requirement of connection d

List of Tables

List of Figures

Chapter 1

Introduction

Computer Networks which provide vast amount of information and resources greatly facilitating communications, business and studies have become indispensable in our lives today. Computer networks can be broadly classified into Local Area Networks (LANs) commonly covering a building or a university, Metropolitan Area Networks (MANs) commonly covering a metropolitan region, and Wide Area Network (WANs) covering as large as a whole country. In LANs, Ethernet, a family of frame-based computer networking technologies, plays the dominant role. After being originally developed by Xerox Corporation in early 1970s, Ethernet has evolved into the most widely implemented physical and link layer protocol in LANs. Nowadays, almost all computers are equipped with Ethernet cards, and more than 90% of data traffic starts or terminates at Ethernet LANs [1]. Moreover, the development of Ethernet keeps on accelerating even though a lot of replacement

network technologies have emerged.

Recent progress on Ethernet technology paves the way for its further deployment in MANs, termed Metro Ethernet networks. A Metro Ethernet network is based on Ethernet technology and covers a metropolitan region. Ethernet protocols to support metro and wide are network services have been and are being standardized, such as IEEE 802.1Qay which lets network operators define the end-to-end path in Ethernet networks; IEEE 802.1ah which resolves the scalability problem of Ethernet when it is used in large networks, and IEEE 802.1ad which provides more VLAN to support various services. In hardware, 10Gbps Ethernet switch is available off-the-shelf, and 40 Gbps Ethernet switch is under development. Compared with the traditional technologies deployed in MANs, Metro Ethernet technology has the advantage of economy of scale, ubiquity, high-bandwidth, ease of configurations, support on various network layer technologies and scalability [2].

In Metro Ethernet networks, survivability which is the ability to continuously transmit traffic even when parts of the network has failed is strongly required due to the fact that a failure of a network component in MANs would result in serious data traffic disruptions [3]. However, traditional Ethernet lacks survivability. With more and more critical data services provided in metropolitan regions, it is necessary to design fast, reliable and efficient survivability schemes for Metro Ethernet networks.

In the subsequent sections of this chapter, Metropolitan Area Networks and traditional Ethernet technology will be introduced at first, followed by an overview of Metro Ethernet networks. Then the research objectives of the thesis are presented.

1.1 Metropolitan Area Networks

Metropolitan Area networks are large computer networks usually spanning a city, typically consisting of thousands or millions of hosts and servers and are established over optical fibers [4]. A MAN covers a geographical area larger than a LAN, but smaller than a WAN. A MAN is structured as a set of interconnected LANs that work together in order to provide access and services within a metropolitan region. It is also the first span of the network that connects subscribers to the WAN [5]. Current MANs can provide various network services, e.g., Internet Connectivity, Transparent LAN, Virtual Private Network (VPN), and Voice over IP (VoIP) services.

SONET (Synchronous Optical Networks) is the most widely used protocol in MANs, which was originally introduced by Bell Labs in 1985 [6]. SONET is a synchronous system controlled by a master clock. The basic unit of framing in SONET is a STM-1 (Synchronous Transport Module level-1), which operates at the rate of 155.52 Mbps. Each frame consists of two parts, the transport overhead

and path virtual envelope. Data encapsulated in the path virtual envelop is transmitted by frames on an established end-to-end path. Since SONET is a connection oriented network and originally designed for voice and lease line services, it is not applicable for emerging date applications.

Asynchronous Transfer Mode (ATM) is another technology used in MANs. ATM, as a standardized digital data transmission technology, is implemented as a network protocol. It was first developed in the mid 1980s [7]. ATM is a cell-based switching technique that uses asynchronous time division multiplexing. An ATM cell consists of a 5-byte header and a 48-byte payload. ATM has properties from both circuit switched and small packet switched networking. It uses a connection-oriented model and establishes a virtual circuit between two endpoints for transmission. With fixed sized cell switching and virtual circuit establishment, ATM can support voice services as well as packet based services, making it suitable for MANs. It also can realize various QoS and traffic engineering functions, such as shaping, policing and admission control. However, ATM is in the process of being displaced by Ethernet-based technologies due to its high cost and complicated management.

1.2 Switching Ethernet Technology

1.2.1 Switching and Bridging

Ethernet was originally designed for LANs which has a shared medium architecture. With the increase of the network scale, bridges and switches are developed to improve the network transmission efficiency and reliability. Bridging and switching are created to communicate at the data link layer while isolating the physical layer. Bridges are used to connect two Ethernet segments, while switches can connect multiple Ethernet segments. With them, only well-formed Ethernet packets are forwarded from one Ethernet segment to another. Meanwhile, collisions and packet errors are isolated. Bridges and switches learn where devices are by examining MAC addresses of incoming packets before they were either dropped or forwarded to another segment, and do not forward packets across segments when they know the destination address is not located in that direction. Finally, switching and bridging make Ethernet a store-and-forward network called Ethernet switched network, where the frame in the networks would be read into a buffer on the switch in its entirety, verified against its checksum and then forwarded according to its MAC address.

Unlike IP networks which require complex signaling structure to exchange information among each station and establish routing tables, Ethernet switch or

bridge maintains and self-learns a forwarding table for frame forwarding. The forwarding table is used to map a destination MAC address to an output port. An incoming Ethernet frame looks up the forwarding table and is switched to the corresponding output port of a switch. If the frame cannot find an output port, it is broadcasted to all the downstream links, which is called flooding. The frame may be flooded all over the network until it reaches the destination. If the source of the frame is not listed in the forwarding table of the switch, the switch creates an entry and registers the MAC address. In future, it will forward all the frames with this MAC address to the corresponding output port. This procedure is called backward learning. Since Ethernet frames do not have a time to live (TTL) field, to avoid deadlock during flooding phase, Ethernet frames must be transported in a topology without any loops.

1.2.2 IEEE Spanning Tree Protocol

Current Ethernet switched networks use a spanning tree protocol family to yield loop free topology, including IEEE 802.1d Spanning Tree Protocol (STP) [8], IEEE 802.1w Rapid Spanning Tree Protocol (RSTP) [9] and IEEE 802.1s Multiple Spanning Tree Protocol (MSTP) [10].

Spanning Tree Protocol

IEEE 802.1d Spanning Tree Protocol (STP) [8] is a link layer network protocol that ensures a loop-free topology for any bridged LAN and switched Ethernet networks. The spanning tree computed by Ethernet switches using STP is a root based shortest path spanning tree, which means the path between root switch and any other switch on the spanning tree is the shortest one. To build the spanning tree, every switch in the network broadcasts its unique identifier, and a switch with the smallest identifier is selected as the root switch. Each switch then computes the cost of all its possible paths to the root, selects the least cost path, and sets the port connecting to that path as the *root port*. Each switch also computes the path cost to the root from any other switch. The switch with the least cost path is selected, and the path connecting to the switch is set as the *designated port*. Bridge Protocol Data Unit (BPDU) is used for STP to exchange identifiers of each switch and path cost.

With STP, there are five possible port states for a port of an Ethernet switch. In the blocking state, data frames will be blocked but BPDUs can still be received and processed. In the listening state, data frames are still blocked, the port can receive and send BPDUs. It waits for new information to cause it return to blocking state. In the learning state, the port does not forward frames but learns source addresses of frames and adds them to the forwarding table. In the forwarding state, the port

receives and sends data frames. In the disable state, BPDU and data frames are all discarded.

The problem of STP is its slow convergence speed after spanning tree topology change resulting from link failure, switch failure or the addition of a link or a switch. Upon a topology change, all the designated switches intervened by the change would send BPDUs to the root switch. After being notified, the root switch broadcasts the topology change messages to all the designated switches. The designated switches receive the message, then wait until the root switch clear the topology change. In the end, all the designated switches resume learning and forwarding operations. The problem is during the re-convergence, traffic would be disrupted by twice the forwarding delay. Commonly, the spanning tree re-convergence would take about tens of seconds after a topology change.

Rapid Spanning Tree Protocol

IEEE 802.1w Rapid Spanning Tree Protocol (RSTP) [9] is based on STP with backward compatibility. It is developed to provide fast recovery of spanning trees from topology change and can be regarded as an evolution of STP.

RSTP increase the speed of spanning tree re-convergence after the topology change in several aspects:

- A switch monitors the change of link states at each of its port. It generates

topology change message upon any change of link status. Hence, failure detection can be done in 3 hello time.

- Add new port designations, *alternate port* and *backup port*. The *alternate port* is the backup of the *root port*. For any failure that makes the *root port* disabled, The *alternate port* is quickly used to play the role of new *root port*. The *backup port* is the backup of the *designated port*.

- In RSTP, a switch will respond to BPDUs sent from the direction of the root bridge, called sync operation. As soon as the switch receives a BPDU from another switch in the direction of the root and determine that this is a superior root information, it changes its other ports to discarding state. It then negotiates with the switch who sends the BPDU and conforms the spanning tree information received. The switch who sends the BPDU can rapidly change that port connecting with the first switch from the discarding to the forwarding state bypassing the traditional listening and learning state transition. This creates a cascading effect from the root bridge where each designated bridge handshakes to its neighbors to determine if it can make a rapid transition.

Spanning tree re-convergence time with RSTP is much faster than STP, but still can be as long as several seconds which contributes to two major problems [11]. When a root bridge failure happens, RSTP may take several seconds to converge

(5s). The count-to-infinity behavior can occur when the root fails and the resulting reconfiguration holds a loop [12]. To prevent loops, RSTP negotiates every port transition, and port negotiation may result in large reconfiguration delays if the root bridge is involved in the failure.

Multiple Spanning Tree Protocol

Both STP and RSTP are based on a single spanning tree, which prohibit many links in the network. Per VLAN Spanning Tree Protocol (PVST) is developed by Cisco to fully utilize the network resources, in which multiple spanning trees are established in the network, and each spanning tree is given a dedicated Virtual LAN ID. By tagging different VLAN tags in the edge node, frames could be transported over different spanning trees. IEEE 802.1s Multiple Spanning Tree Protocol (MSTP) [10] is the evolution of PVST. Besides building multiple spanning trees, MSTP also allows multiple VLANs to be mapped to one spanning tree, making the protocol more flexible than PVST.

The strength of MSTP is that it can provide better network resource utilization and network connectivity than STP and RSTP. Traffic engineering can be realized by using MSTP. The MSTP can provide multiple paths between each node pair, and thus is more flexible to provide guaranteed QoS tunnels and load balancing in the network. However, it is not a trivial task for MSTP to configure and manual

configuration has to be used to properly configure all the Ethernet elements. Compared with the routing algorithm in MPLS and IP networks, multiple spanning trees based routing algorithms still lack flexibility. Moreover, since an Ethernet switch cannot support too many spanning trees due to the cost and complexity of Ethernet switch (commonly less than 64), how to provide services with a limited number of spanning trees is still challenging.

1.2.3 IEEE Virtual LAN Protocol

A virtual LAN, commonly known as VLAN, is a set of hosts which are located at different parts of the network, but communicate in a way as if these hosts are in the same physical LAN.

The protocol commonly used today to configure VLAN is IEEE 802.1q [13]. Frames are tagged with VLAN information. 802.1q uses a frame-internal field for tagging, so frame modification is needed. The IEEE 802.1q header contains a 4-byte tag header including a 2-byte tag protocol identifier (TPID) and a 2-byte tag control information (TCI). TCI field contains 3 bit priority field which is also known as Class of Service (CoS) field, 1 bit canonical format indicator, and 12 bit VLAN ID. A VLAN ID is added to a frame when they enter into the Ethernet switched network, then the frame can only be transmitted among the members of the assigned VLAN. The frame cannot be sent to a host not belonging to the

VLAN unless its VLAN ID is changed or the host participates in the VLAN. VLAN assignment can be static or dynamic. Static VLANs assignment is also called port based VLANs in which a port is assigned to a VLAN, and all the hosts attached to a port should be members of the same VLAN. Dynamic VLAN assignment can dynamically map different VLANs to a port.

With the development of Ethernet technologies and increase of network scale, 802.1q suffers from the scalability problem. It is because traditional 802.1q protocol only has 12 bit to represent the unique VLAN ID for each VLAN. However, this is not enough in large scale networks especially when Ethernet is deployed in provider networks. Hence IEEE 802.1ad protocol [14], as an amendment of 802.1q, is developed to resolve the scalability problem. The main idea is to add a new VLAN field. A provider network stack its own VLAN ID outside the original VLAN ID of the frames. Therefore, frames are transmitted using providers VLAN ID in the provider network without knowing the internal VLAN ID. The protocol is also called Q-in-Q protocol, and it can resolve the scalability problem of Ethernet to some extent.

1.3 Metro Ethernet Networks

Metro Ethernet technology has become an integral element of todays telecommunications industry. Metro Ethernet, with its sophisticated carrier-class features,

has significantly transformed the telecom industry and has enabled the industry to provide numerous services to vast areas at a lower cost. A comprehensive global report on Metro Ethernet markets released recently predict that Global Metro Ethernet market is projected to reach 19.5 billion US dollar by the year 2015. At present, there are mainly three proposals of Ethernet deployment in MANs: Ethernet transport network, Ethernet over SONET and Ethernet over MPLS. The first one uses Ethernet switches in metro networks, thus no translation between protocols is needed. Ethernet over SONET is to transport Ethernet connection over SONET networks, which is due to the fact that SONET have been partly deployed and are being displaced. Ethernet over MPLS is to use MPLS switch in metro region, and Ethernet frames are encapsulated in MPLS labels at the ingress nodes of the network.

1.3.1 Pure Ethernet MANs

The architecture of pure Ethernet MANs is shown in Fig. 1.1, which is an extension of the native Ethernet into metro networks [2]. The network comprises Ethernet switches/bridges, including Access Points (APs) which connect LANs, and Core Ethernet Switches (CESes). APs take responsibility for managing frames that enter or leave the Metro Ethernet, while CES simply forwards frames in the Metro Ethernet networks. A spanning tree protocol is used to establish one or more trees

that span all APs. Each tree provides a path between all the customer sites in the virtual LAN (VLAN) of that customer. Encapsulation schemes are used to address the scalability problems of deploying Ethernet in the metro domain. Due to the limited number of VLANs that can be supported (limited to 12bits), an additional Q-tag can be inserted by the metro provider into the customer Ethernet frames at the APs of the metro domain, referred to Q-in-Q or VLAN stacking. Another encapsulation scheme resolves the MAC address table explosion problem, since it is impossible for a core switch to learn all customer MAC addresses. The scheme is called MAC-in-MAC, in which each ingress node inserts two additional MAC address fields. These fields are the source and destination MAC addresses that correspond to the respective APs and have local significance within the metro domain. In this way, core switches only need to learn the edge switch MAC address, and huge end-user MAC address entries are avoided in the core switches.

Metro Ethernet Forum (MEF) has defined two types of services for pure Ethernet MAN based on the network connectivity [15] : Ethernet Line (E-Line) and E-LAN services. The E-Line service provides a point-to-point Ethernet Virtual Connection (EVC) between two APs, while E-LAN service can provide multi-point connectivity.

The shortcomings of pure Ethernet MANs mainly come from its scalability, traffic engineering and survivability. Scalability problem can be alleviated by using Q-in-Q and MAC-in-MAC schemes, but Ethernet still cannot compare with

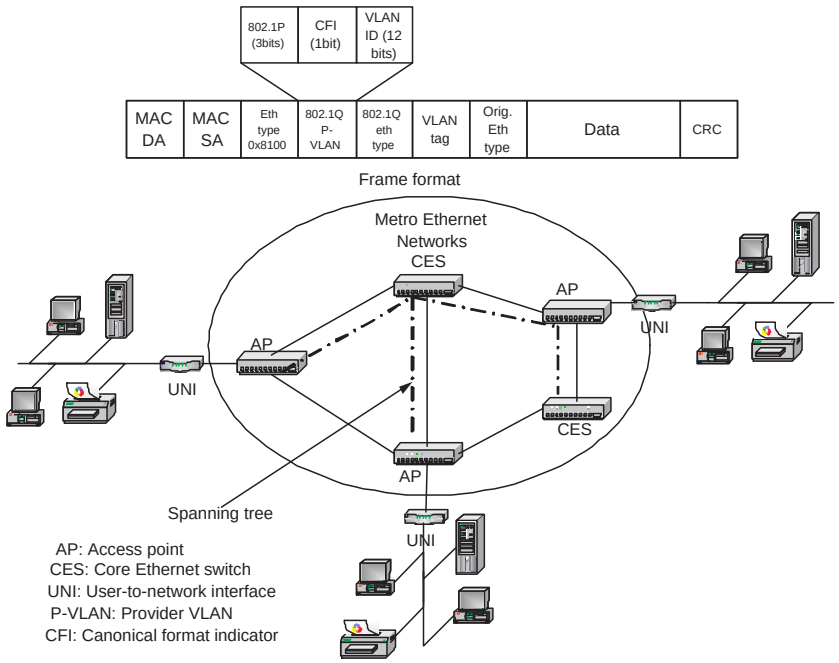


Figure 1.1: Pure Ethernet MANs Architecture

MPLS in which labels are local variables. The traffic engineering that Ethernet can provide is also limited. Spanning Tree Protocol (STP) used in Ethernet can only construct one single spanning tree in the network, and each node pair has only one path for transmission. Many links in the network would be prohibited, which is not cost effective and bandwidth efficient. Multiple Spanning Tree Protocol (M-STP) can utilize as many links as possible in the network. In addition, for the purpose of load balancing and survivability, some advanced forwarding technologies which do not use spanning tree solution and can provide more connectivity to Ethernet networks have been developed, including SmatrBridge [16], STAR [17], and Turn Prohibition Switch [18].

1.3.2 SONET/SDH Ethernet MANs

An SONET/SDH based Ethernet MAN is commonly regarded as an intermediate step in the transition from a traditional, time-division based, and connection-oriented network (SONET), to a modern statistical network (Ethernet) [19]. In this model, the existing SDH infrastructure is used to transport high-speed Ethernet connections. The main advantage of this approach is that SONET could provide high survivability achieved through the use of the native SONET protection mechanisms which can handle network component failures within a recovery time of 50 ms. On the other hand, an Ethernet over SONET is more expensive, due to the high cost of SONET devices. Traffic engineering also tends to be very limited, since SONET as an infrastructure of the Ethernet can only provide connectivity with a certain capacity.

Further, when Ethernet is deployed over SONET, different traffic granularity between the two protocols would cause the inefficiency of bandwidth utilization [20]. For example, the closest hierarchy to Gigabit Ethernet (1GB) is STS-48 (2.5GB), which leads to a 60%wastage of bandwidth. An approach that can solve this problem is to use *Virtual Concatenation* protocol. Instead of using single high bandwidth demands, the source node can split traffic into several low bandwidth demands and transmit them using multi-paths. The destination reconstructs the data stream back. If *Virtual Concatenation* protocol is used, for the above example

a 1G demands can be mapped to 7 STS-3s, imposed only 8% overhead. The problem here is that delay of different paths would be different, and the large differential delay between paths that would require large buffer at destination to reconstruct the original traffic stream. How to find paths with minimum differential delay has been proved to be NP-Complete.

Another challenge in Ethernet over SONET is the construction of spanning tree for bandwidth guaranteed services. Unlike the traditional minimum spanning tree protocol which run in polynomial time, to find a spanning tree that can carry traffic demands between all node pairs is proved to be NP-Complete [21]. Some researchers propose to split the so called BEST problem into two subproblems: first find the virtual tree satisfying all bandwidth requirements and then find feasible physical routing for the tree. This problem is not unique for SONET networks, and other transport networks also have the same problem to find a bandwidth guaranteed spanning tree. In addition, although creation of single bandwidth guaranteed spanning tree has been considered, how to construct bandwidth guaranteed multiple spanning trees is still a challenging issue.

1.3.3 MPLS Based Ethernet MANs

An MPLS based Metro Ethernet network uses MPLS in the Service Provider Network to carry Ethernet connections. Multiple Protocol Label Switching technology

is an evolution from ATM technology. MPLS has so far been deployed to solve various backbone network problems, such as integrating IP and ATM networks, reducing IP Router overhead, and solving route propagation problems. MPLS generally has connection-oriented, path-switching capabilities. In MANs, its most obvious advantage comes from creating guaranteed and secure service capabilities [22]. In MPLS based Ethernet MANs, the customer's Ethernet packet is transported over MPLS and the service provider network uses Ethernet again as the underlying technology to transport MPLS.

MPLS has been well developed for MANs and WANs. It can provide protection, guaranteed QoS, and traffic engineering. Some researchers have address the question [4]: since we have MPLS, why we need Ethernet directly over metropolitan network?

Compared with MPLS, Ethernet has several strengths: (1) Ethernet is plug-and-play, we do not need to configure them by hand, also the management of Ethernet switch is much simpler than MPLS switch. (2) In the view of business, Ethernet switch is much cheaper and more ubiquitous.(3) Ethernet is a layer 2 service, while MPLS is always regarded as layer 2.5 service. It is unreasonable to have a layer 2 service over layer 2.5 service if we could achieve metropolitan network services requirement by changing the present Ethernet protocols.

1.4 Network Failures and Survivability

A network can fail at any time due to the unexpected events, such as nature disasters, maintenance, or power cuts. In MANs where the physical infrastructure commonly consists of copper wires and optical fibers housed in cables and then embedded underground or underwater, a failure could be more frequent and harder to repair in a short time. A statistical study on backbone network [23] shows that network failures are part of every day operation and affect many links. It also indicates that the main causes of failure come from maintenance operations, routers down and fiber cuts. Data on Bellcore networks [24] presents a fiber cut rate of 4.39 cust/year/1000 sheath lines, which means more than one fiber cut every day on a typical long-haul network and one fiber cut every four days on a typical MAN. Due to the high line transmission rate on MANs, network failures especially fiber cuts can lead to serious traffic disruptions and outages of various network services, which further bring about huge financial losses and significant social impacts.

Network survivability is defined as the ability to maintain or restore an acceptable level of performance during network failures by applying various protection and restoration techniques, and prevent or mitigate service outages from network failures [25]. Various survivability techniques can be implemented at different layers, including physical layer, system layer, logical layer and service layer [26].

Survivability techniques at different layers use different resources and apply different survivability strategies, which will be reviewed in Chapter 2. According to the layering definition of survivability schemes, the mechanisms we propose and study in this thesis belong to logical layer survivability techniques.

1.5 Research Objectives and Scope

In MANs, it is a critical requirement to have a fast, reliable and efficient failure handling mechanism to support carrier-grade services, e.g. Virtual Private Network (VPN), Voice over IP (VoIP) and Video Conference. Unfortunately, current Ethernet switched network uses spanning tree protocol family without fast failure recovery mechanism. Research gaps for current study on resilient schemes for Metro Ethernet networks are summarized below:

- The survivability mechanism's capability for reliability and fast recovery in Metro Ethernet networks should be similar, if not more stringent than those of SONET which have sub-50 ms switchover capability in case of a link failure. However, none of the traditional Ethernet protocols can achieve the objective. The IEEE 802.1d Spanning Tree Protocol (STP) suffers from long spanning tree convergence time of 30 to 60 seconds and inefficient bandwidth usage in case of failures. The IEEE 802.1w Rapid Spanning Tree Protocol (RSTP) can provide faster convergence by resolving the main drawback in

STP. However, the recovery time for RSTP is in the order of seconds. It is still not applicable for large scale networks with hundreds of switches.

- Since traditional Ethernet networks are connectionless networks, spanning tree protocol family can only provide network connectivity without any guaranteed bandwidth for Ethernet connection. In case of a failure, the spanning tree protocol family has no mechanism to provide guaranteed backup capacity for the traffic affected by the failure after spanning tree reconvergence. To satisfy the requirements of Quality of Service (QoS) and Protection of Service (PoS), survivability schemes with guaranteed backup capacity provisioning are necessary.

The main aim of this study is to propose several distributed survivability mechanisms for Metro Ethernet Networks. The specific objectives of the research are to:

- Propose a local restoration mechanism with multiple spanning trees for Metro Ethernet networks to handle single link failures. The local restoration mechanism has features of fast recovery speed and guaranteed protection.

- Propose a Fast Spanning Tree Reconnection mechanism for single link failures in Metro Ethernet networks, which could be used on both single spanning tree and multiple spanning tree scenarios. The mechanism has the same features as local restoration mechanism with lower implementation cost.

- Study the performance of Fast Spanning Tree Reconnection mechanism when handling double link failures in Metro Ethernet networks. With mechanism, a new model is used for network configuration.

- Study the survivability schemes of Metro Ethernet networks over WDM optical networks. Develop a two layer architecture for survivability in Ethernet over WDM and Fast Virtual Spanning Tree Reconnection mechanism.

The proposed resilient mechanisms are distributed and can guarantee the bandwidth for protection. The proposed mechanisms may provide insights on future survivability schemes for Metro Ethernet networks and Ethernet protocol standardizations. It can solve the survivability problem in Metro Ethernet that traditional Ethernet technology is unable to solve, maintaining Ethernet's features of plug-and-play. It is possible that these mechanisms could be implemented in current Ethernet switch with some modifications for Metro Ethernet deployment and commercialization.

Our study assumes pure Ethernet MANs model in which transport networks are built by a set of Ethernet switches and various spanning tree protocols are implemented. Our thesis focuses on link failures in Metro Ethernet networks, including single link failures and double link failures. We also consider the single fiber cuts in Ethernet over WDM networks which can lead to multiple logical link failures at Ethernet layer. We note that link failures are the most common and

serious failure scenarios in Metro Ethernet networks. We also note that the network traffic load in this study is restricted to static traffic which does not change abruptly over time. Static traffic pattern assumption is widely used in optical and MPLS networks when they are used in metro and wide area networks because the traffic at these scales typically vary quite slowly over time and is less dynamic in nature. Traffic is commonly transmitted in optical fiber in Metro Ethernet networks, so we believe that static traffic pattern assumption is also valid in Metro Ethernet networks. The survivability schemes for Metro Ethernet networks for dynamic traffic are beyond the scope of this study.

1.6 Thesis Outline

The remainder of the thesis is organized as follows:

In **Chapter 2**, we review the survivability techniques in traditional MANs and the emerging survivability schemes for Metro Ethernet networks.

In **Chapter 3**, we propose a local restoration mechanism for Metro Ethernet networks using multiple spanning trees. We propose two approaches, connection-based and destination-based, for Ethernet frames to select backup trees. The problems of working spanning tree assignment and backup spanning tree configuration are formulated, and we prove that the problem is NP-Complete. We also develop heuristics for each strategy to reduce the computational complexity.

In **Chapter 4**, we propose a Fast Spanning Tree Reconnection (FSTR) mechanism for Metro Ethernet networks to handle single link failures. We present the details of the protocol, including failure notification and forwarding table reconfiguration procedures. The issues of the *reconnect-link* pre-configuration to reconnect each spanning tree are addressed. We prove that the pre-configuration problem is NP-complete, and propose an efficient augmentation based algorithm that can obtain close approximation to the optimal solutions.

In **Chapter 5**, we address the issues of handling double-link failure using Fast Spanning Tree Reconnection mechanism. We give and prove the loop avoidance condition under double-link failures, and analyze the protection grade that can be provided.

In **Chapter 6**, we introduce a two layer model for Ethernet over WDM networks, and develop Fast Virtual Spanning Tree Reconnection (FVSTR) mechanism which optimizes the network resources by coordination of lightpath routing at the optical layer and FVSTR configuration at Ethernet layer. We present an optimization solver to compute lightpath routing and FVSTR configuration.

In **Chapter 7**, we summarize and conclude the contributions in the thesis, and suggest several future research directions.

Chapter 2

Background and Related Work

Restoration methods are broadly classified into reactive and pro-active methods [27]. The reactive method finds a way to recover after a failure. When a network component fails, a search mechanism is activated to find a new path to bypass the failed network component. The reactive method has low overhead, since no resources need to be reserved before the failure. However, it would take longer recovery time and cannot guarantee the survivability. Pro-active method, on the other hand, pre-allocates redundant resources for survivability in network. Obviously, a pro-active method can guarantee the survivability with smaller recovery time, but requires more network resources.

Commonly survivability techniques can be classified into protection, dynamic restoration and recovery [28]. Protection techniques are fully pro-active methods, which establish and provide primary and backup paths a priori for connections.

Dynamic restoration techniques also establish the primary and backup path in advance, but only provide backup path for each connection upon a failure. Hence restoration is regarded as partial pro-active methods. Recovery techniques are pure reactive methods. They find the backup path for a connection only after a failure occurrence.

2.1 Survivability Techniques in MANs

2.1.1 Survivability Techniques in SONET

Currently, there are mainly three protection techniques defined in SONET networks: Automatic Protection Switching (APS), Unidirectional Path-Switched Ring (UPSR), and Bidirectional Line-Switched Ring (BLSR).

Automatic Protection Switching (APS)

APS allows a pair of SONET links to be configured for line redundancy. In the event of a fiber cut, the active line switches automatically to the backup fiber within 60 milliseconds, including 10 ms initiation and 50 ms switch-over. It involves four optical fibers, two working fibers (one in each direction) and two protection fibers. In linear APS 1+1 protection, data traffic is carried by working fibers and protection fibers at the same time. Only when a cut on a working fiber occurs, the

receiver receives the traffic on the corresponding protection fiber. Linear 1:1 APS is an evolution of 1+1 APS, in which protection fibers carries the protected traffic upon failure of working fibers only. In the normal operation time, low priority traffic can be transmitted on the protection fibers, which improves the network resources utilization. Other extensions of APS are 1:N and k:N APS where N working fibers share the redundancy brought from one or k protection fibers.

Unidirectional Path-Switched Ring (UPSR)

Ring based transmission systems are an evolution from APS. Nodes in the ring are connected by working fibers and protection fibers in a closed loop. In UPSR, each connection is transmitted on both working fibers in one direction around the ring and protection fibers in the opposite direction. Upon a fiber cut on working fibers, the receiving node monitors the signal loss and switch to receive data from protection fibers. Protection switching decisions are made for each path but not for the whole link. UPSR can provide a protection switching time of 50ms to SONET ring networks. Since traffic demands are transmitted on every fiber in UPSR, the total demand carried on one fiber equals to the sum of all demands between all node pairs around the ring. Hence the line transmission rate of UPSR must be greater than the total demands carried by the ring.

Bidirectional Line-Switched Ring (BLSR)

BLSR is more efficient than UPSR and APS. Unlike UPSR which requires data transmission on both working fibers and protection fibers in a closed loop, 4-fiber BLSR uses a pair of bidirectional fibers for working and protection. The bidirectional working fibers and protection fibers form two separate cycles, and traffic is only transmitted on working fibers during normal operation. Upon a failure, traffic is restored to the opposite direction from the working fiber to the protection fiber at the node adjacent to the failed network component. 2-fiber BLSR is similar to UPSR in fiber layout, but uses the same mechanism as 4-fiber BLSR. Working traffic is transmitted on one direction and restored to the opposite direction at failure neighbor. BLSR commonly uses the shortest path on the ring to carry working traffic and allows reuse of working capacity around the ring. Further protection bandwidth can also be shared among different traffic demands.

2.1.2 Survivability Techniques in ATM and MPLS

ATM and MPLS networks commonly adapt restoration methods which have more flexibility and efficiency than protection techniques in SONET attributing to the fact that they can create redundant paths according to real traffic transmitted in the networks. Restoration techniques can be broadly classified as link-based restoration, path-based restoration, and local restoration [29].

Link-Based Restoration

In link-based restoration, a backup path is established between two end nodes of a link which reroutes traffic locally upon the link failure. In case of a link failure, the traffic on the link is rerouted at the upstream node of the failure to the backup path, bypasses the failed network component and finally is sent back at the downstream node of the failed link. Backup paths for the links along the primary path are chosen during the connection set-up period. The recovery time of link-based restoration is fast since the failure is detected and recovered locally.

Path-Based Restoration

In path-based protection, a backup path for each connection is set up which is link or node disjoint to the working path of the connection. The backup path restores the failed traffic between the two end nodes of the working path affected by the failure. Path restoration can reuse the surviving working capacity of the failed paths if "stub-release" can be supported. Stub-release implies that the capacity on the failed working path can be released after the failure to carry other traffic demands. Hence, stub-release improves the efficiency of the path restoration and makes it a failure specific mechanism since the restoration response depends on the specific failure scenario.

Shared Backup Path Protection (SBPP) is a path-based restoration scheme

standardized by IETF which is amenable to IP-centric control. A 1:1 APS system is established at the level of each service path. Capacity on backup paths must be pre-reserved for guaranteed protection. Spare backup capacity of failure-disjoint paths can be shared, and spare backup capacity of link disjoint working paths can also be shared. Such sharing greatly increase the capacity utilization. Compared to the link-based restoration , a SBPP system often provides much better spare capacity sharing efficiency due to its end-to-end protection and a wider range of spare capacity sharing. Typically, for a mesh network, a link-based restorable network can have spare capacity efficiency of about $50-70\%$, while a SBPP network can achieve spare capacity efficiency of about $30-40\%$ [30]. On the other hand, the recovery time of SBPP is much longer than link-based protection. Segmentation-based protection mechanisms have also been proposed to provide a tradeoff between link and path protection, where protection is provided at sub-path level [31].

Local Restoration

MPLS local restoration (also called MPLS fast reroute) uses a feature of RSVP Traffic Engineering to provide fast and guaranteed protection in MPLS networks [32][33]. In MPLS local restoration, a backup path is pre-set for each link or node along the primary path. The backup path starts at the immediate upstream node of the protected component and ends at the tail of the primary path (local recovery) or at the immediate downstream node (restricted local recovery) [34]. Upon

failure, the first upstream node from the failure should switch the traffic to the corresponding backup path. The MPLS local restoration provides faster recovery because the decision of recovery is strictly local, and it can meet the requirements of real-time services with recovery time comparable to those of SONET rings (within 50 ms). It can support backup capacity sharing schemes similar to SBPP. The capacity utilization of local restoration is better than link-based restoration, since each working path has its own set of backup paths and these backup paths are established with more flexibility.

2.1.3 Survivability Techniques in Connectionless Networks

Unlike in MPLS networks, restoration in connectionless networks, e.g. IP networks and Ethernet, cannot use established detour paths. Some researchers have studied local restoration in IP networks which is also called IP fast reroute. The schemes aim at integrating local recovery with the traditional IP routing protocol, and solve the low speed routing table convergence problem. In [35], a multiple routing configuration method is proposed for failure recovery. On each routing configuration which is built at the network initiation phase, traditional IP routing algorithms are run to establish the routing tables. The primary configuration is used to transmit traffic during normal operation. On each backup routing configuration, a set of links or nodes are blocked, hence the failure of these links or nodes can

be recovered by using this backup routing configuration. When a failure happens, instead of running the routing table convergence algorithm, routing would use the corresponding backup routing configuration. The method in [36] exploits the characteristic of Border Gate Protocol, and improves it to provide sub-50ms second fast recovery. The method also involves pre-configuration in IP layer and sets up backup BGP ports at each router.

2.2 Survivability Techniques in Metro Ethernet Networks

The survivability issues for Metro Ethernet networks are different from those on other network architectures due to the spanning tree topology in Ethernet networks. Spanning tree is 1-connected, so any link or node failure will make the network disconnected and require spanning tree re-convergence. The tree based restoration mechanism was first proposed in [37]. In tree-based protection, each source node establishes two spanning trees, red tree and blue trees. The two trees are not necessarily to be link disjoint, however, they satisfy the condition that any node is reachable from the source node via at least one tree after any single link failure. This tree-based protection mechanism is originally designed for MPLS networks, but not for Metro Ethernet networks. The two trees generated by each

source node cannot be used by other source nodes. However, in Ethernet networks, all the nodes can share the same spanning tree. With the development of Ethernet protocol and switches, more survivability schemes have been proposed for Metro Ethernet based on a single spanning tree and multiple spanning trees.

2.2.1 Schemes on Single Spanning Tree

Several mechanisms to reduce the recovery time in Metro Ethernet based on single spanning tree have been suggested by modifying IEEE Spanning Tree Protocol and IEEE Rapid Spanning Tree Protocol.

Ethernet Automatic Protection Switching (EAPS) [38]

Ethernet Automatic Protection Switching (EAPS) [38] is Extreme Networks solution for fault-tolerant Ethernet ring topologies. A switch is designated as the master, and one of its two ring ports is designated as the primary port and the other as the secondary port. The Master switch blocks the secondary port for working traffic, thereby avoiding a loop on the ring. The master switch periodically sends poll packets from its primary port and receives them from the secondary port to ensure no failure on the ring. As long as a failure happens, the master switch detects the failure from poll packets and unblocks the secondary port. The traffic then is restored from the secondary port and the failure recovery time can

be in the region of 50ms. Ethernet Protection Switching Ring (EPSR) [39] and Rapid Ring Protect Protocol (RRPP) [40] are similar to EAPS in principle, but differentiate on failure detection and notification. These technologies can provide sub-50m second recovery for Metro Ethernet, but are constrained on ring or multi-ring topologies.

Fast Failure Recovery Spanning Tree (FFRST) [41]

Fast Failure Recovery Spanning Tree (FFRST) approach proposed in [41] builds a master tree and a set of sub trees for each node in advance. Each Ethernet switch maintains a list to store the information of the sub trees if the particular switch is the root of the sub trees. Traffic is transmitted on the master tree during normal operation. Upon a network component failure, the switch adjacent to the failure detects it, and uses the corresponding sub tree to replace the failed part of the master tree. The switch then broadcasts the reconfiguration message over the sub tree, asking other switches to change their configuration of the spanning tree. The re-convergence of the whole master tree is avoided. This approach is improved in [42] by building a protection sub-tree for each switch rooted at itself. Hence the improved mechanism can support backward compatibility with the legacy Ethernet switch. However, both of the above works did not address the problem of how to provide guaranteed backup capacity. In addition, the traffic demands are transmitted only on a single working spanning tree even though multiple redundant trees

are established.

Optimal RSTP ($RSTP_{opt}$) [43]

A tool for RSTP optimization is introduced in [43], so that protection of the traffic in Metro Ethernet is guaranteed. An integer linear programming model is developed to minimize the bandwidth required for protection. Upon any single link failure, the network uses RSTP to re-converge the spanning tree. Redundant capacity on each link is reserved to carry additional traffic after failure. However, since the mechanism is based on RSTP, failure recovery time is not shorter than the traditional RSTP protocol which is in order of seconds.

EPOCH [12]

In [12], the drawback of RSTP which leads to slow spanning tree convergence when switch failure near the root happens is investigated. It is found that too many negotiations upon failure of a root switch in the network would result in count-to-infinity problem which takes RSTP tens of seconds to rebuild the spanning tree. Eimeleegy in [12] proposes EPOCH, a modification of RSTP, which is backward compatible and can reduce the spanning tree re-convergence time upon a root failure from tens of seconds to milliseconds. However, since EPOCH is still based on RSTP, it only increases the spanning tree re-convergence speed upon root failure, but does not resolve the problem that spanning tree needs to be recomputed after

the failure. The spanning tree re-convergence time under EPOCH has no upper bound and still takes several seconds in some cases.

2.2.2 Schemes on Multiple Spanning Trees

Due to the slow convergence and low capacity utilization of the single spanning tree protocol, resilience mechanisms based on IEEE 802.1s Multiple Spanning Tree Protocol (MSTP) have been proposed for Metro Ethernet.

Viking [44]

Viking is a promising large scale Ethernet system which uses multiple spanning trees for protection and load balancing. In Viking, a central scheduler is needed to make decisions on how to construct multiple spanning trees and how to distribute traffic onto spanning trees based on long term traffic matrix. When a switch detects a failure, it informs the central scheduler. The central scheduler then figures out the affected traffic flows and informs the corresponding ingress switches to use pre-configured end-to-end backup paths on other spanning trees. The traffic affected by the failure is safely transmitted on the new spanning tree, bypassing the failed network component. The system has fast recovery speed (positive) within 1 second, and it can efficiently utilize the network resources. An optimization model based on Viking [45] has also been proposed using integer linear programming

formulation. Instead of the heuristic algorithm proposed in Viking, the model provides an optimal solution on the mapping of primary and backup paths to multiple spanning trees. The solution from the model has 10% better bandwidth utilization than Viking system.

Fast Failure Handling (FFH) [46]

Farkas proposes a distributed resilient architecture for Metro Ethernet named Fast Failure Handling (FFH) approach. With this architecture, Keep-alive packets are periodically broadcasted over each spanning tree generated by selected nodes. Notification nodes check the arrival of keep-alive messages. If the messages on a specific spanning tree have not arrived during a predefined interval, notification nodes know that a failure has happened on the spanning tree, and then broadcast fail messages over the network. The edge nodes of the Metro Ethernet check whether a spanning tree is down from fail messages and prohibits transmission on the failed spanning tree if it receives a fail message from the particular tree. In this mechanism, failure notification and handling are fully distributed. A central manager is only needed for building multiple spanning trees. The network component failure is resolved without any centralized control. However, this approach needs periodical broadcast of messages in the network which greatly increases the signaling overhead. Furthermore, a single failure would cause the prohibition of transmissions in the whole spanning tree. Even though some traffic do not traverse

the failed network element, they have to change their working spanning trees as long as their edge nodes receive a fail message from the failed spanning tree.

Spanning Tree Elevation Protocol (STEP) [47]

Huynh has presented Spanning Tree Elevation Protocol (STEP) to provide fast recovery in Metro Ethernet. In this mechanism, frames are switched to another spanning tree locally by the Ethernet switch for survivability and load balancing. To avoid possible loops caused in the network, the system allows that the traffic switched to a spanning tree with larger VLAN ID only. Since no transmissions of failure notification and failure detection messages are involved, the recovery time of the crossover spanning tree mechanism is one processing delay of a switch, shorter than other resilient mechanisms for Metro Ethernet networks. Nevertheless, high cost Ethernet switches with additional functions should be designed to modify Ethernet header at line speed (1-10 Gbps) for implementation of the mechanism, otherwise frames would be jammed at the switches adjacent to the failed link. It is a rigid requirement asking for high speed processors and large amount of memories which could not be provided on the current Ethernet switch. In addition, protection with guaranteed bandwidth of the mechanism has not been considered in the mechanism.

2.2.3 Schemes for Ethernet Over WDM

With optical interfaces and wavelength division multiplexing technologies, Ethernet can provide tens of Gbps or even higher transmission rate at low price, which makes Ethernet over WDM networks a suitable candidate for metro and carrier grade networks. Commonly, an Ethernet over WDM networks conform to a two layer model consisting of optical layer and Ethernet layer. Lightpaths are established at the optical layer which form a virtual topology. Ethernet layer uses the virtual topology to build spanning trees. Protection can be provided at optical layer and restoration can be applied at Ethernet layer. Various resilient mechanisms for Ethernet over WDM networks have been discussed. In [48], optical layer protection has been proposed which is similar as traditional 1+1 protection. In [49], Ethernet layer path protection is studied and is formulated as an optimization problem. Since a single physical fiber cut would cause multiple link failures at the Ethernet layer, Ethernet layer path protection in [49] protects against shared risk link group instead of only a single link. GMPLS (Generalized Multi-protocol Label Switching) method which is a technology that provides enhancements to MPLS to support network switching for optical networks is also an option to provide survivability in Metro Ethernet networks [50]. With GMPLS, VLAN IDs can be used as labels to provide safety transmission tunnels. The question of whether

Ethernet over WDM should be protected at optical layer or Ethernet layer is discussed in [51], in which connection level protection provided by Ethernet layer is compared with lightpath level protection provided by optical layer in Carrier Ethernet. However, to the best of our knowledge, little works have considered about the coordination of optical layer and ethernet layer survivability schemes.

2.2.4 Summary

From the above review, it can be seen that the survivability schemes for Metro Ethernet networks proposed so far are either centralized requiring highly reliable central manager or distributed with low bandwidth utilization and no guaranteed protection. Previous distributed survivability schemes for Metro Ethernet networks also need complicated signaling mechanisms. Due to the trend of using distributed survivability schemes in Metro Ethernet networks which are more fault-tolerant than centralized schemes, distributed survivability schemes with simplicity, guaranteed bandwidth, and fast recovery speed should be designed for Metro Ethernet networks.

Chapter 3

Local Restoration with Multiple Spanning Trees

3.1 Introduction

In this Chapter, we present a local restoration mechanism for Metro Ethernet with multiple spanning trees, which aims at fast handling of single link failures in a distributed manner. Each switch has the ability to detect failure and reroute traffic to the backup path on another spanning tree. Pre-configuration is required to establish multiple spanning trees, assign traffic to primary spanning trees and configure backup trees at each switch a priori according to traffic demands. Since switches handle network failures locally without any notification to other part of the network, the recovery time is short and no complex signalling is involved. We

formulate the tree pre-configuration problem (primary spanning tree assignment and backup spanning tree configuration), and prove its NP-Completeness. Then we develop an integer linear programming model. Further, we design heuristics for the tree pre-configuration problem to reduce the computational complexity.

3.2 Framework of Local Restoration

3.2.1 Basic Concept

Different from MPLS local restoration and IP fast reroute, our proposed local restoration mechanism in Metro Ethernet selects appropriate backup spanning trees for rerouting traffic on a working spanning tree and locally restores traffic to the backup spanning trees upon a link failure. The path on a backup tree to reroute the traffic is from the immediate upstream node of the failed link to the destination node, and should exclude the failed link.

Figure 3.1 illustrates the local restoration mechanism in Metro Ethernet: two link disjoint spanning trees are pre-configured in this network so that every single link failure on one spanning tree can be protected by another tree. Suppose that Ethernet frames on connection $1 \rightarrow 5$ are transmitted via path $1 \rightarrow 2 \rightarrow 5$ using ST 1 (solid line). When link $2 - 5$ fails, node 2 detects the failure on ST 1 and switches frames to ST 2 (dash line). Since the path to node 5 on ST 2 does not

include the failed link, the frames can then be safely transmitted. The backup path after restoration for connection $1 \rightarrow 5$ becomes $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$. Generally more than two spanning trees are needed due to topology constraints and traffic engineering issues. These spanning trees may not be link disjoint, but each one could handle several failure scenarios. In such a case, an Ethernet switch would have several candidate spanning trees to protect a particular link failure. Therefore, selection of working spanning trees to transmit traffic and backup trees to guarantee restoration becomes an important issue.

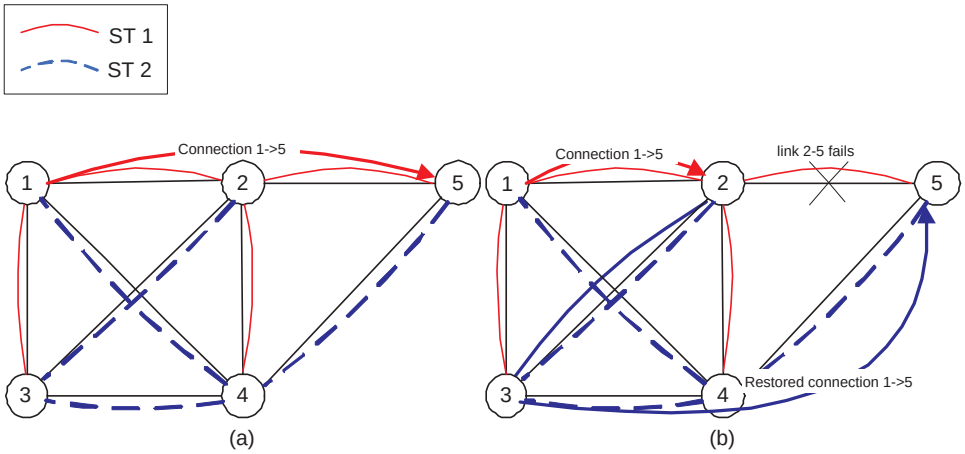


Figure 3.1: Illustration of Local Restoration: (a) transmission before failure (b) local restoration after single link failure

3.2.2 Local Restoration Implementation

Local restoration mechanism in Metro Ethernet could be realized by using present Ethernet protocols. The Ethernet switch should have an additional module for checking the appropriate backup spanning tree and restoring frames to the backup

tree after failure.

Per VLAN Spanning Tree

To implement local restoration from one spanning tree to another, we assume that each spanning tree is assigned a dedicated Virtual LAN ID [13]. Each VLAN does not represent a group of users, but a spanning tree. By changing VLAN ID on Ethernet header, Ethernet frames can be switched among spanning trees. Frames that frequently switch among spanning trees may form unexpected loops. We thus only allow VLAN switching once by setting a bit in frame's CoS field as restoration bit. The Ethernet switch must first check a frames's restoration bit before restoration, and drop those frames that have been earlier restored once. Our mechanism is scalable since the number of spanning trees in the network is not large.

Local Restoration Mechanism

By using local restoration, convergence of spanning trees after failure is no longer necessary. Each switch periodically sends a message to its neighbors to inform its liveness. When a switch does not receive any message from a port for a predefined interval, it marks the port as "failed". If an incoming frame is forwarded to the failed port, the restoration module is activated. A backup VLAN ID which is pre-configured using the algorithms presented later in section 3.4 replaces frames's

original VLAN ID. At the same time, its restoration bit is set to 1. Then, the modified frame is forwarded to the alternative output port according to its new VLAN ID. The switches self-learn the destination of the MAC address on the new VLAN. If the destination is unknown, the frame gets flooded. To avoid flooding upon failure which may lead to heave traffic load in the network, switch can self-learn the destination address on each VLAN by flooding a priori. Figure 3.2 shows a simple example of how the mechanism works: assume a frame on 'VLAN A' arrives and finds that the corresponding output port as "failed" from the forwarding table. Then the restoration bit of the frame is checked and the frame's VLAN ID is changed to 'VLAN B', the pre-configured backup VLAN. Finally, the frame is forwarded to the alternative port on 'VLAN B' according to the forwarding table.

Pre-configuration by Network Manager

The pre-configuration by network manager includes three parts: multiple spanning trees generation, working spanning tree assignment and backup spanning tree configuration.

Multiple Spanning Trees Generation The network manager is responsible for generating multiple spanning trees a priori. The trees should satisfy the condition to handle a link failure: for each link, there is at least one spanning tree that does

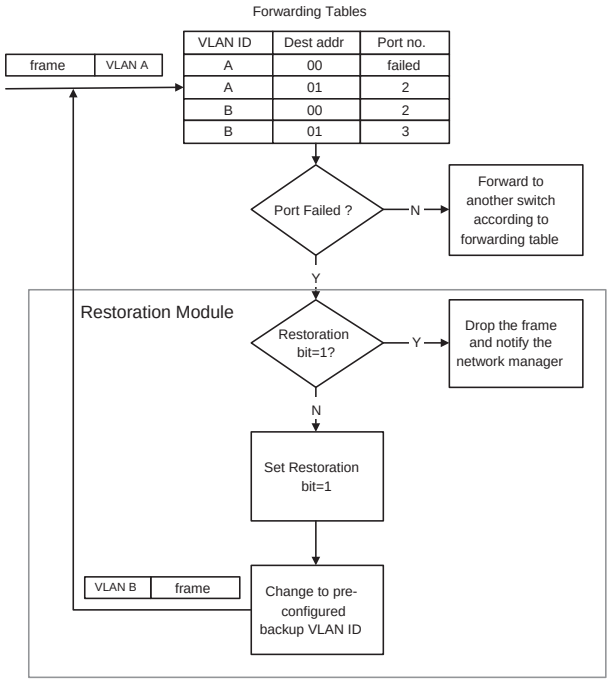


Figure 3.2: Local Restoration Mechanism

not include that particular link [52]. Commonly, more spanning trees should be generated to utilize network resources efficiently.

Working Spanning Tree Assignment The network manager should assign VLAN ID to each source and destination (s-d) pair based on the long term traffic demand matrix. The frames entering the network are attached with VLAN IDs at the ingress switches according to their source and destination addresses, and are forwarded to the proper working spanning trees.

Backup Spanning Tree Configuration When failure happens, frames traversing the failed link should be restored to a pre-configured backup spanning tree

locally. Backup trees at each switch should be carefully configured according to the traffic demand such that there is enough spare capacity on the backup tree for restoration.

3.2.3 Backup Tree Selection Strategy

Backup tree configuration is determined by how Ethernet switch selects the backup tree for each frame. We propose three backup spanning tree selection strategies: link-based, connection-based and destination-based. Under link-based strategy, traffic on the failed link is all restored to one backup spanning tree, no matter which spanning tree or connection these traffic flows belong to. Since link-based strategy has the least flexibility and poor bandwidth sharing, we will not discuss it in the chapter and focus on connection-based and destination-based strategy.

A connection is defined as the traffic between an s-d pair. Connection-based strategy indicates that an Ethernet switch determines the incoming frame's backup VLAN ID according to its source address, destination address, and original VLAN ID. Therefore, traffic between different s-d pairs traversing the same failed link may be restored to different backup trees. Figure 3.3 (b) illustrates an example of connection-based backup tree selection strategy. Three spanning trees are considered to carry working traffic and restored traffic. Connection 1 from node 0 to 2 and connection 2 from node 1 to 2 use ST1 as the working spanning tree before

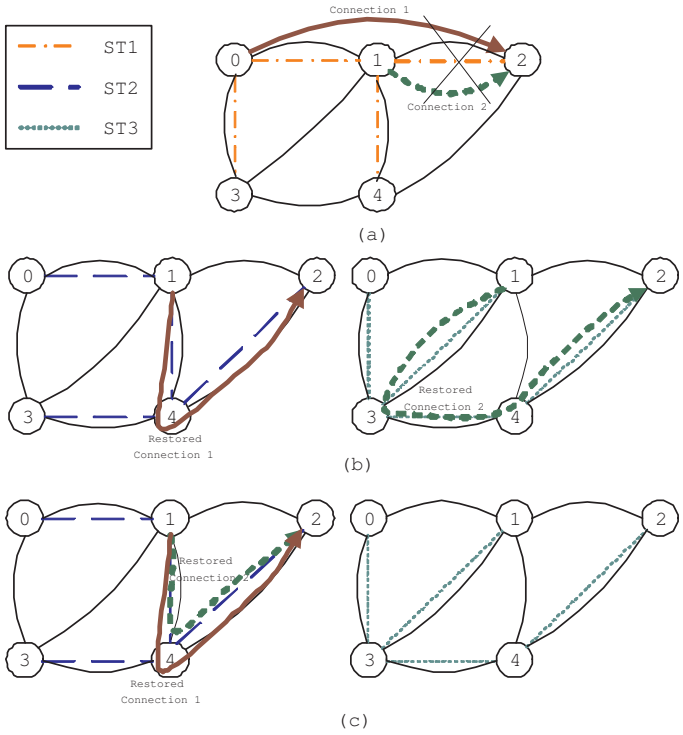


Figure 3.3: Backup tree selection strategy (a) two connections on ST1 before the failure of link $1-2$ (b) two connections are restored to different STs after failure in connection-based strategy (c) two connections are restored to the same ST in destination-based strategy

failure (Fig. 3.3 (a)). When link between node 1 and 2 fails, node 1 restores connection 1 to ST2 and connection 2 to ST3 based on pre-configuration. Frames are restored according to their source and destination MAC addresses, and different connections are assigned backup spanning trees independently.

Connection-based backup tree selection requires a complex computation during pre-configuration, and per s-d pair information should be maintained by each

switch. To reduce the complexity and implementation cost, we propose destination-based backup tree selection strategy, in which frame's backup VLAN ID is determined by its destination address and original VLAN ID, regardless of its source address. Frames with the same VLAN ID and destination address would use the same backup tree in a local Ethernet switch.

Figure 3.3 (c) shows an example of destination-based backup tree selection strategy. Different from the connection-based strategy, since connection 1 and connection 2 have the same destination, they have to use the same backup spanning tree in node 1 upon failure of link 12. Node 1 is unable to restore the two connections with the same destination to the different spanning trees. Destination-based strategy has less flexibility in choosing the backup spanning trees compared with the connection-based one. On the other hand, it has lower computation cost during pre-configuration and frame restoration.

3.2.4 Multiple-Link Failure Issues

The main challenge of our local restoration mechanism to handle multiple-link failures comes from the fact that when a connection is switched to a backup spanning tree, it has no record of the original working spanning tree. Therefore, when the connection encounters another failure on the backup spanning tree, there is a possibility that it would be switched back to the original working spanning tree and

forms a loop in the network.

Our local restoration mechanism is designed to resolve single link failure in Metro Ethernet Networks. However, when multiple-link failure happens in the network and both the primary and backup spanning tree fail simultaneously, some packets would be dropped when they encounter the failure on the backup spanning tree. It is because the mechanism only allows that the traffic switches to the backup tree once for loop avoidance. In [47], traffic can be switched among spanning trees several times, but the traffic can only switch to those backup trees with higher VLAN ID to avoid loops in the network. The method can resolve multiple-link failure problems but adds many constraints on backup tree selection, which would increase the reserved backup capacity in the network.

We now describe two possible approaches to handle these multiple link failures. One approach is to allow multiple VLAN switching, and add more information in the header of the frames, e.g. VLAN ID of the original working spanning tree, when the frames are switched to a backup tree. Therefore, frames will not be switched back to the previous tree. Thus they are able to select a backup tree without forming a loop when they are affected by the second failure in the network. Another approach is to activate the spanning tree re-convergence protocol upon multiple-link failures, e.g. RSTP. When an Ethernet switch finds a packet that has been rerouted once and its output port on the backup tree also fails, the switch should notify the network manager or broadcast the failure message asking for

spanning tree re-convergence by RSTP. This approach cannot handle the failure rapidly, but it is still efficient considering the rare occurrence of multiple-link failure scenarios.

3.3 Problem Formulation

The pre-configuration by network manager determines the network performance. Three parts of pre-configuration: multiple spanning tree generation, working spanning tree assignment, and backup spanning tree configuration could be jointly regarded as an optimization problem. To simplify the problem, we assume that multiple spanning trees are given by a spanning tree generation algorithm. We focus on the working spanning tree assignment and backup spanning tree configuration problems.

We define the problem as below: **Given a network topology, traffic demand matrix, and a set of multiple spanning trees, assign a primary spanning tree to each s-d pair and configure backup trees at each switch to maximize the total admitted end-to-end traffic, with the condition that on each link, the sum of working capacity and spare capacity for restoration does not exceed the link capacity.**

3.3.1 Proof of NP-Completeness

In computational complexity theory, the complexity class NP-complete (abbreviated NP-C or NPC) is a class of decision problems. A decision problem C is NP-complete if:

1. $C \in NP$.

2. Every problem in NP is reducible to C in polynomial time.

Theorem 3.1. *The primary spanning tree assignment and backup spanning tree configuration problem in Metro Ethernet is NP-Complete.*

Proof. At first, we give the decision version of the primary tree assignment and backup tree configuration problem:

Instance: given K spanning trees, traffic demand between each node pair d_{ij}, link capacity C, and a integer value M.

Question: subject to the capacity constraints, is the total admitted traffic greater than M by primary tree assignment and backup tree configuration?

Polynomial-time verification

It is easy to see that the problem $\in NP$, since the non-deterministic algorithm can guess the primary tree assignment for each connection and backup tree at each node, and check in polynomial time whether the solution satisfy the constraints.

Reducibility

We can reduce the known Subset Sum Problem which is NP-Complete [53] to the primary tree assignment and backup tree configuration problem. The decision version of Subset Sum Problem is given below:

Instance: given an integer value M, a set $S = \{s_1, s_2 \cdot \cdot, s_n\}$, and a bound C.

Question: is there a subset $S' \subseteq S$ such that $M \leq \sum_{s_i \in S'} s_i \leq C$?

The reduction is done by constructing a reduction network based on Subset Sum problem as shown in Fig. 3.4. For each element $s_i \in S$, a source node n_i is created. In addition, two intermediate nodes i_1, i_2 and a terminal node t are created. All the solid links in the network are set to have infinite bandwidth, while the capacity of dashed links are set to C. Between each node pair (n_i, t), there exists a connection with the demand of $d_{n_i t} = s_i$. Obviously, the construction can be accomplished in polynomial time. In this particular network, we can establish

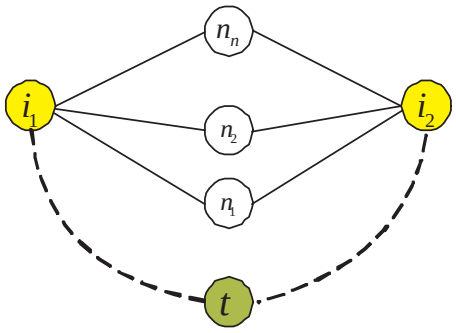


Figure 3.4: Reduction network

four spanning trees as shown in Fig. 3.5 to protect against any single link failure.

It should be noted that if a connection $d_{n_i t}$ selects a primary tree with link $i_1 - t$, its backup tree must include link $i_2 - t$, and vice versa. Let r_i be a binary variable

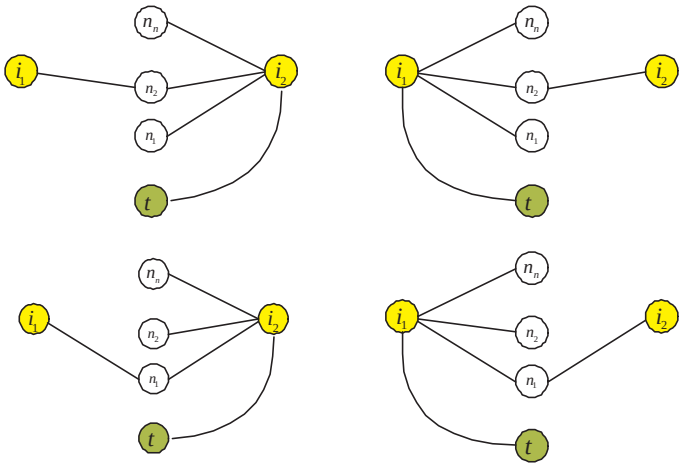


Figure 3.5: Four spanning trees established in the reduction network

which is set to 1 if the primary tree of connection $d_{n_i t}$ includes link $i_1 - t$; g_i be a binary variable which is set to 1 if the primary tree of connection $d_{n_i t}$ includes link $i_2 - t$, and a_i be a binary variable which is set to 1 if the connection $d_{n_i t}$ is admitted. It can be seen that the total working traffic on $i_1 - t$ is $\sum_{s_i \in S} r_i s_i$.

On the other hand, the reserved capacity on $i_1 t$ to protect against any failure of link between the source nodes and intermediate nodes is $\max_{s_i \in S} g_i s_i$, and that to protect link $i_2 t$ is $\sum_{s_i \in S} g_i s_i$, the total working traffic on link $i_2 t$. Hence the total traffic, including working and backup capacity on link $i_1 t$ is $\sum_{s_i \in S} r_i s_i + \max(\max_{s_i \in S} g_i s_i, \sum_{s_i \in S} g_i s_i) = \sum_{s_i \in S} r_i s_i + \sum_{s_i \in S} g_i s_i = \sum_{s_i \in S} a_i s_i$ which is also the total admitted traffic in the network. Obviously, the total traffic on link $i_2 t$ is also $\sum_{s_i \in S} a_i s_i$, and its value must be smaller than the link capacity C.

Suppose there exists a subset $S' \subseteq S$ such that $M \leq \sum_{s_i \in S'} s_i \leq C$ in the instance of the Subset Sum problem. It means that in the reduction network,$M \leq \sum_{s_i \in S} a_i s_i \leq C$. The traffic on each link satisfies the capacity constraints, and the total admitted traffic could be larger than a predefined value. Hence the value of a_i is a feasible solution to the primary tree assignment and backup tree configuration problem.

Suppose the instance of the primary tree assignment and backup tree configuration problem has a solution. It means that there exists a set of a_i such that $M \leq \sum_{s_i \in S} a_i s_i \leq C$. Then we could construct the subset $S' \subseteq S$ by each elements s_i if $a_i = 1$. This concludes our proof that the primary tree assignment and backup tree configuration problem is NP-Complete. □

3.3.2 Integer Linear Programming Model

Since the working spanning tree assignment and backup spanning tree configuration problem is NP-Complete, we use integer linear programming (ILP) approach to solve it. Commonly, links are assumed to be undirected in Ethernet. In our notation, we consider an undirected link l as composed of two directed arcs $\vec{l}$ and $\vec{l'}$, each with the capacity of u_l. Failure of link l indicates the failure of both arcs $\vec{l}$ and $\vec{l'}$. Since single link failure is assumed, we allow spare capacity for restoration

to be shared efficiently. Two types of capacity sharing are used that include inter-sharing and intra-sharing. Inter-sharing allows different connections to share the same backup capacity as long as they do not fail concurrently, and intra-sharing allows different protected links on the working path of a single connection to share the same backup capacity. The variables whose values are obtain from the model are:

α_d^k : binary variable, $\alpha_d^k = 1$ if traffic demand d uses spanning tree k as working spanning tree.

$\beta_{d,\vec{l}}^k$: binary variable, $\beta_{d,l}^k = 1$ if traffic demand d uses backup tree k upon failure on link l (used in connection-based backup tree selection).

$\beta_{ij,k'}^k$: binary variable, $\beta_{ij,k'}^k = 1$ if at node i, frames on spanning tree k' destined to j use backup tree k (used in destination-based backup tree selection).

Our objective is to maximize the total admitted end to end traffic in the network given by,

$$\max \sum_{d \in \mathcal{D}} \sum_{k} d * \alpha_d^k$$

Conditions for connection-based strategy

$$\sum_{k} \alpha_d^k = \{0,1\} \qquad \forall d \in \mathcal{D} \tag{1}$$

$$\sum_{k',\vec{l} \in P_{o_{\vec{l}} t_d}^{k'}} \beta_{d,\vec{l}}^{k'} = 0 \qquad \forall d \in \mathcal{D}, \forall \vec{l} \in E \tag{2}$$

$$\sum_{k',\vec{l} \notin P_{o_{\vec{l}} t_d}^{k'}} \beta_{d,\vec{l}}^{k'} = \sum_{k,\vec{l} \in P_{o_d t_d}^{k}} \alpha_d^k \qquad \forall d \in \mathcal{D}, \forall \vec{l} \in E \tag{3}$$

$$w_{\vec{l}} = \sum_{d \in \mathcal{D}} \sum_{k,\vec{l} \in P_{o_d t_d}^{k}} \alpha_d^k c_d \qquad \forall \vec{l} \in E \tag{4}$$

$$r_{\vec{l}}^{\vec{m}} = \sum_{d \in \mathcal{D}} \sum_{k,\vec{m} \in P_{o_d t_d}^{k}} \sum_{k',\vec{l} \in P_{o_{\vec{m}} t_d}^{k'} \setminus P_{o_d t_d}^{k}} \beta_{d,\vec{m}}^{k'} \alpha_d^k c_d \quad \forall \vec{l}, \vec{m} \in E \tag{5}$$

$$w_{\vec{l}} + r_{\vec{l}}^{\vec{m}} + r_{\vec{l}}^{\vec{m'}} \leq u_l \qquad \forall \vec{l}, m \in E \tag{6}$$

Constraint 1 ensures that a connection is either rejected ($\sum_k b_d^k = 0$) or assigned one and only one primary VLAN ID ($\sum_k b_d^k = 1$). Constraint 2 makes sure that the backup path on backup tree does not include the protected link, and constraint 3 ensures that only one backup spanning tree is selected for each link along the primary path of the connection. Constraint 4 calculates the total working capacity on one link, while constraint 5 calculates the spare capacity on one link to protect another link. Finally, constraint 6 ensures that the spare capacity reserved for restoration on each link plus the working traffic do not exceed the link capacity.

Conditions for destination-based strategy

When destination-based backup selection strategy is used, we need to replace the binary variable $\beta^k_{d,\vec{l}}$ by binary variable $\beta^k_{ij,k'}$. In addition, since destination-based strategy selects a backup tree based on frame's destination and original VLAN ID, we also need to replace constraints 2, 3 in conditions for connection-based strategy by constraints 7, 8, and replace constraint 5 by constraint 9 as below:

$$\sum_{k', P^{k'}_{ij}(1)=P^k_{ij}(1)} \beta^{k'}_{ij,k} = 0 \qquad \forall k, \forall i,j \in V, i \neq j \tag{7}$$

$$\sum_{k', P^{k'}_{ij}(1)\neq P^k_{ij}(1)} \beta^{k'}_{ij,k} = 1 \qquad \forall k, \forall i,j \in V, i \neq j \tag{8}$$

$$r^{\vec{m}}_{\vec{l}} = \sum_{d\in\mathcal{D}} \sum_{k,\vec{m}\in P^k_{o_d t_d}} \sum_{k',\vec{l}\in P^{k'}_{o_{\vec{m}} t_d} \setminus P^k_{o_{\vec{m}} t_d}} \beta^{k'}_{o_{\vec{m}} t_d, k} \alpha^k_d c_d \quad \forall \vec{l}, \vec{m} \in E \tag{9}$$

We note that by changing the objective or constraints of the ILP model, we can provide different quality of service (QoS) for different connections, or balance the load in the network using the local restoration mechanism. Since our objective is to maximize the total admitted traffic in the network, equivalent to minimizing the total link capacity used in the network, we would mainly focus on utilizing the network resources efficiently in the paper. QoS and load balancing issues are beyond the scope of the thesis.

It should be noted that our optimization problem is based on the assumption that multiple spanning trees have been constructed in advance, thus spanning tree generation is not included in the optimization procedure. It is an acceptable practice by the researchers to work with such an assumption. For example, the assumption of pre-calculated path sets has been widely used in MPLS and optical network survivability problems [54][55]. Since current Ethernet switches can only support a limited number of spanning trees, the number of spanning trees in the network should be controlled. It is the fact that such optimization may not produce global optimal results. With K spanning trees established in the network, each connection can select one of K trees as the primary spanning tree, and one of at most $K-1$ trees as the backup spanning tree to protect each link along the primary path. We believe that this selection space is large enough to produce good results. We note that global optimal solution can be obtained by integrating spanning tree generation into the optimization problem.

3.4 Heuristic Algorithms

Since the primary tree assignment and backup tree configuration problem is NP-Complete, we develop heuristic algorithms to solve the problem for both connection-based and destination-based backup tree selection strategies in this section. We use big-O notation to describe the time complexity of each algorithm. The definition

of big-O notation is: Let $f(x)$ and $g(x)$ be two functions defined on some subset of the real numbers. One writes

$$f(x) = O(g(x)) \quad as \quad x \to \infty$$

if and only if, for sufficiently large values of x, $f(x)$ is at most a constant multiplied by $g(x)$ in absolute value. That is, $f(x) = O(g(x))$ if and only if there exists a positive real number M and a real number x_0 such that

$$|f(x)| \leq M|g(x)| \quad \forall x > x_0$$

For each s-d connection in the traffic demand matrix, it is required to select a primary spanning tree to transmit the traffic and a set of backup spanning trees to protect against any single link failure along the primary path of the connection. Sufficient spare capacity needs to be reserved on backup paths on each backup spanning tree. The key idea of the heuristic algorithms is to randomly select a connection at first, and find a primary spanning tree and a set of backup spanning trees with the minimum cost to transmit the traffic. The cost is the sum of backup cost and primary cost of the connection, which is defined in the following section.

3.4.1 Cost Definition

To improve restoration capacity sharing between different connections, the backup cost of a connection to select a backup spanning tree is defined as the total additional amount of backup capacity required on this backup spanning tree. Suppose that we need to select a backup tree for connection d for failure of link l. We define $r^l_{\vec{m}}$ as the existing spare capacity reserved on arc $\vec{m}$ for failure of link l, then the necessary spare capacity reserved for restoration on $\vec{m}$ should be $\max_l r^l_{\vec{m}}$, and the additional reserved spare capacity on $\vec{m}$ for failure of link l used by connection d is $(r^l_{\vec{m}} + c_d - \max_l r^l_{\vec{m}})^+$ which should be equal or larger than zero. The backup cost of connection d's using backup spanning tree k for failure of link l on primary tree k' should be the sum of the backup cost of those links which are on the backup path of the connection d on tree k, but not on the primary path on tree k'.

$$backcost_k(d, \vec{l}) = \sum_{\vec{m} \in P^k_{o_{\vec{l}}, t_d} \setminus P^{k'}_{o_{\vec{d}}, t_d}} \frac{(r^l_{\vec{m}} + c_d - \max_l r^l_{\vec{m}})^+}{u_m}$$

It should be noted that in the definition, $P^k_{o_{\vec{l}}, t_d} \setminus P^{k'}_{o_{\vec{d}}, t_d}$ is the set of links that on backup path but not on primary path, and only these links should reserve spare capacity. Links on both backup and primary path should not be added in backup cost calculation, since the connection could use the reserved working capacity on these links before and after the failure of link l. According to the above definitions,

we could find the backup spanning tree used by connection d for failure of link l with the minimum backup cost, which is also the backup tree for connection d with minimum additional spare capacity.

Once we find the minimum backup cost for failure of any links of the path on primary spanning tree, we calculate the total cost to use the primary tree and a set of backup trees. The total cost is defined as the sum of working capacity of connection's primary path on primary spanning tree and backup cost for each backup tree. The cost of connection d to use working spanning tree k should be:

$$cost_k(d) = \sum_{\vec{l} \in P^k_{o_d,t_d}} (\frac{w_{\vec{l}} + c_d}{u_l} + \min_{k'} backcost_{k'}(d, \vec{l}))$$

Finally a primary spanning tree and a set of backup trees for restoration with minimum total cost are selected for the connection.

3.4.2 Connection-Based Heuristic

The connection-based heuristic aims at improving overall network utilization by considering both working traffic and spare capacity for restoration at the same time. The inputs of our algorithm are the known traffic demand matrix $\mathcal{D}$ and established multiple spanning trees. Paths between node pairs on each spanning tree can be easily computed using breadth first search. Algorithm 3.1 (see Appendix) shows the details of the heuristic. We randomly select an s-d connection d from traffic

matrix at first, then select one spanning tree k as d's candidate primary tree (step 4-9). For every link of d's path on candidate primary tree k, we calculate the backup cost of each candidate backup tree in step 8. In step 8, the total cost of d to use primary spanning tree k is computed, which is the cost of primary tree plus the minimum backup cost. The spanning tree with minimum total cost is selected as d's primary tree (step 11), and backup trees with minimum backup cost are selected (step 13). Connection d is rejected if the primary tree or backup trees do not have sufficient capacity. The outputs of the algorithm are primary VLAN ID $k_{assign}(d)$ assigned to connection d, and the backup VLAN configuration $k_{bkp}(d, \vec{l})$, which is determined by connection d and protected link l.

It can be seen that the dominant computation cost comes from the selection of working spanning trees and backup spanning trees. For each s-d connection, the selection of a working spanning tree requires $O(K|V|)$ time where $|V|$ is the cardinality of node set V. For each link of the primary path in working spanning tree, the selection of a backup spanning tree also takes $O(K|V|)$ time. Hence the time complexity to find a working spanning tree and a set of backup spanning trees for each connection is $O(K^2|V|^2)$. Since the number of connections in a network is at most $|V|(|V|-1)$, the worst case time complexity of the algorithm is $O(K^2|V|^4)$.

3.4.3 Destination-Based Heuristic

The heuristic algorithm for destination-based backup tree selection strategy is similar to that for the connection-based one. We try to find a primary tree and a set of backup trees with the minimum combined cost. However, in the destination-based heuristic, we do not select the backup spanning trees for each connection, instead, connection with the same destination using the same primary tree would be restored to a single backup tree in the local switch. Therefore, destination-based heuristic is separated into two phases. In phase 1, the connection is assigned to a spanning tree with the minimum combined cost. In phase 2, backup VLAN for each destination at each switch is adjusted to reduce the capacity reserved for restoration. At each switch, a new backup tree for one destination is selected if the new backup tree has the minimum additional reserved capacity for restoration of the aggregated traffic to the particular destination on the primary spanning tree. The algorithm is described in Fig. 7. The inputs of the algorithm are the same as those of the connection based heuristic. The output are the primary VLAN ID $k_{assign}(d)$ assigned to connection d, and backup VLAN configuration $k_{bkp}(i, j, k)$ for node i, which is determined by destination j and VLAN ID k.

In the destination-based heuristic, backup spanning tree is not selected for each connection, so the time complexity for a connection to select a working spanning tree and a set of backup spanning trees is $O(K|V|^2)$. In phase 2, the total number

of backup spanning trees that need to be adjusted is $|V|$ and each adjustment takes $O(K|V|)$. The worst case time complexity of the algorithm is $O(K|V|^4)$.

3.5 Performance Evaluation

We carry out the performance study of the proposed local restoration mechanism in Metro Ethernet through simulations. The optimization result is obtained through CPLEX 9.0. Various topologies are used for simulation. Some scalable and flexible optimization techniques to improve the enumeration optimization tools have been developed to solve the ILP problem [56][57]. For example, in [56], a decomposition method based on column generation has been developed to reduce the number of variables in an ILP model for a tree based survivability mechanism, leading to a fast solution.

3.5.1 Spanning Tree Generation

Multiple spanning trees can be generated in many ways. We use a modified version of the spanning tree algorithm proposed in [52] to generate K spanning trees a priori. The original algorithm in [52] generates a necessary number of spanning trees satisfying the condition that for each link, there is at least one spanning tree that does not include the particular link, so the link can be protected by this tree. The spanning tree generation algorithm needs centralized algorithm to assign

weight on each link, and spanning tree generation is distributed. The algorithm starts at a root vertex with the largest degree in the network and generates the first spanning tree according to the degree of each vertex. The second tree is generated to include as many links not belonging to the first tree as possible. The algorithm iterates until the above condition is satisfied.

However, the number of spanning trees generated by the algorithm is limited to less than 10 even in a large network. To provide more working and backup spanning tree selections for each connection so that network resources could be utilized more efficiently, we generate more spanning trees based on the concept that for each link, there should be as many as possible spanning trees not including the particular link. We set a weight $w(i)$ for each link i representing the number of spanning tree that do not include link i. The algorithm maximizes the minimum weight $w(i)$ in the network during construction of each spanning tree. In this way, the selection space of backup spanning trees for each link is increased.

3.5.2 Optimal vs. Heuristic

We compare the numerical results obtained by CPLEX and simulation results by our heuristic algorithm for a small 3×3 grid network and a medium size 4×4 grid network. The capacity for each link is assumed to be 100Mbps. We choose identical traffic pattern with 10Mbps traffic between each node pair in 3×3 grid

network. Different number of spanning trees are generated by the spanning tree algorithm. Due to the scalability problem, in 4×4 grid network, we randomly select 8 nodes working as sources and destinations while other nodes is cross-connects. One connection is established between each s-d pair and the total number of connections does not exceed 100. The results in Table 3.1 show that our heuristic algorithms for both connection-based and destination-based backup spanning tree selection strategies achieve very closely to the optimization values. Gap which is the percentage of the difference between optimal and heuristic solution is within 10% in any of the three network scenarios. The number of variables needed when 5x5 grid network is run for local restoration optimization is larger than 30000. We observed that it took more than 15 hours to obtain the optimal solution on 3×3 grid network with 4 spanning trees, and more than 4 days to obtain the optimal solution on 3×3 grid network with 6 spanning trees and 4×4 grid network with 4 spanning trees. Assume there are four spanning trees in the network, and the number of connections is 100. The number of primary VLAN assignment variable is 400, and the number of backup VLAN selection variable is 32000. The number of constraints is as large as 23200. With brutal force search, there are totally $2^{100*320}$ operations which is not scalable. Therefore, for larger networks, we study the performance of the heuristic algorithms only. In our optimization, we have assumed that all the nodes in the network are both sources/destinations and cross-connects of network traffic, which results in large traffic exchange on each

node. In a real network scenario with hundreds of nodes, only a small portion of nodes in the network carry heavy traffic. Since the network performance (admitted traffic and network redundancy) are mainly affected by the large exchanges, even though the number of nodes in our experiment is smaller than that in real network scenario, it still can provide evidence that proposed algorithms are efficient in real networks.

Table 3.1: Total Admitted Traffic for Different Networks (Mbps)

	3×3 Grid (4 trees)		3×3 Grid (6 trees)		4×4 Grid (4 trees)	
	Connection	Destination	Connection	Destination	Connection	Destination
Optimal	640	570	660	610	700	650
Heuristic	620	560	630	600	650	620
Gap	3.1%	1.8%	4.8%	1.4%	7.7%	4.8%

3.5.3 Throughput and Redundancy

We also evaluate the performance of the two backup tree selection strategies in terms of total admitted traffic and resource redundancy in larger size networks, including a regular network (grid network) and irregular networks (random networks). The resource redundancy is defined as the ratio of the spare capacity used for restoration over working capacity, which is similar to network redundancy as defined in [54].

Grid Networks

In this part, the simulation runs on grids with 16, 25, 36, 49 and 64 nodes. The capacity of duplex links are set to be 100Mbps. Two traffic patterns, identical and non-identical traffic pattern, are used. In identical traffic scenario, traffic demand for each node pair is set to be 5Mbps; and in non-identical scenario, s-d traffic is uniformly distributed in [0,10]Mbps. Different number of spanning trees is generated according to the network size. The number of spanning trees for 16, 25, 36, 49 and 64 nodes networks are 10, 25, 25, 35, 35, respectively.

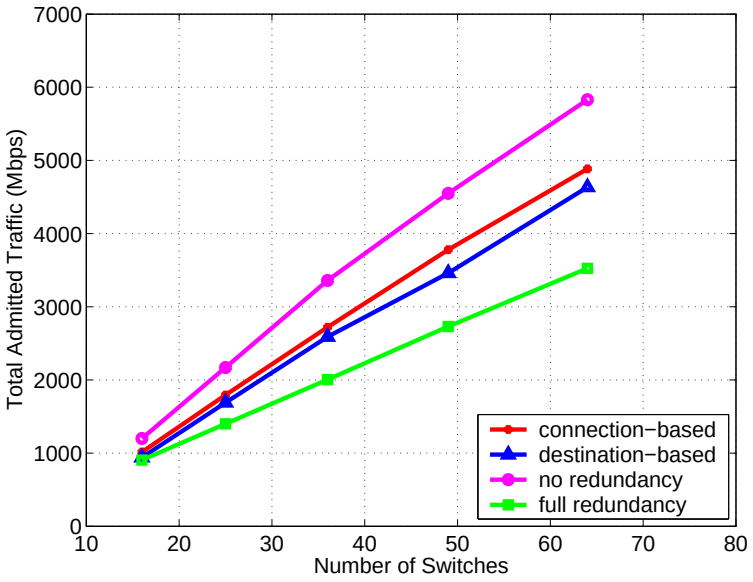
To examine the efficiency of our backup tree selection strategy. We also compare our algorithms with other two schemes. One scheme is the traffic assignment to spanning trees without redundancy of local restoration. Another scheme is the restoration with full redundancy, in which half of the capacity on each link is used for working traffic while the other half is reserved for restoration. The scheme can fully protect any single link failure no matter what kind of backup tree selection strategy is used by local switch.

Results in both identical traffic (Fig. 3.6) and non-identical traffic scenario (Fig. 3.7) show that our local restoration mechanisms could efficiently utilize the network bandwidth due to bandwidth sharing. Compared with full redundancy scheme, the total admitted traffic using our algorithm is much higher, especially when network size increases. It can also be seen that the performances of

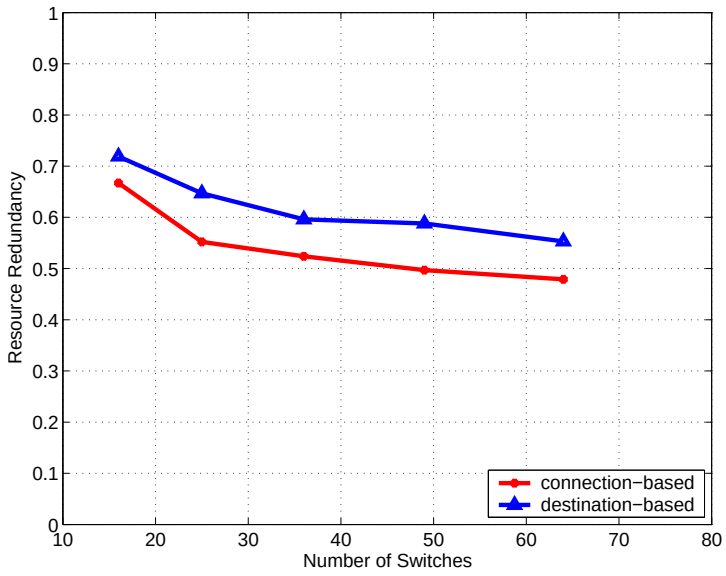
connection-based and destination-based backup tree selection strategy are close. Even though destination-based strategy uses less frame's information to restore traffic, it also could utilize network resources efficiently. In terms of resource redundancy, the result show that the redundancy decreases when network size increases. It is because that more paths could be obtained in large network, which helps more connections to share the same spare capacity for restoration.

Random Networks

In this part, random networks with 100 nodes is used to evaluate the heuristic algorithms. We fix the network average degree to be from 3 to 6, and then generate 10 different networks randomly with each degree using Waxman's algorithm [58]. The capacity of each duplex link is also set 100Mbps, and the connection between each node pair is set 1Mbps. We adjust the number of spanning trees used in the network to see whether it would greatly influence the performance of the heuristic algorithms. Apparently, the more spanning trees are used, the more number of different paths are generated between each node pairs, which increases the selection spaces of working spanning tree and backup spanning tree during pre-configuration. Therefore more amount of end to end traffic is expected to be admitted in the network. However, large number of spanning trees leads to high network cost. If the performance improvement by increasing number of trees is limited, it would be unnecessary to use such a large number of trees. It is possible that there

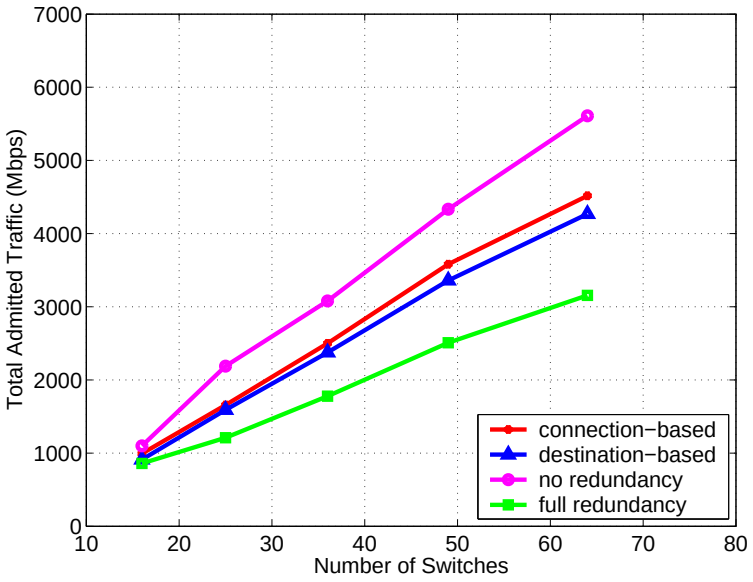


(a)

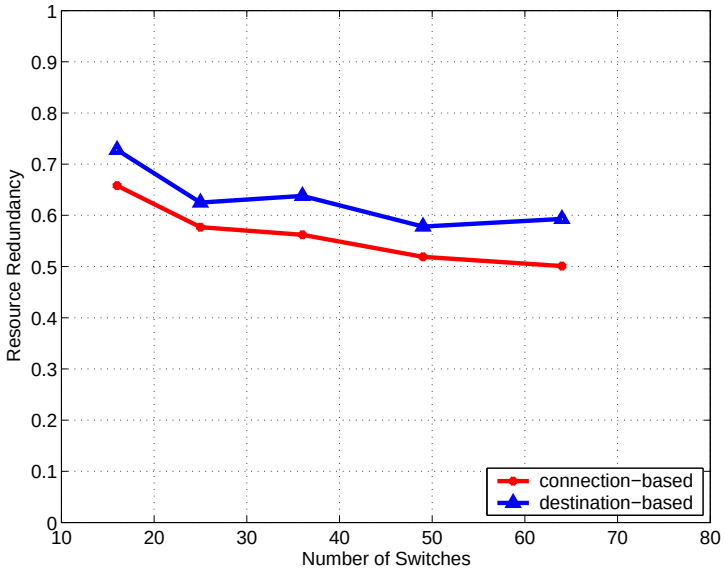


(b)

Figure 3.6: Identical traffic scenario on grid topologies (a) Total admitted traffic, (b) Resource redundancy



(a)

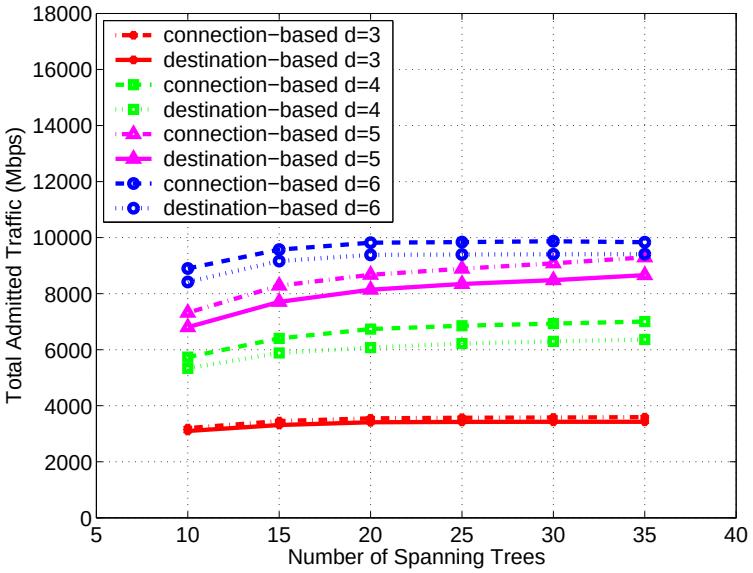


(b)

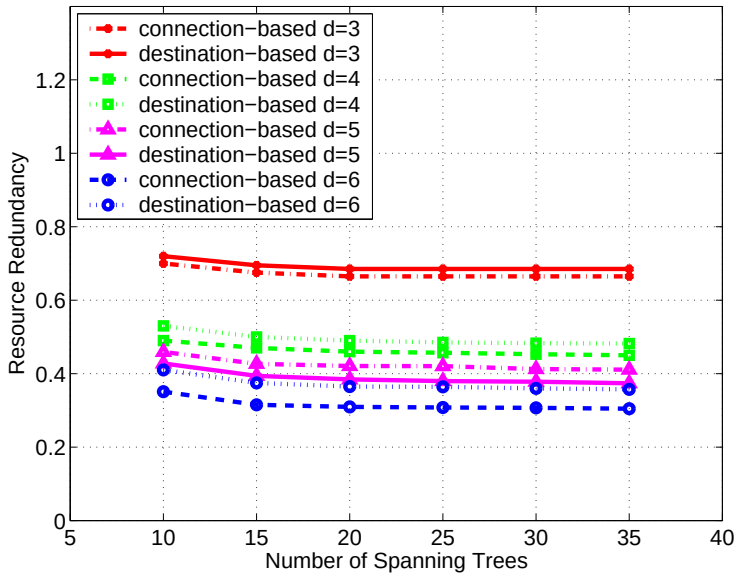
Figure 3.7: Non-identical traffic scenario on grid topologies (a) Total admitted traffic, (b) Resource redundancy

exists an optimal group of spanning trees maximizing the total admitted traffic with our local restoration mechanism. The optimal spanning tree group not only depends on the network topology, but on the traffic demands as well. To obtain the optimal spanning tree group, spanning tree generation can be integrated into the optimization problem [56] which is not considered in our work.

We run the simulations on groups of ten different topologies with each degree (d=3, d=4, d=5 and d=6), and take the average results on each ten networks. Figure 3.8 shows the total admitted traffic and resource redundancy of the two heuristics in random networks. The X-axis in two diagrams is the number of spanning trees in the network. From the results, we observe that when the number of spanning trees increases, the performance of the two heuristics are not improved drastically, which indicates that even with small number of spanning tree, the heuristic algorithms can achieve good performance. It also can be concluded that the necessary number of spanning trees for local restoration is limited (< 35 in 100 nodes random networks). Further, it can be seen that the performance of the destination-based heuristic is close to that of the connection-based heuristic in random networks, which is same as our observation in grid topologies. The difference of the two heuristics in networks with degree 4 (nearly 10%) is larger than that of the two algorithms in networks with degree 3. The reason is that in denser network, the connection-based heuristic has more flexibility in choosing backup spanning trees to share spare capacity.



(a)



(b)

Figure 3.8: Random topologies (a) Total admitted traffic, (b) Resource redundancy

Another observation from the results is that when the degree of the network increases to 5 and 6, the performance gap of the total admitted traffic between connection-based and destination-based heuristics remains almost constant. However, the performance gap between the two heuristics in terms of the redundancy keeps on increasing. It indicates that, in dense networks where most of the connections are admitted, even though the network has better bandwidth sharing under connection-based strategy, the difference of admitted traffic between two heuristic algorithms does not change significantly. It can be attributed to the fact that most of the connections rejected in the dense networks results from the topology constraints but not due to insufficient backup capacity.

3.5.4 Implementation Cost

As discussed in Section 3.2, a restoration module is used to change the VLAN ID of a frame to a pre-configured backup VLAN ID in our local restoration mechanism. Since the failure is handled without any failure notification to other Ethernet switches, the signaling overhead of the mechanism is zero.

The dominant cost of the local restoration mechanism comes from the restoration module. To obtain the pre-configured backup VLAN ID of a frame upon failure, the restoration module can be implemented by setting a backup VLAN list. The backup VLAN list is a hash table which is computed in advance by the

central manager of the network based on our optimization model or heuristics, and then sent to each Ethernet switch. The output of the list is a backup VLAN ID, while the entry depends on the backup spanning tree selection strategy used. When the connection-based strategy is used, the entry of the backup VLAN list is frame's primary VLAN ID, source and destination MAC address. When the destination-based strategy is used, the entry is only frame's primary VLAN ID and destination MAC address.

A larger backup VLAN list requires more storage space and faster processer in the Ethernet switch, increasing the implementation cost. If we assume the number of nodes in the network is N and the number of spanning trees is K, the number of the entries for connection-based backup VLAN list in a Ethernet switch should be less than KN^2, while the number of those for destination-based backup VLAN list is at most KN.

Figure 3.9 shows the average number of entries per Ethernet switch in grid networks with different sizes. Here, we assume the identical traffic scenario. It can be observed that the size of the backup VLAN list increases when the network size increases, and the backup VLAN list for the destination-based strategy is about 80% as large as that for the connection-based strategy for grid networks. Hence in large-scale networks, local restoration mechanism with the destination-based backup selection strategy is more promising, since it has good performance close to that of the connection-based strategy, but lower implementation cost.

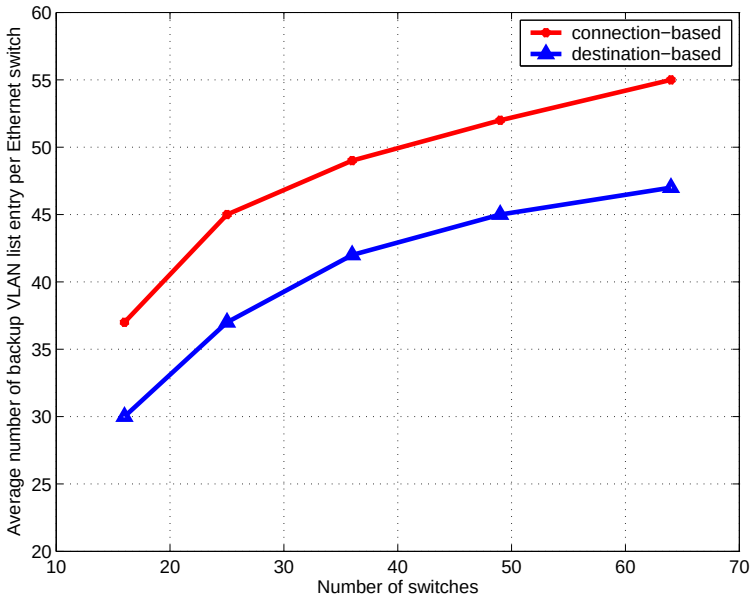


Figure 3.9: Average number of backup VLAN list's entries per Ethernet switch in grid networks

3.6 Conclusions

In this chapter, we proposed a fast and distributed local restoration mechanism in Metro Ethernet networks. Upon failure of a link, a switch can restore traffic to pre-configured backup spanning trees locally. We presented the implementation details of the mechanism. We formulated the tree pre-configuration problem that includes the working spanning tree assignment and backup spanning tree configuration. We proved that the pre-configuration problem is NP-Complete and developed ILP models based on different backup tree selection strategies (connection-based and destination-based). We also developed heuristic algorithms for each backup tree selection strategy. Simulation results show that the heuristic algorithms are effective

and performs close to the optimum one. They also show that the network capacity can be efficiently utilized by both the connection-based and destination-based backup tree selection strategies.

3.7 Appendix

Algorithm 3.1 Connection-Based Heuristic

Compute the path between each node pair on each spanning tree P_{ij}^k.
repeat
  Randomly select a traffic demand $d \in D$.
  for $k = 1$ to K and each arc $\vec{l} \in P_{o_d t_d}^k$ **do**
    for $k' = 1$ to K and $\vec{l} \notin P_{o_d t_d}^{k'}$ **do**
      $backcost_{k'}(d, \vec{l}) = \sum_{\vec{m} \in P_{o_{\vec{l}}, t_d}^{k'} \setminus P_{o_{\vec{d}}, t_d}^{k}} \frac{(r_{\vec{m}}^l + c_d - \max_l r_{\vec{m}}^l)^+}{u_m}$
    end for
    $cost_k(d) = \sum_{\vec{l} \in P_{o_d, t_d}^k} (\frac{w_{\vec{l}} + c_d}{u_l} + \min_{k'} backcost_{k'}(d, \vec{l}))$
  end for
  if primary and backup trees have sufficient capacity for d **then**
    Assign Demand d to spanning tree $k_{assign}(d) = arg\min_k cost_k(d)$.
    for each arc $\vec{l} \in P_{o_d t_d}^{k_{assign}(d)}$ **do**
      $k_{bkp}(d, \vec{l}) = arg\min_{k'} backcost_{k'}(d, \vec{l})$
      Update $w_{\vec{l}}$ and $r_{\vec{m}}^l$ on each arc $\vec{m}$.
    end for
  else
    Reject the connection d.
  end if
until No traffic demand can be assigned

Algorithm 3.2 Destination-Based Heuristic

Compute the path between each node pair on each spanning tree P_{ij}^k.
Set initial value of $k_{bkp}(i,j,k)$ to be a backup tree with the minimum hop
repeat
  Randomly select a traffic demand $d \in D$.
  PHASE 1: Traffic Assignment
  for $k = 1$ to K and each arc $\vec{l} \in P_{o_d t_d}^k$ **do**
    $backcost_{k_{bkp}(o_{\vec{l}},t_d,k)}(d,\vec{l}) = \sum_{\vec{m} \in P_{o_{\vec{l}},t_d}^{k_{bkp}(o_{\vec{l}},t_d,k)} \setminus P_{o_{\vec{d}},t_d}^k} \frac{(r_{\vec{m}}^l + c_d - \max_l r_{\vec{m}}^l)^+}{u_m}$
    $cost_k(d) = \sum_{\vec{l} \in P_{o_d,t_d}^k} (\frac{w_{\vec{l}} + c_d}{u_l} + backcost_{k_{bkp}(o_{\vec{l}},t_d,k)}(d,\vec{l}))$
  end for
  if primary and backup trees have sufficient capacity for d **then**
    Assign Demand d to spanning tree $k_{assign}(d) = arg\min_k cost_k(d)$.
  else
    Reject the connection d. *PHASE 2*: Backup Tree Adjustment
    for each arc $\vec{l} \in P_{o_d t_d}^{k_{assign}(d)}$ **do**
      for $k' = 1$ to K and $P_{o_l t_d}^{k'}(1) \neq P_{o_l t_d}^{k_{assign}(d)}(1)$ **do**
        $k_{bkp}(o_{\vec{l}}, t_d, k_{assign}) = arg\min_{k'} \sum_{\vec{m} \in P_{o_{\vec{l}},t_d}^{k'} \setminus P_{o_{\vec{d}},t_d}^k} \frac{(r_{\vec{m}}^l + c_d - \max_l r_{\vec{m}}^l)^+}{u_m}$
      end for
      Update $w_{\vec{l}}$ and $r_{\vec{m}}^l$ on each arc $\vec{m}$.
    end for
  end if
until No traffic demand can be assigned

Chapter 4

Fast Spanning Tree Reconnection

4.1 Introduction

Local Restoration mechanism proposed in the last chapter can provide fast and guaranteed protection for Metro Ethernet networks, however, spanning tree switching by changing frames' VLAN ID requires high speed processor (as fast as line speed) in Ethernet switch, greatly increasing the implementation cost.

In this chapter, we propose a fast spanning tree reconnection (FSTR) mechanism to protect against single link failures in Metro Ethernet networks. When a link on a spanning tree fails, the spanning tree is divided into two separated subtrees, and a fast spanning tree reconnection protocol is activated to reconnect the two subtrees by using a pre-configured link called as *reconnect-link*. Therefore, a major part of the spanning tree structure is kept unchanged after the failure.

In the protocol, a *failure notification table* and an *alternate output port table* are configured at each switch based on the pre-computed *reconnect-link*s. By checking the two tables, switching tables of the Ethernet switches affected by the spanning tree reconnection are reconfigured in a way that traffic traversing the failed link is quickly rerouted and backward learning is avoided. Pre-configuration of *reconnect-link*s to reconnect the spanning tree is considered as an optimization problem to efficiently utilized the network resources. We prove that this problem is NP-complete, and an efficient polynomial time algorithm that can obtain close approximation to optimal solution is given. The mechanism has the features of fast recovery and backup capacity guarantees. Moreover, the implementation cost is low with small modifications on the legacy Ethernet switches.

4.2 Fast Spanning Tree Reconnection Mechanism

4.2.1 Concept

Upon a single link failure on a spanning tree, the tree is separated into two subtrees. If there exists another link between any two nodes each in a different subtree, the link can be used to reconnect the spanning tree.

Theorem 4.1. *In a two-connected network, a* reconnect-link *to reconnect the failed*

spanning tree always exists.

Proof. A two-connected network is still connected after any single link failure, which indicates that there must be at least one path between any two nodes on different sub-trees. If we cannot find a single link to reconnect the two subtrees, the path cannot be found, which is contradictory to the definition of two-connected network. □

In FSTR, when a link fails, The adjacent nodes of the failed link send notification messages based on pre-configuration to the end nodes of a link asking them to reconnect the spanning tree by activating the link called as *reconnect-link*. We call the path on the reconnected spanning tree between the two end nodes of the failed link as *reconnect-path*. The traffic on the failed link is rerouted onto the *reconnect-path* after tree reconnection, bypassing the failed link. It should be noted that a *reconnect-path* is determined by a failed link and its *reconnect-link*. We illustrate the working of the FSTR mechanism in Fig. 4.1. In this 9-node network, a spanning tree is represented by the solid line. Links $F-H$ and $A-I$ are the possible candidate links that can be used to reconnect the spanning tree upon failure of link $G-C$. Let link $F-H$ be the *reconnect-link* for failure of link $G-C$, represented by the dashed line. Upon failure of link $G-C$, the spanning tree is separated into two subtrees, subtree 1 with nodes set $\{A, B, C, D, E, F\}$ and subtree 2 with nodes set $\{G, H, I\}$. Based on pre-configuration, node C sends notification messages to

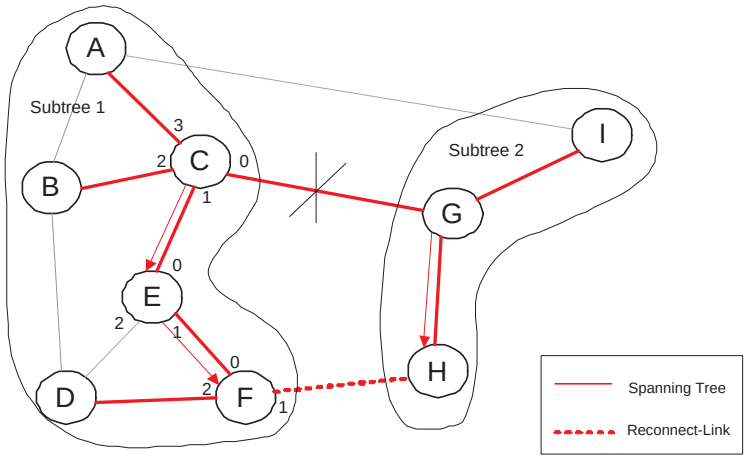


Figure 4.1: Illustration of Fast Spanning Tree Reconnection: when link $G-C$ fails, C notifies F ($C \rightarrow E \rightarrow F$), G notifies H ($G \rightarrow H$), then link $F-H$ reconnects the tree

node F via $C \rightarrow E \rightarrow F$ and node G sends message to node H via $G \rightarrow H$. Every node on the *reconnect-path* $C-E-F-H-G$ re-configures its switching table after receiving the message. When F and H are notified, link $F-H$ (dashed line) is activated to reconnect the spanning tree. We can observe from the example that only those nodes along the *reconnect-path* need failure notification and switching table reconfiguration, while the failure is transparent to other nodes in the network.

4.2.2 FSTR Protocol

To implement the fast spanning tree reconnection, a protocol for failure notification and switching table reconfiguration in switches along the *reconnect-path* is required. There are two implementation approaches of FSTR on Ethernet switches, learning approach and non-learning approach. Both of them require each Ethernet switch

to maintain a *failure notification table*. The second approach needs another table called *alternate output port table* to be maintained at each switch.

The *failure notification table* helps the neighboring node of a failure to find the reconnected nodes and send the a failure notification message to reconnect the spanning tree. The input of the table are the failed port number, and the table includes two elements:

1. Reconnected-node MAC address: address of the end node of the *reconnect-link*.

2. Failed-Link ID: identity of the failed link.

According to the failed-link ID in failure notification message sent from end nodes of the failed link, *alternate output port table* tells switches on the *reconnect-path* which output ports to destinations are changed on the reconnected tree, and replace them by alternative ports. The two elements in *alternate output port table* are:

1. Disconnected-destination: a set of destinations that is disconnected with the switch due to the failure indicated by Failed Link ID.

2. Alternative-port: the output port number to replace the original port number in the switching table.

When a switch detects the failure from one of its output ports, it immediately

obtains the MAC address of the node adjacent to *reconnect-link* (reconnected node) from the *failure notification table*. It then sends a failure notification message containing failed-link ID to the reconnected node. When the learning approach is used, an Ethernet switch receives a failure notification message, changes its port state according to the failure notification message, flushes its switching table and returns back to the backward learning stage. In this case, the switch does not need to know which destination MAC address is not reachable after the failure. The recovery speed is slower since the switch has to return back to learning stage. When non-learning approach is used, as soon as a node on the *reconnect-path* receives the failure notification message, it uses the failed-link ID to find the alternative output port of disconnected destinations in the alternate output port table. Disconnected-destinations are a set of destinations that the node on the *reconnect-path* cannot communicate due to the particular link failure. Then, the output port to these destinations in switching table is changed to the alternative-port. When the notification message arrives at the reconnected node, in addition to the switching table reconfiguration, the node also changes the state of the alternative port from "discarding" to "forwarding" to reconnect the spanning tree. The non-learning approach can provide faster recovery speed, however, each Ethernet switch must maintain the information of which nodes are not reachable upon each link failure, increasing the cost of the switches.

Figure 4.2 shows the *failure notification table* of Node C and *alternate output*

Failed Port	0	1	2	3
Reconnected-node MAC	F	H	A	B
Failed-link ID	CG	CE	CA	CB

(a)

Failed Link ID	CG	GH	DF	...
Disconnected-destinations	G,H,I	H	D	...
Alternative-port	1	1	2	...

(b)

Failed Link ID	CG	GH	DF	...
Disconnected destinations	G,H,I	H	D	...
Alternative-port	1	1	0	...

(c)

Figure 4.2: (a) Failed Notification Table on Node C (b) Alternate Output Port Table on Node E (c) Alternate Output Port Table on Node F

port table of Node E and F on the network in Fig. 4.1. When C detects the failure of port 0, after checking the *failure notification table* (4.2 (a)), it sends a message containing failed link id 'CG' to F along the path $C \rightarrow E \rightarrow F$. It also reconfigures its switching table.

When node E receives the failure notification message, it uses the failed link ID to check its *alternate output port table* and reconfigure the switching table. According to the table, frames to destinations G, H, and I need to use alternate ports, so the output ports of these destinations are changed to port 1 at node E (4.2 (b)). The failure notification message then is forwarded to node F. After receiving the message from node E, node F does the same *alternate output port table* checking and switching table reconfiguration as E does (4.2 (c)). Further, F

should change the state of output port 1 from "discarding" to "forwarding".

No matter whether learning or non-learning approach is used to implement the FSTR mechanism, a central scheduler is needed to pre-configure each Ethernet switch according to the network topology and traffic demands. We note that the assumption of centralized scheduler has been widely used in Metro Ethernet networks for the purpose of spanning tree generation, failure handling and VLAN assignment

The FSTR mechanism is designed for single link failure recovery. When more than one link fail concurrently and the two *reconnect-path*s are link disjoint, the two reconnect-links can reconnect the spanning tree independently without any loops. However, if a switch receives two failure notification messages from different nodes indicating that a loop may be formed in the network with FSTR., it must notify the network manager or broadcast to the network that the failure scenario cannot be handled, then RSTP shall be activated for spanning tree re-convergence. We will discuss and study the details of handling multiple link failure with FSTR mechanism in the next chapter

The previous discussion demonstrates several attractive features of FSTR mechanism:

1. With *failure notification table* and *alternate output port table* at each Ethernet switch, the FSTR mechanism can avoid flushing the entire spanning

tree and the backward learning through appropriate re-configuration of the switching table.

2. The recovery time of FSTR is short. As long as the two end nodes of the failed link reconfigure their switching tables, affected Ethernet frames can be immediately rerouted to the *reconnect-link* along the *reconnect-path*. The failure notification message is sent to the switches along the *reconnect-path* before rerouting the frames. To ensure that these switches have been reconfigured before rerouted frames arrive, such that frames can be safely transmitted on the reconnected spanning tree, a delay for reconfiguration should be added between the failure notification message and rerouted traffic. Hence the recovery time is likely to be slightly longer than the maximum reconfiguration delay of an Ethernet switch, which is acceptable in MAN.

3. The protocol can be implemented in conventional Ethernet switch, and the signaling overhead is low. Only two messages from opposite direction of the *reconnect-path* is sent after the failure. The cost of FSTR mechanism comes from pre-computed *failure notification table* and *alternate output port table*. The forwarding logic of the Ethernet switch does not need significant modification. FSTR mechanism only changes the port states of the Ethernet switches to build a reconnected spanning tree. We believe that the

implementation cost of the FSTR mechanism is lower than that of the local restoration mechanism, since the local restoration mechanism needs line speed VLAN switching in each Ethernet switch which requires a high speed processor.

It should be noted that *failure notification table* and *alternate output port table* on each Ethernet switch can be easily computed in advance according to the *reconnect-link*s to reconnect the spanning tree upon single link failure. The *reconnect-link*s should be properly pre-computed off-line to make the FSTR mechanism achieve a good performance. In the next section, we discuss how much backup capacity should be reserved on the *reconnect-path* when the *reconnect-link* has been selected, and formulate the FSTR pre-configuration problem as an optimization problem.

4.3 Backup Capacity Provisioning-Problem Formulation

4.3.1 Backup Capacity Calculation

On the reconnected spanning tree upon a link failure, traffic on some links would increase due to traffic rerouting. Backup capacity should be reserved to carry the increased amount of traffic on these links. We need to determine the amount of backup capacity reserved on each link for bandwidth guarantees. Backup capacity

planning have been widely studied in MPLS networks and optical networks. The problem on tree topology has also been studied. However, the backup capacity reservation with FSTR mechanism is different from previous studies. Upon failure of a protected link, the working traffic on the link is rerouted onto the *reconnect-path*. Apparently, only links on the *reconnect-path* of a protected link need backup capacity reservation to guarantee protection of the particular link.

Let w_f be the working traffic on protected undirected link f, and undirected link l be a link on the *reconnect-path*. When the working traffic on f is rerouted onto the *reconnect-path*, the backup capacity reserved on link l for f should be at most w_f. Let $w_{l,f}$ be the traffic traversing both link f and l on the original tree, traffic $w_{l,f}$ would not traverse link l after the spanning tree reconnection. This is because that the path on reconnected spanning tree to carry traffic $w_{l,f}$ does not include link l. Hence, rerouted traffic on link l is $w_f - w_{l,f}$ instead of w_f. Meanwhile, the working traffic on link l is reduced to $w_l - w_{l,f}$ after the spanning tree reconnection. Since the released part of the working capacity can be used to carry backup traffic, the additional backup capacity reserved on link l should be $\max\{(w_l - w_{l,f}) + (w_f - w_{l,f}) - w_l, 0\} = \max\{w_f - 2w_{l,f}, 0\}$. It also can be represented as $(w_f - 2w_{l,f})^+$.

The example in Fig. 4.3 shows the traffic on spanning tree before failure and traffic on the reconnected tree after failure. We assume that there are three 1 unit bidirectional connections in the network ($0 \leftrightarrow 7$,$5 \leftrightarrow 2$,$6 \leftrightarrow 2$). Link $6-7$ is

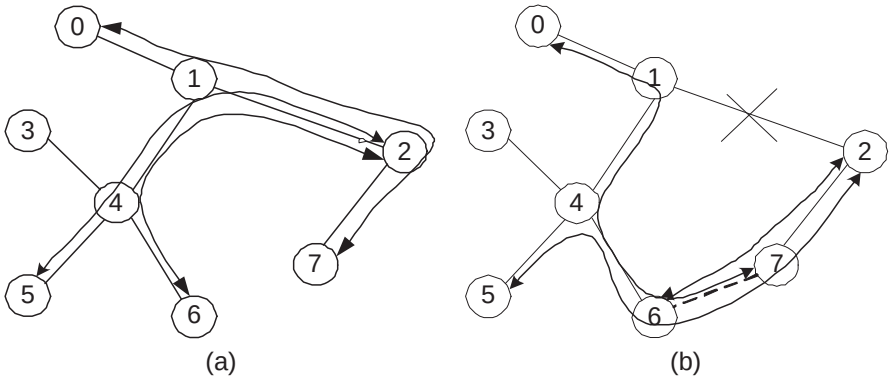


Figure 4.3: ((a) Traffic on a spanning tree before failure (b) Traffic on the reconnected spanning tree after failure of link $1-2$

configured as the *reconnect-link* to reconnect the spanning tree upon failure of link $1-2$, so the *reconnect-path* of the protected link $1-2$ is $1 \leftrightarrow 4 \leftrightarrow 6 \leftrightarrow 7 \leftrightarrow 2$. Let us consider the backup capacity reserved on link $4-6$. On the spanning tree before failure, working traffic on $4-6$ is $w_{46}=1$ and on protected link $1-2$ is $w_{12}=3$. It can also be seen that traffic traversing both link $1-2$ and $4-6$ is $w_{12,46}=1$. On the reconnected tree after the failure of link $1-2$, since connection $6 \leftrightarrow 2$ no longer traverses link $4-6$, working traffic on $4-6$ would be reduced by $w_{12,46}=1$. We note that the $w_{12,46}$ units of working capacity reserved on spanning tree before failure is released and can be used to carry rerouted traffic on reconnected tree. Since the rerouted traffic on link $4-6$ after the reconnection is $w_{12}-w_{12,46}=2$, the additional backup capacity reserved on link $4-6$ to account for the failure of link $1-2$ is $w_{12}-2w_{12,46}=1$.

4.3.2 Problem Formulation

FSTR pre-configuration is based on the knowledge of established spanning trees, and traffic of long term end-to-end connections.

FSTR pre-configuration problem: Given one or more spanning trees established in a network and a set of end-to-end connections, assign each connection to an appropriate working spanning tree, and select the best *reconnect-link*s to reconnect each spanning tree in case of single link failures, such that the backup capacity reserved in the network is minimized. We note that when only a single spanning tree has been built, we only need to solve the *reconnect-link* pre-configuration problem since connections can traverse only on the single working spanning tree. However, with multiple spanning trees, connection assignment and *reconnect-link* pre-configuration become a joint optimization problem.

4.3.3 Integer Linear Programming Model

FSTR pre-configuration problem can be represented by a integer linear programming model. We assume that links are undirected, and end-to-end connections are bidirectional. Before the optimization, we should pre-compute *reconnect-link* set R_f^k for each protected link f, which is the set of links that can reconnect spanning tree k when link f fails. The *reconnect-link* to protect f is the one selected from

R_f^k. The *reconnect-path* RP_{fl}^k can then be computed for a protected link f on tree k and a potential *reconnect-link* l in set R_f^k. The variables that the ILP model tries to determine are:

α_d^k : binary variable, $\alpha_d^k = 1$ if connection d uses spanning tree k as working spanning tree

γ_{fl}^k : binary variable, $\gamma_{fl}^k = 1$ if on spanning tree k, link l is the *reconnect-link* upon failure of link f.

The objective is to minimize the total backup capacity reserved on links:

$$\min \sum_{m \in E} r_m$$

Connection assignment constraint: it ensures that each end-to-end connection can be assigned to one and only one working spanning tree.

$$\sum_k \alpha_d^k = 1 \qquad \forall d \in \mathcal{D}$$

Reconnect-link configuration constraint: it ensures that one and only one *reconnect-link* is selected to reconnect the spanning tree in case of a single link failure.

$$\sum_{l \in R_f^k} \gamma_{fl}^k = 1 \qquad \forall k, \forall f \in T^k$$

Working traffic constraints: these constraints give the working traffic on each link, and working traffic traversing any two links on each spanning tree.

$$w_m^k = \sum_{d \in \mathcal{D}; m \in P_d^k} \alpha_d^k c_d \qquad \forall k, \forall m \in T^k$$

$$w_{fm}^k = \sum_{d \in \mathcal{D}; f,m \in P_d^k} \alpha_d^k c_d \qquad \forall k, \forall f, m \in T^k$$

Backup capacity constraints: these constraints calculate the amount of backup capacity reserved on each link for full protection guarantees of fast spanning tree reconnection. The backup capacity needed on one link to protect another link is given based on the backup capacity calculation method we have described earlier.

$$r_m^f = \sum_{k; f \in T^k} \sum_{l \in R_f^k; m \in RP_{fl}^k} \gamma_{fl}^k (w_f^k - 2w_{fm}^k)^+ \quad \forall f, m \in E$$

$$r_m \geq r_m^f \qquad \forall f, m \in E$$

4.4 Proof of NP-Completeness

In this section, we analyze the complexity of the FSTR pre-configuration problem. If we assume there is only one spanning tree established in the network, the pre-configuration problem becomes how to select a *reconnect-link* for each link on one spanning tree to minimize the backup capacity, regardless of the working traffic

assignment problem.

Theorem 4.2. *FSTR pre-configuration problem on a single spanning tree is NP-Complete*

Proof. The decision version of the problem is as follows.

Instance: A Graph $G(V, E)$, a spanning tree T on G, a set of connections $\mathcal{C}$ and a given integer value ρ.

Question: Does there exist a set of reconnect links to protect all the links on the spanning tree, and the total backup capacity is no more than ρ?

The question is equivalent to another question: does there exist a set of links L, so that the graph $T + L$ is 2-edge connected and the total backup capacity is no more than ρ?

Polynomial-time verification

It is easy to see that the problem$\in NP$, since the non-deterministic algorithm can guess a set of *reconnect-link*s, and check in polynomial time whether the solution is smaller than ρ using the equation shown in the last section.

Reducibility

We reduce the 3-Dimensional Matching which is NP-Complete to the FSTR pre-configuration problem with single spanning tree by a similar reduction approach

which was used to prove the NP-Completeness of 2-edge connectivity augmentation problem in [59]. 3-Dimensional Matching problem can be described as below.

Instance: A set $M \subseteq W \times X \times Y$, where W, X and Y are disjoint sets having the same number q of elements.

Question: Does M contain a matching, that is a subset $M' \subseteq M$, such that $|M'| = q$ and no two elements of M' agree in any coordinate?

Let M be an instance of 3-Dimensional Matching with $|M| = p$ where p is the number of elements in M, and $W = \{w_i | i = 1, 2...q\}$, $X = \{x_i | i = 1, 2...q\}$ and $Y = \{y_i | i = 1, 2...q\}$, we construct a spanning tree with the vertex set

$$\begin{aligned} V =& \{r\} \cup \{w_i, x_i, y_i | i = 1, 2...q\} \\ & \cup \{a_{ijk}, \bar{a}_{ijk}, b_{ijk}, \bar{b}_{ijk} | (w_i, x_j, y_k) \in M\} \end{aligned}$$

r is the root of the tree. One vertex is established for each element in W, X, and Y. In addition for each element in M, four vertices are established represented by a_{ijk}, $\bar{a}_{ijk}$, b_{ijk} and $\bar{b}_{ijk}$. The edge set of the spanning tree is defined as

$$\begin{aligned} T =& \{(r, w_i), (r, x_i), (r, y_i) | i = 1, 2...q\} \\ & \cup \{(w_i, b_{ijk}), (w_i, \bar{b}_{ijk}) | (w_i, x_j, y_k) \in M\} \\ & \cup \{(b_{ijk}, a_{ijk}), (\bar{b}_{ijk}, \bar{a}_{ijk}) | (w_i, x_j, y_k) \in M\} \end{aligned}$$

And the set of off-tree edges are defined as

$$E - T = \{(a_{ijk}, x_j), (\bar{a}_{ijk}, y_k), (a_{ijk}, \bar{a}_{ijk}) | (w_i, x_j, y_k) \in M\}$$

Figure 4.4 shows an instance of a constructed graph to reduce 3DM problem to FSTR pre-configuration problem. Furthermore, let each neighboring node pair connected by a link on the spanning tree have a connection of 1 unit. We will show that by solving the pre-configuration problem in this network with this set of connections, we can solve the corresponding 3DM problem.

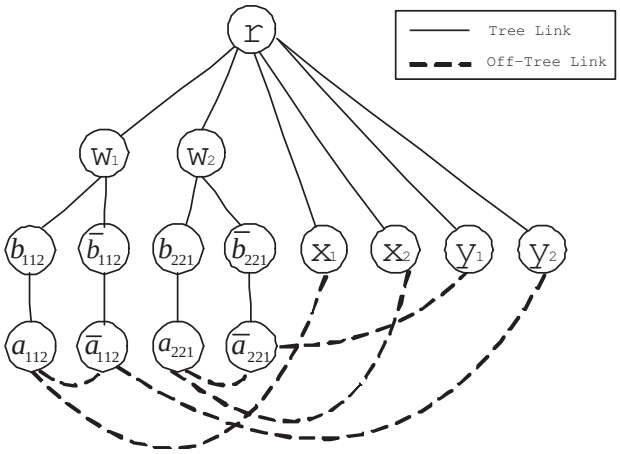


Figure 4.4: An instance of 3-D Matching: $p = q = 2$,$M = \{(w_1, x_1, y_2), (w_2, x_2, y_1)\}$

Obviously, no matter how a set of *reconnect-link*s is selected to make the tree 2-edge connected, 1 unit of backup capacity must be reserved on each link on the spanning tree. Hence the total reserved backup capacity on the spanning tree should be the number of tree links $3q + 4p$. On the other hand, if a *reconnect-link*

is used, 1 unit of backup capacity is reserved on it, and the total backup capacity reserved in the network should be $3q+4p$+Number of *reconnect-link*s. Consider in the constructed spanning tree, there are $2q+2p$ leaves. To make sure that all the links can be protected, at least $q+p$ *reconnect-link*s are needed, which means that each leaf on the spanning tree has only one *reconnect-link* incident on it. Therefore, if we assume link (m_{ijk}, x_j) is used as a *reconnect-link*, link $(m_{ijk}\bar{m}_{ijk})$ cannot be used as a reconnect link, and link $(\bar{m}_{ijk}, y_k)$ must be a reconnect link. This implies that $(w_i, x_j, y_k) \in M'$. In other words, to make $(w_i, x_j, y_k) \in M'$, link (m_{ijk}, x_j) and $(\bar{m}_{ijk}, y_k)$ must be used as *reconnect-link*, otherwise link $(m_{ijk}, \bar{m}_{ijk})$ is used as *reconnect-link*s and (w_i, x_j, y_k) is not in M'

If we can find a solution of the FSTR pre-configuration problem with the total backup capacity no more than $\rho = 4q+5p$, it means that the number of *reconnect-link*s used for protection should be and only be $q+p$. For each triple $(w_i, x_j, y_k) \in M'$, two *reconnect-link*s are involved. Let the number of triples in M' be n. The number of which not in M' should be $p-n$. Then we can get the equation $2n+p-n = p+q$, and $n = q$.

In conclusion, suppose the instance of the FSTR pre-configuration problem has a solution. It means that a set of *reconnect-link*s with the total backup capacity no more than $B = 4q+5p$ can be found. Then 3-dimensional matching has a solution which is shown by nodes using link (m_{ijk}, x_j) and $(\bar{m}_{ijk}, y_k)$ as *reconnect-link*s. By this way, we can get all triples in M', and $|M'| = q$. This concludes

our proof that the FSTR pre-configuration problem with single spanning tree is NP-Complete. □

Note that FSTR pre-configuration with single spanning tree is a special case of the general FSTR pre-configuration problem. Hence it can be deduced that FSTR pre-configuration problem is NP-Complete.

4.5 Augmentation Based Spanning Tree Reconnection Algorithm

Due to the fact that the FSTR pre-configuration problem is NP-Complete, we develop an efficient algorithm that obtain close approximation to the optimal solution. For simplification, we develop the FSTR pre-configuration algorithm with two phases: working spanning tree assignment and *reconnect-link* selection. The *reconnect-link* selection phase is equivalent to solving the FSTR pre-configuration problem on each single spanning tree.

4.5.1 Working Spanning Tree Assignment

The working spanning tree assignment problem is to assign a proper spanning tree to each connection for for transmission during normal working condition. This phase determines the working traffic on each link, so it would greatly influence the

total backup capacity reserved in a network. Based on the assumption that backup capacity sharing is allowed, intuitively, the objectives of the working spanning tree assignment are to balance the load on each link and minimize the total working traffic.

Some algorithms for working spanning tree assignment in Metro Ethernet have been proposed for load balancing or improving resource utilization [60][61]. Different from previous work, we develop a simple algorithm which is more compatible with FSTR mechanism. It successively assigns a working spanning tree to each connection with the minimum working capacity needed on the tree. The algorithm is shown as Algorithm 4.3. The worst case time complexity of the Algorithm 4.3 is $O(K|D|E)$.

Algorithm 4.3 Working Spanning Tree Assignment
Input: Connection set $\mathcal{D}$, established K multiple spanning trees
Output: Tree assignment $k_{assign}(d)$

1: **repeat**
2: Randomly select a traffic demand $d \in \mathcal{D}$
3: **for** $k = 1$ to K **do**
4: $cost_k(d) = \sum_{l \in P_d^k}(w_l + d)$
5: **end for**
6: Assign demand d to spanning tree
$k_{assign}(d) = \arg\min_k cost_k(d)$
7: **until** No traffic demand can be assigned

4.5.2 Reconnect-Link Selection

After assigning the working spanning tree to each connection, the amount of traffic on each link per tree w_l^k, and the amount of traffic traversing on any link pair per tree $w_{l,m}^k$ are obtained. With the two parameters, backup capacity reserved on each spanning tree by a given set of *reconnect-link*s can be calculated, and we can solve the FSTR pre-configuration problem on each single spanning tree. A set of *reconnect-link*s with the minimum backup capacity is selected for each tree.

As we mentioned in the last section, the FSTR pre-configuration problem on a single spanning tree is similar to the 2-edge connectivity augmentation problem [62], which can be solved by a 2-approximation algorithm in polynomial time. The main difference between the two problems is that the objective of the 2-edge connectivity augmentation problem is to minimize the total cost of the link set which makes the graph 2-edge connected (same as *reconnect-link* set). However, FSTR pre-configuration problem is to minimize the total backup capacity in the whole network. Hence in FSTR pre-configuration problem, not only the backup capacity on *reconnect-link*s, but also the backup capacity on other links should be considered. What makes the problem even more difficult is bandwidth sharing, which lets the selection of *reconnect-link*s not independent, but influenced by each other.

In our algorithm, we first construct an augmented graph from the spanning

tree and its *reconnect-link*s. A cost is given to each *reconnect-link*, which is not a single value but a vector. Then a modified 2-approximation algorithm is used to find a set of *reconnect-link*s on augmented graph with the minimum cost.

Augmented Graph Construction

Before the construction of augmented graph, we need to define the *reconnect-link path* as the path between the two end nodes of *reconnect-link* on the spanning tree, which is different from *reconnect-path*. notice that a *reconnect-link path* is only determined by a *reconnect-link*, and a *reconnect-link* only has one corresponding *reconnect-link path*, otherwise, there would be a cycle on the spanning tree

In FSTR Pre-configuration problem, we need to select a *reconnect-link* for each link on the spanning tree, but it does not mean that every link on the *reconnect-link path* of a *reconnect-link* uses that particular *reconnect-link*. For example, in Fig. 4.1, the *reconnect-link path* of $F-G$ is $F-E-C-G-H$, and $E-C$ may use $B-D$ as the *reconnect-link*, but not $F-G$. To ease the cost computation, we can modify the graph by adding some virtual *reconnect-link*s on the spanning tree to ensure that every link on each *reconnect-path* can be protected by a reconnect-link. Note that these virtual *reconnect-link*s have no physical meanings, but only represent a relationship between the protected link and actual *reconnect-link* used. Figure 4.5 shows an example of an augmented graph based on the network in Fig. 1. The dotted links represent the virtual *reconnect-link*s added in the augmented graph.

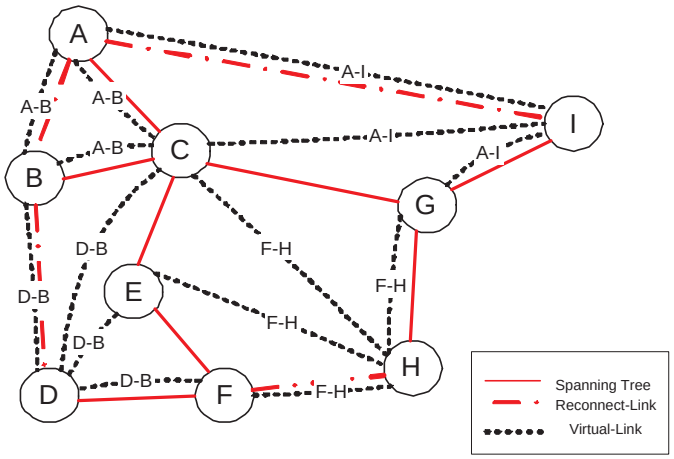


Figure 4.5: Augmented Graph

The meaning of a virtual *reconnect-link* is that all the links on the *reconnect-link path* of it are protected by the corresponding real *reconnect-link*. For instance, the meaning of link $C-D$ is that $C-E-F-D$ is protected by $D-B$ if $C-D$ is selected as a *reconnect-link*. We label '$D-B$' on the dotted line $C-D$ to indicate that the real *reconnect-link* of virtual *reconnect-link* $C-D$ is $D-B$. A link on the tree may belong to several *reconnect-link path*s, but it can be guaranteed that only one *reconnect-link* is selected to protect it by carefully assigning the link cost. The algorithm to construct the augmented graph is described as 4.4.

The Algorithm 4.4 first adds one virtual *reconnect-link* for each node pair on the *reconnect-link path* of a real *reconnect-link* r, and labels it with $'r'$. If a link on the path has only one *reconnect-link*, indicating that no other *reconnect-link*s except r can protect it, all the virtual *reconnect-link*s related to r must protect the link. Hence we delete those virtual *reconnect-link* related to r which cannot

Algorithm 4.4 Augmented Graph Generation
Input: *Reconnect-link* set R, spanning tree T
Output: Augmented Graph G'

```
1: for Each candidate reconnect-link r ∈ R do
2:    for Each node pair on reconnect-link path of r do
3:       Add one virtual reconnect-link between the node pair
4:       Label the virtual reconnect-link as 'r'
5:    end for
6:    if Any link on reconnect-link path of r has only one reconnect-link then
7:       Delete all the virtual reconnect-links which do not protect the link
8:    end if
9: end for
```

protect the particular link, and decrease the number of virtual *reconnect-link*s.

Cost Computation

The cost of a virtual *reconnect-link* in our algorithm is defined as a vector c_l, when the ith element in c_l represents the amount of backup capacity reserved on link i when l is selected as a *reconnect-link*. Denote the *reconnect-link path* of l as RLP_l. We have

$$c_l^i = \max_{m \in RLP_l} (w_m - 2w_{m,i})^+$$

Obviously for any two virtual *reconnect-link*s m and n, if $RLP_m \subset RLP_n$, then $c_m \subset c_n$. If a link belonging to RLP_n but not RLP_m has not been protected by n, *reconnect-link* m will be used to protect the other links rather than n due to the

fact that the backup capacity required by m must be no more than that required by n. Therefore, with the objective of finding the minimum backup capacity, it can be guaranteed that a link is protected by one and only one virtual *reconnect-link*. In Fig. 4.5, $C-E$ can be protected by virtual *reconnect-link* $C-D$ and $F-H$, but these two *reconnect-link*s will not be selected at the same time. Instead, $C-D$ and $C-H$ would be selected due to the reason that these two links can protect the same set of links as that can be protected by $C-D$ and $F-H$, and the cost of $C-H$ is lower than $F-H$.

Reconnect-link Selection

After constructing the augmented graph and defining the cost of each virtual *reconnect-link* on the augmented graph, we design the *reconnect-link* selection algorithm based on the 2-approximation algorithm of 2-edge connectivity augmentation problem in [62]. In the original 2-approximation algorithm, to find the minimum branching in the network, a link with the minimum additional cost is selected to break the cycle. Since the cost in our algorithm is a vector, we redefine the additional cost of an edge as the additional backup capacity when the edge is selected. Denote $l(u,v)$ the link to break the cycle, and $m(p,v)$ the link to be replaced. The additional cost of the the edge can be represented as:

$$Cost_{add}(l,m) = \sum_{i \in E}(c_l^i - \max_{p \in E} c_l^p)^+ - \sum_{i \in E}(c_m^i - \max_{p \in E-\{i\}} c_m^p)^+$$

reconnect-link selection algorithm is shown as Algorithm 4.5.

Algorithm 4.5 *reconnect-link* Selection
Input: Augmented Graph G', cost vector for each virtual reconnect-link $\vec{c}_l$, spanning tree T
Output: Selected *reconnect-link*s

1: Select a node randomly as the root of the T.
2: Direct all the edges of T towards the root.
3: For every virtual *reconnect-link* $l = (u, v)$, direct the edge from u to v, and v to u, and set cost of the two edges $\vec{c}_l$.
4: Find a minimum weight branching in the directed graph with the modified cost.
5: For each directed edge picked as a part of the branching, check the label of the virtual *reconnect-link* represented by it, and configure all the links on the *reconnect-link path* of the virtual *reconnect-link* to be protected by the corresponding real *reconnect-link*.

4.6 Performance Evaluation

To evaluate the performance of the mechanism, we obtain the optimization results and results from algorithms on various network topologies. A number of end-to-end connections are selected randomly in the network. All the connections have the same traffic amount of 1Gbps.

4.6.1 Comparison with Other Mechanism

In this section, ten groups of randomly selected connections with certain number are used in the optimization on different network topologies. The confidence intervals in all scenarios are in the range of $[-9, +9]$.

FSTR vs. $RSTP_{opt}$

We compare the FSTR mechanism with the $RSTP_{opt}$ proposed in [43], a resilient mechanism on single spanning tree. In $RSTP_{opt}$ mechanism, an optimal spanning tree is reconstructed for a particular failure, such that the network resource reservation is minimized. The original model in [43] reserves backup capacity independently for each failure scenario. We modify the model to make it support backup capacity sharing. Compared with $RSTP_{opt}$, FSTR has faster recovery speed and does not need backward learning. In addition, $RSTP_{opt}$ needs precomputing a spanning tree for each failure scenario. FSTR is simpler since only a single reconnect-link is selected in case of any single link failure.

The total backup capacity needed by $RSTP_{opt}$ and FSTR in 4×4 networks are shown in Fig. 4.6. Since the $RSTP_{opt}$ mechanism rebuilds the optimal spanning tree, it reserves less backup capacity than our mechanism. With $RSTP_{opt}$, many links may change after the failure for the optimal backup capacity saving, comparing with one link change in our mechanism. However, the performance gap

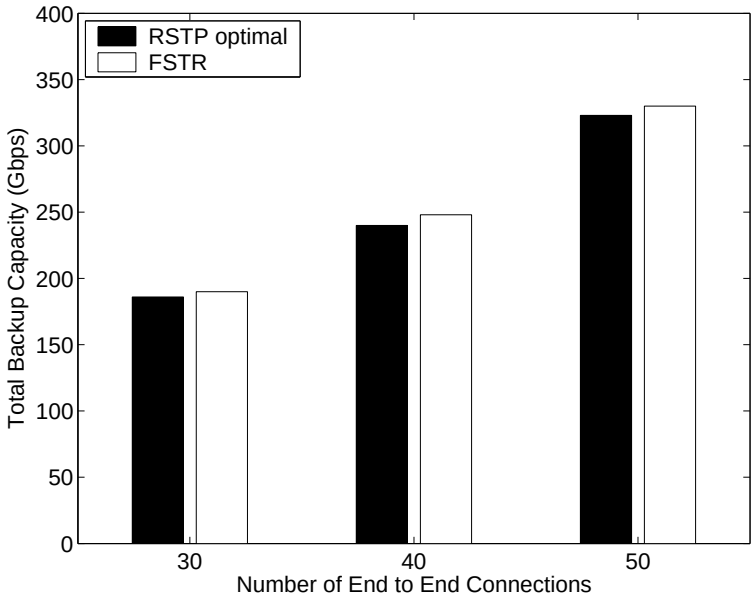


Figure 4.6: Total Backup Capacity Reserved in 4×4 Grid Network

between $RSTP_{opt}$ and FSTR is marginal. The interesting observation indicates that FSTR can have almost the same performance as $RSTP_{opt}$ by only using a single link to reconnect the spanning tree. It also supports our argument that in Metro Ethernet networks, rebuilding the whole spanning tree upon a single link failure is unnecessary. Through single link reconnection, FSTR can have faster recovery using almost the same amount of resources.

FSTR vs. Local Restoration

With multiple spanning trees built in the network, the fast spanning tree reconnection mechanism is compared with the local restoration mechanism proposed in the last chapter. The local restoration (LR) mechanism has no signaling overhead

and handles failure quickly, however, its implementation cost is higher than FSTR mechanism. The local restoration mechanism includes the connection-based and the link-based mechanisms. In the connection-based LR, each end-to-end connection independently selects a set of backup spanning trees, while in the link-based LR, a backup spanning tree is selected to protect the failure of a particular link on a working spanning tree. Obviously, the connection-based LR has more flexibility in finding backup trees, since working traffic on one link can be separated onto multiple backup trees. In the FSTR and link-based LR, aggregated traffic on the failed link of a spanning tree is rerouted to a single *reconnect-path* or backup spanning tree. On the other hand, the connection based LR is more complex than the FSTR and link based LR in terms of pre-configuration and implementation on Ethernet switches. Figure 4.7 and Figure 4.8 show the backup capacity required by three mechanisms to protect against single link failure in 4×4 and 6×6 grid networks. Four spanning trees on each topology are built by the algorithm proposed in the last chapter. The connection-based local restoration mechanism (LR connection-based) has the best performance, since in this the mechanism each connection individually selects the local backup spanning tree. FSTR reserves only around 5% more backup capacity than the connection-based, performing better than the link-based local restoration mechanism (LR link-based). About 15% of backup capacity is saved in FSTR compared with the link-based local restoration. The reason for FSTR's better performance than the link-based local restoration

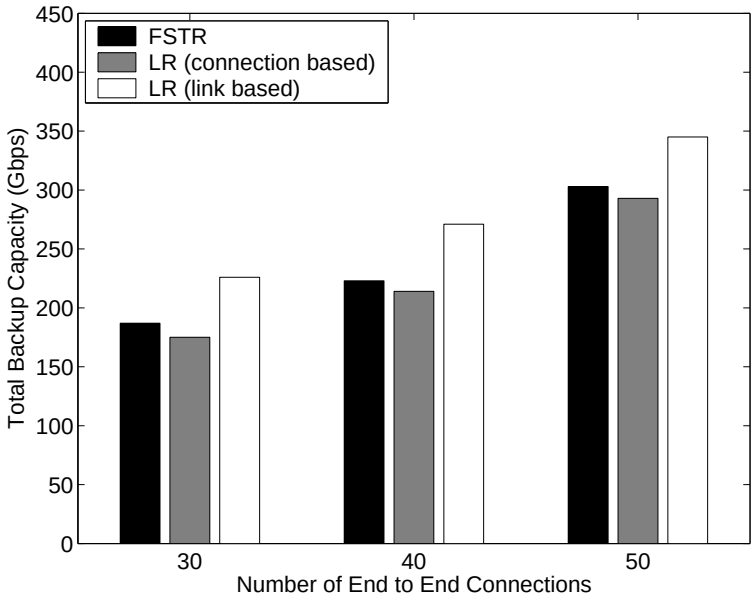


Figure 4.7: Total Backup Capacity Reserved in 4×4 Grid network

mechanism comes from the improved capacity sharing. In FSTR, the working capacity used before failure could be released to carry the rerouted traffic after failure, which saves a lot of backup capacity. However, since the traffic is restored to backup spanning trees in the LR mechanisms, working capacity on working spanning trees cannot be reused. Due to the scalability issues, the comparison of optimization results are carried out on small networks only. We believe that the capacity savings of FSTR mechanism would also hold in larger networks. It is because backup capacity is reserved on the reconnect-paths only with the FSTR mechanism. In large network where many links on the spanning tree may share the same reconnect-link, the backup capacity of these links can also be shared which leads to more efficient bandwidth utilization.

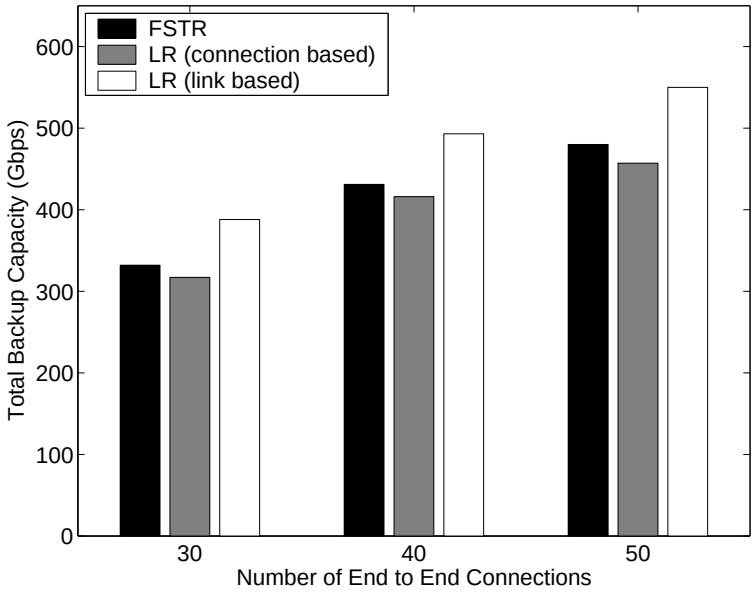


Figure 4.8: Total Backup Capacity Reserved in 6×6 Grid network

4.6.2 Performance of the Algorithm

To analyze the performance of the augmentation based algorithm, we compare its performance with the optimal solution in 4×4 and 6×6 grid topologies. We denote our augmentation based spanning tree reconnection algorithm as "AugAlgorithm" in the figures. For the single spanning tree scenario, we also show the performance of a simple greedy heuristic algorithm, which randomly picks a protected link and selects a *reconnect-link* with the minimum backup capacity for it at each step. The results on single spanning trees are shown in Fig. 4.9 and Fig. 4.10. The results on multiple spanning trees are shown in Fig. 4.11 and Fig. 4.12.

Twenty groups of randomly selected connections with certain number are used in the simulation. The confidence interval is $[-14, +14]$. It can be seen from Fig.

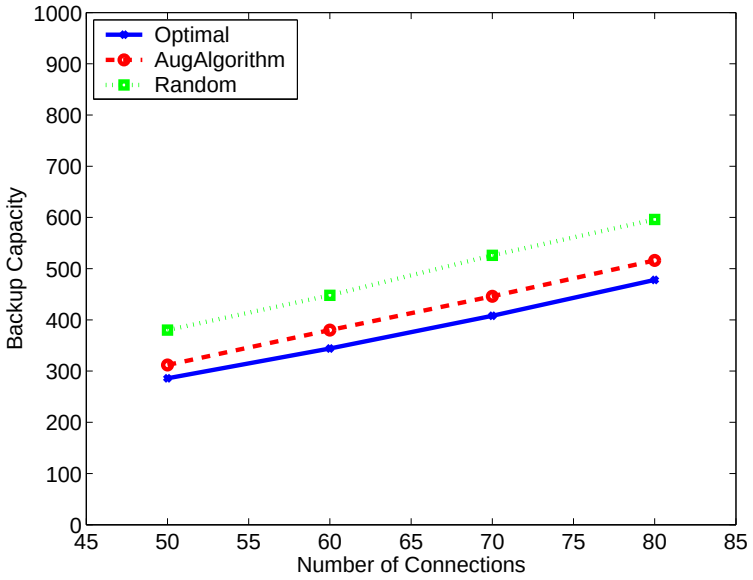


Figure 4.9: Backup capacity reserved on single spanning tree (4×4 Grid)

4.8 and Fig. 4.9 that our algorithm can obtain comparable results to the optimal solutions. In a single spanning tree scenario, the performance gap of the backup capacity obtained from our algorithm and the optimal solution is smaller than 10%, and in multiple spanning trees scenario, the gap is smaller than 15% as shown in Fig. 4.10 and Fig. 4.11. The reason for the increase of the performance gap when multiple spanning trees are used is that our algorithm solves the working spanning tree assignment and *reconnect-link* selection separately. As the matter of fact, the two problems together is a joint optimization problem. It can also be observed from the graph that, with the increasing number of spanning trees, the backup capacity reserved in the network decreases. The reduction of the backup capacity from 1 spanning tree to 2 trees is about 10%, and that from 2 trees to 3 trees is

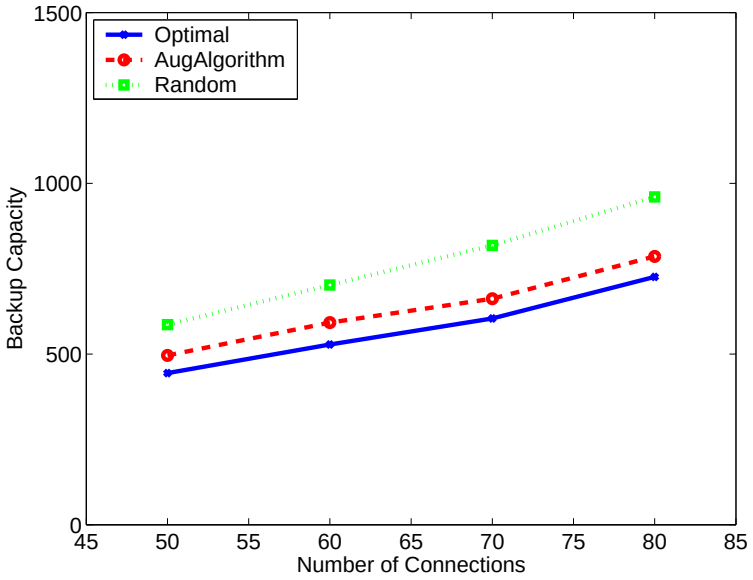


Figure 4.10: Backup capacity reserved on single spanning tree (6×6 Grid)

less than 5%, showing that with a small number of spanning trees, the minimum backup capacity necessary for guaranteed protection of a single link failure by FSTR mechanism can be achieved.

4.6.3 Recovery Time

As we described in the previous section, when an Ethernet switch detects a failure, it immediately reroutes the traffic to the *reconnect-path*. Hence the recovery time of a connection is only slightly larger than the processing time at the switch. However, providing guaranteed protection of the traffic would require a slightly longer time due to the fact that the backup capacity is reserved for the reconnected spanning tree. During the spanning tree reconnection procedure, backup capacity for some

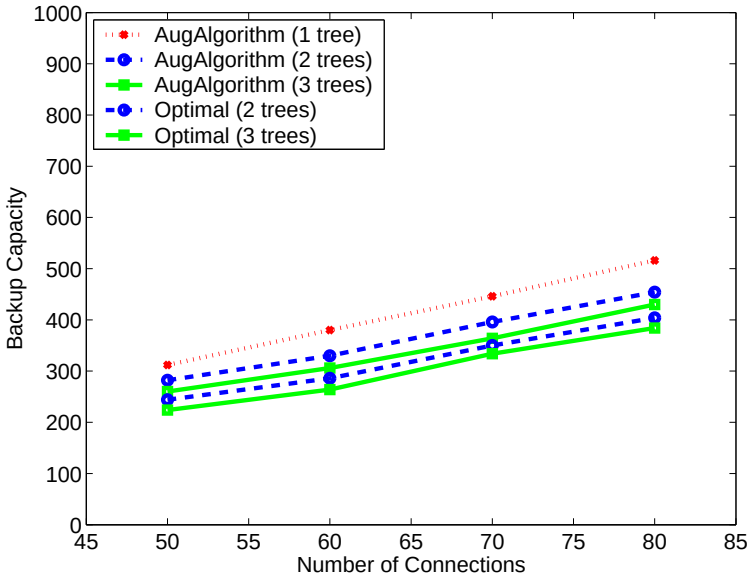


Figure 4.11: Backup capacity reserved on multiples spanning trees (4×4 Grid)

rerouted traffic may not be guaranteed. Hence, the recovery time for guaranteed protection depends on how fast the spanning tree is reconnected. Spanning tree reconnection time period is determined by the length of the *reconnect-path*. With a longer *reconnect-path*, it takes more time to notify the failure by notification messages and to reconnect the spanning tree upon failure.

In this section, we simulate the augmentation based algorithm on large network. 100-node networks generated randomly using Waxman's [58] algorithm. We constrain the maximum hop of the *reconnect-path* and study the relationship between the backup capacity reserved in the network and the length of the *reconnect-path*. We choose one thousand connections randomly selected, and 10 spanning trees are used. As shown in Fig. 4.13, if *reconnect-link*s with longer *reconnect-path*s

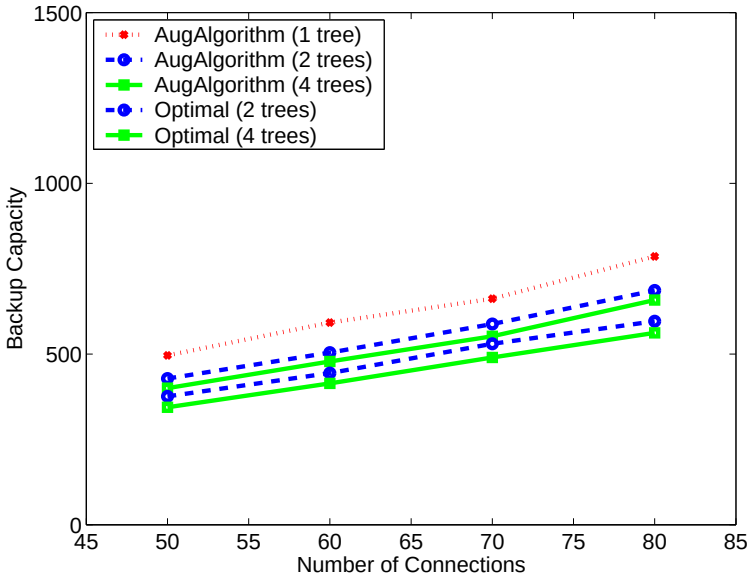


Figure 4.12: Backup capacity reserved on multiples spanning trees (6×6 Grid)

can be used in the network, the backup capacity reserved in the network would be decreased. However, *reconnect-link*s with limited length of *reconnect-path* is enough for backup capacity saving. In a network with degree 5, when the hop count constraints of a reconnect-path reaches 30, the backup capacity slowly decreases.

4.7 Conclusions

To achieve fast and guaranteed recovery for Metro Ethernet networks, a Fast Spanning Tree Reconnection (FSTR) mechanism for resilient Metro Ethernet is developed. In this mechanism, a *reconnect-link* is activated to reconnect the two separated subtrees to form a complete spanning tree in case of a single link

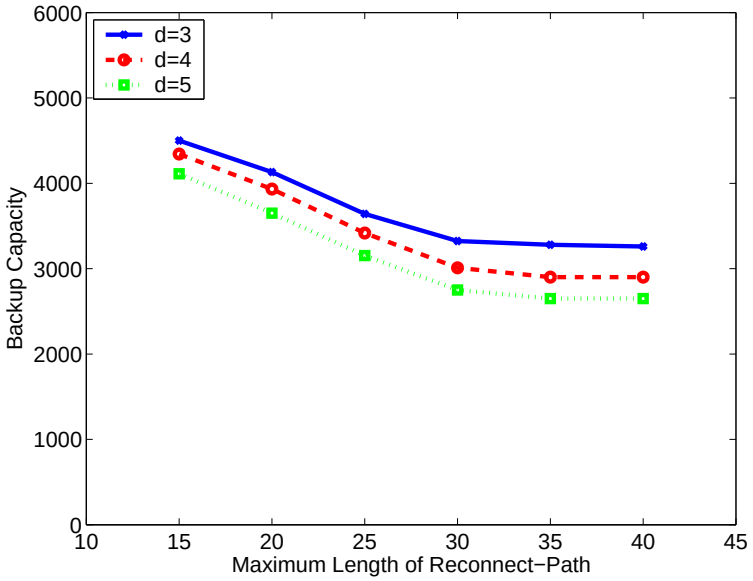


Figure 4.13: Total Backup Capacity Reserved in 100-node Network with different degree

failure. Only Ethernet switches on the *reconnect-path* need to be notified of a failure and re-configured, hence switching table flushing and backward learning are avoided. The mechanism can ensure fast failure recovery. We formulated the F-STR pre-configuration problem as an ILP problem. We prove that this problem is NP-Complete. We developed an efficient augmentation based spanning tree reconnection algorithm which is based on a classical approximation algorithm of 2-edge connectivity augmentation. The numerical results show that FSTR can have similar or considerably better performance than previous mechanisms. The results for the augmentation based algorithm show that the performance of the algorithm is close to the optimal one.

Chapter 5

Handling Double Link Failures Using FSTR

5.1 Introduction

Single link failures are the most common failure scenarios in the network, but unforeseen disasters may result in two or more link failures. Recently, there has been research interest in providing full or partial protection against double or multiple link failures in MANs and WANs. Guaranteed protection (i.e. 100% protection) for multiple link failures is expensive and difficult, since it requires too much network resources and a relatively complicated mechanism. Therefore a resilient mechanism with simplicity and fast failure recovery is also considered in Metro Ethernet networks, which may not provide full protection against all the

multiple link failure scenarios, but has the capability to provide a specific quality of protection according to the requirements from customers.

Fast Spanning Tree Reconnection (FSTR) mechanism presented in the previous chapter can efficiently handle single link failure with fast speed and guaranteed backup capacity. However, when multiple links fail in the network, the FSTR mechanism may generate unexpected loops and cannot function properly. This is due to the fact that FSTR mechanism handles each single link failure in a distributed manner. Upon multiple link failures, an affected Ethernet switch may only get limited failure information, resulting in a wrong decision on when they reconfigure their switching table. Broadcasting the failure information in the entire network seems to be the simplest way to handle this problem, however, it would greatly increase the recovery time.

In this chapter, we improve the FSTR mechanism to handle double-link failures in Metro Ethernet networks without any loops in the network. It can also avoid failure message broadcast in the network, reducing the failure handling time. In addition to 100% protection provisioning for single link failures, this mechanism can handle double-link failures with a specified protection grade requirement of a connection.

5.2 FSTR Protocol with Double-Link Failure

5.2.1 Double-Link Failures in Metro Ethernet

In addition to fully protecting single link failures, a resilient mechanism in Metro Ethernet should also have the capability to handle multiple link failures with a certain protection grade. In this paper, we focus on double-link failures.

There are totally four situations when double-link failure happens in Metro Ethernet.

- **Case 1**: The two failed links are not on the working spanning tree, so that no resilient mechanism is needed to reconnect the tree.
- **Case 2**: One failed link is on the working spanning tree, while another failed link is not, and it is not the *reconnect-link* of the first link either. In this case, FSTR mechanism for single link failure mechanism can provide the protection.
- **Case 3**: One failed link is a link of the working spanning tree, while another failed link is the *reconnect-link* of the first link. This failure can be simply protected by activating the secondary *reconnect-link*.
- **Case 4**: Two failed links are both on the working spanning tree. In this case, two *reconnect-link*s must be used to reconnect the spanning tree.

The resilient mechanism to handle double-link failures in Metro Ethernet should be able to protect against all the four kinds of failures as shown above, especially **Case 4**. However, the original FSTR mechanism cannot handle failures in **Case 4**. It is because that the mechanism handles link failures in a distributed manner, and each switch does not know which failure situation has happened. In the FSTR mechanism, when an Ethernet switch detects a link failure adjacent to it, it treats the failure as a single link failure without any knowledge on possible failures in other parts of the network. Thus, it may notify a wrong *reconnect-link* to reconnect the spanning tree and generate a loop in the network. Figure 5.1 illustrates how a loop is formed when double-link failure happens. Assume that link $C - F$ is pre-configured as *reconnect-link* of link $B - C$ as shown in Fig. 5.1 (a). Based on FSTR mechanism, Ethernet switch B will send a notification knowledge to F when $B - C$ fails, and then $C - F$ will be activated to reconnect the spanning tree. In the same way, we assume link $C - G$ is the *reconnect-link* of $D - F$ (Fig. 5.1 (b)). When the failure of single link $B - C$ or $D - F$ happens, FSTR mechanism can reconnect the spanning tree with the proper *reconnect-link*. However, when link $B - C$ and $D - F$ fail simultaneously, node F does not know the failure of link $B - C$, and activates the *reconnect-link* $C - F$. At the same time, C will activate the *reconnect-link* $C - G$, which will generate a circle between switches C, F and G (Fig. 5.1 (c)).

A possible way to solve the problem is to let every Ethernet switch adjacent

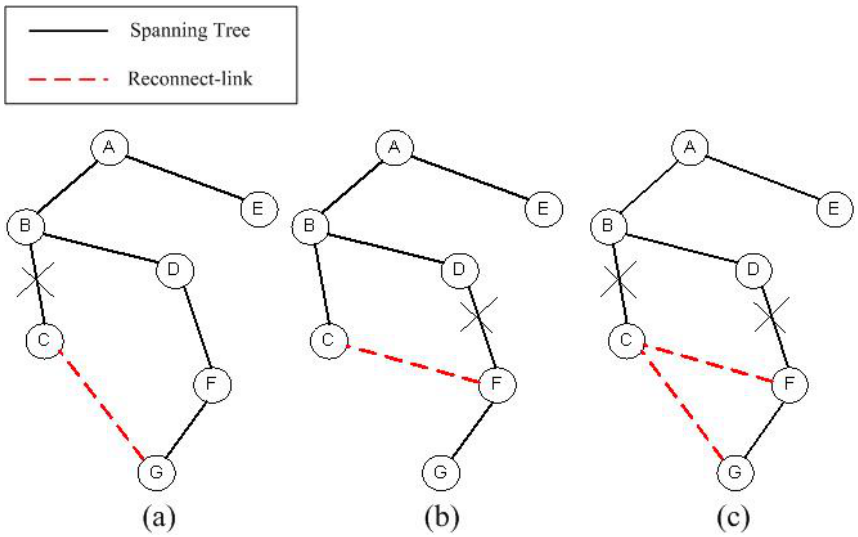


Figure 5.1: (a) link $C-F$ is used as the *reconnect-link* when link $B-C$ fails (b) link $C-G$ is used as the *reconnect-link* when link $D-F$ fails (c) an unexpected loop is formed when link $B-C$ and link $D-F$ fail.

to failed links broadcast a failure message throughout the whole network. Then switches can make the correct decision when they receive all the failure messages, and a loop-free reconnection solution could be found. However, this method takes several Round Trip Times during failure message broadcast and spanning tree reconnection, which can hardly meet the fast failure recovery requirement in Metro Ethernet. We therefore develop a different method to handle the double-link failure problem. By carefully pre-configuring the *reconnect-link*s, each Ethernet switch adjacent to the failed link could handle the failure independently, treating the failure as a single link failure while ensuring that no loops are generated in the network at the same time.

In Fig. 5.2, another possible pre-configuration of the *reconnect-link*s is shown.

The *reconnect-link* of $B - C$ is changed to $C - D$ instead of $C - F$, while the *reconnect-link* of $D - F$ remains to be $C - G$. When the two links $B - C$ and $D - F$ fail, B notifies D to activate $C - D$, and D notifies G to activate $C - G$ via the path $D \rightarrow C$. Thus the spanning tree is reconnected without any loops. Note that node B does not know the failure of $D - F$ during the reconnection procedure, only handling the failure of link $B - C$ as a single link failure. The example shows that broadcasting failure message all over the network to handle double-link failures is unnecessary. The *reconnect-link*s can be pre-configured in a way that loops are avoided even though each failure is handled independently. Next we present the necessary and sufficient condition for loop avoidance of FSTR mechanism in Metro Ethernet.

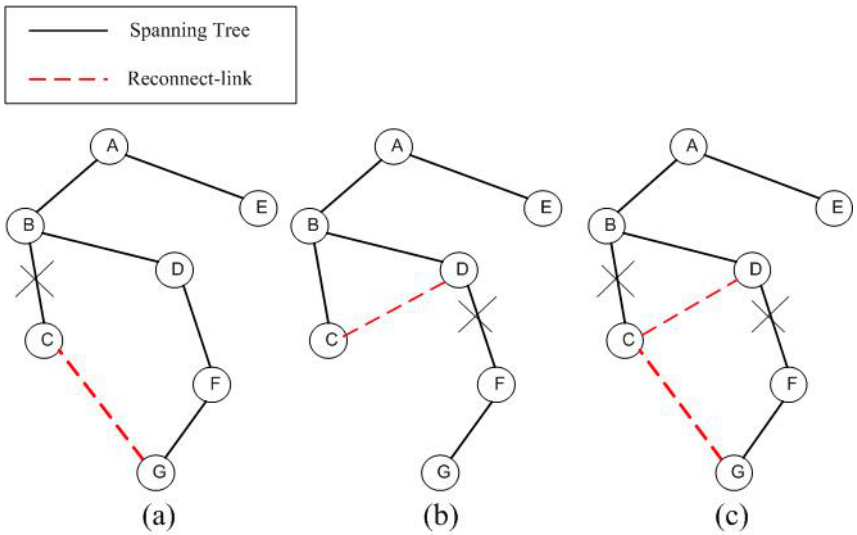


Figure 5.2: (a) link $C - D$ is used as the *reconnect-link* when link $B - C$ fails (b) link $C - G$ is used as the *reconnect-link* when link $D - F$ fails (c) the spanning tree can be safely reconnected without any loop

5.2.2 Loop Free Condition of Handling Double-Link Failure

First We recall that the *reconnect-path* is defined the path between the two end nodes of a protected link on the reconnected spanning tree upon the link failure. It is determined by the protected link and the *reconnect-link*. For example, in Fig. 5.2, *reconnect-path* of link $B-C$ is $B-D-C$, and that of $D-F$ is $D-B-C-G-F$. Obviously, the end nodes of protected link and *reconnect-link* are all on the corresponding *reconnect-path*.

Theorem 5.1. *Let RP_i be the reconnect-path of link i on a spanning tree and let m and n be two links on a spanning tree. Loops cannot be created by double-link failures of m and n during spanning tree reconnection if and only if for m and n, the following condition are met:*

1. *If $m \in RP_n$, then $n \notin RP_m$*
2. *If $n \in RP_m$, then $m \notin RP_n$*

Proof. Assume that links $A-B$ and $M-N$ are on a spanning tree. Further assume that r_{A-B} and r_{M-N} are the *reconnect-link*s of $A-B$ and $M-N$.

If the condition 1 and 2 are not satisfied, we have $A-B \in RP_{M-N}$ and $M-N \in RP_{A-B}$. Assume that the length of RP_{M-N} is more than RP_{A-B}. Since $A-B \in RP_{M-N}$, then A, B, M and N which are end nodes of the two links all

belong to RP_{M-N}. According to the definition of a spanning tree, there is only one path on the spanning tree between any node pair, thus any node between M and A or N and B on the reconnected tree upon failure of $A-B$ should belong to RP_{M-N}. Otherwise, there will be two paths between a node pair on the spanning tree. On the other hand, since $M-N \in RP_{A-B}$, any nodes between M and A or N and B on the reconnected tree upon failure of $A-B$ should also belongs to RP_{A-B}. Due to the assumption that RP_{M-N} is longer than RP_{A-B}, it can be concluded that if a node belongs to RP_{A-B}, then it belongs to RP_{M-N}. Because the two end nodes of r_{A-B} are on RP_{A-B}, they must be on RP_{M-N}. Upon failure of $A-B$ and $M-N$, r_{M-N} will connect two nodes on RP_{M-N}, and r_{A-B} will also connect two nodes on RP_{M-N}. r_{M-N}, r_{A-B} and RP_{M-N} will generate a circle as shown in Fig. 5.3. Therefore, the loop will be formed if condition 1 and 2 are not satisfied.

If the two links satisfy the condition that $A-B \in RP_{M-N}$ and $M-N \notin RP_{A-B}$. Since $M-N \notin RP_{A-B}$, failure of $M-N$ does not affect the reconnection upon failure of $A-B$. Hence reconnecting the spanning tree with r_{A-B} will not generate a loop. Also because of $M-N \notin RP_{A-B}$, there are at least one end node of $M-N$ not on RP_{A-B}. Let node M be the node not on RP_{A-B}, every node between M and one end node of reconnect link r_{M-N} do not belong to RP_{A-B} either. Otherwise, these links would form a loop in a spanning tree, which is contradictory to the definition of spanning tree. Since these nodes on RP_{M-N} do not belong to RP_{A-B},

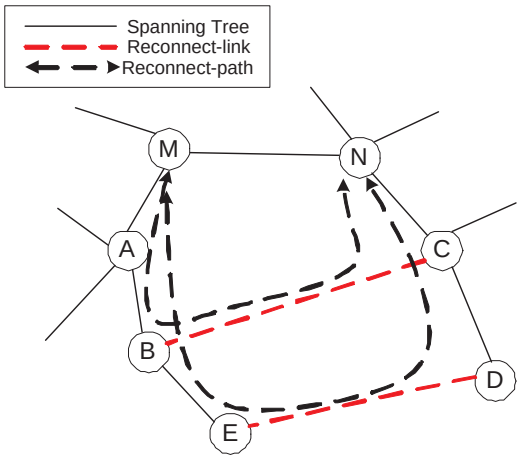


Figure 5.3: *Reconnect-link*s: $r_{M-N} = E - D$ and $r_{A-B} = B - C$. *Reconnect-path* RP_{M-N} is $M \leftrightarrow A \leftrightarrow B \leftrightarrow E \leftrightarrow D \leftrightarrow C \leftrightarrow N$; *Reconnect-path* RP_{A-B} is $A \leftrightarrow M \leftrightarrow N \leftrightarrow C \leftrightarrow B$.

it can be concluded that at least one end node of r_{M-N} is not on RP_{A-B}. Therefore when spanning tree is reconnected upon failure of $A-B$ and $M-N$, r_{M-N} is not a link between two nodes of RP_{A-B} and will not generate a loop with RP_{A-B} and r_{A-B}. □

Obviously, we have $B - C \in RP_{D-F}$ and $D - F \notin RP_{B-C}$ in Fig. 5.2. The condition is satisfied and no loop exists. It also can be seen that the example in Fig. 5.1 does not satisfy the conditions in Theorem 5.1

5.2.3 Loop Free Condition of Handling General Failure Scenarios

The loop free condition for double-link failures cannot be directly used in the general multiple failure scenarios. Consider a single node failure which results in all the links adjacent to the node fail. If the number of links adjacent to the node is M, the number of links to reconnect the spanning tree should be $M-1$ but not M. If loop free condition for double-link failure is used for pre-configuration of single node failure, there would be a loop in the network.

The loop free condition for handling general failure scenarios is given below:

Theorem 5.2. *Assume a failure in Metro Ethernet networks which separates the original spanning tree into M subtrees, the sufficient and necessary condition for loop free topology with FSTR is that there must be at least one* reconnect-link *connecting each sub tree.*

Proof. Assume that a general network failure happens which can be single link, multiple link, or node failures. The original spanning tree is separated into several subtrees. The number of subtrees depends on the number of failed links. If M links fail in the network, the spanning tree is divided into $M+1$ subtrees as long as no node failure happens. If a node failure happens and the degree of node is M, the spanning tree is divided into M subtrees. To ensure that no loops are formed with FSTR mechanism, the reconnect-link to reconnect these subtrees must be

configured in a way that these sub trees cannot be formed into a circle.

Assume that there are M subtrees after a failure. $M-1$ and only $M-1$ *reconnect-link*s are required to reconnect these sub trees. With $M-1$ *reconnect-link*s, there must be one or several isolated subtrees if there is a loop formed by the *reconnect-link*s. Hence, to ensure loop free condition with FSTR, it should be ensured that no subtree is isolated or disconnected by the *reconnect-link*s, which means that there must be at least one *reconnect-link* to connect each sub tree. □

Theorem 5.2 can help resolve the loop problem upon general failure scenarios, including node failure and multiple link failures. However, we mainly focus on double-link failure scenario which can use Theorem 5.1 to solve the loop problem in this chapter, since the probability that more than two links fail in the network is trivial. The FSTR mechanism for multiple link failures is discussed in the next chapter.

Our mechanism to handle double-link failure in Metro Ethernet is based on FSTR mechanism, but with a smarter pre-configuration of *reconnect-link*s. When single link failure or the failure as in **Case 1** and **Case 2** happens, the new mechanism has the same procedure as the original FSTR mechanism to reconnect the spanning tree. To handle failure as in **Case 3**, two different *reconnect-links* are pre-configured for each protected link on the spanning tree. If the protected link and one of its *reconnect-link*s fail at the same time, end nodes adjacent to the failed

reconnect-link will send a message to the end nodes of the protected link, asking them to use secondary *reconnect-link*. The new *reconnect-link* then can be used to reconnect the spanning tree. To handle failures as in **Case 4**, the pre-configuration of *reconnect-link*s should make the protected link pairs on the spanning tree satisfy the condition in Theory 5.1. The mechanism uses the same failure notification and switching table reconfiguration method as the FSTR mechanism, therefore it also has fast recovery speed and low signaling overhead.

However, there is possibility that due to the topology and spanning tree constraints, some protected link pairs on the spanning tree do not satisfy the conditions in Theorem 5.1, and their failure cannot be protected. Our mechanism for double-link failure can provide 100% protection for single link failures, and can provide partial protection for double-link failures. To achieve optimal performance, it should protect the links that are more fragile or more necessary to be protected. In the next section, we formulate the problem of *reconnect-link* pre-configuration for double-link failures so that the protection grade guarantees in the network can be met.

5.3 Problem Formulation

5.3.1 Failure Patterns

We formulate the problem under the assumption that network link failures happen independently and constitute a memoryless process [63]. Denote f_i the failure of single link i, and f_{ij} the failure of link i and link j, and the scenario of no failure is denoted by f_0. In addition, the stationary distribution of f_i and f_{ij} are given by $\pi(f_i)$ and $\pi(f_{ij})$. If we only consider single and double-link failures, there should be $\pi(f_0) + \sum \pi(f_i) + \sum \pi(f_{ij}) = 1$.

5.3.2 Definition of Protection Grade

A protection grade of a connection is defined as the probability that the connection can be protected by the mechanism. It also represents the percentage of the connection's traffic that can be protected, under the assumption that the failure patterns constitute a memoryless process. In case of a failure, spanning trees should be reconnected and bandwidth provision should be reconfigured to guarantee the protection grade requirement.

Denote $\rho_d(f_{ij})$ the percentage of bandwidth that connection d is provided in case of failure f_{ij}, there should be $\rho_d(f_{ij}) \in [0, 1]$. Since our mechanism can fully protect single link failure, the total protection grade the network can provide to a

particular connection c is $\sum \pi(f_{ij})\rho_d(f_{ij}) + \sum \pi(f_i) + \pi(f_0)$, which must be equal to or larger than the specified protection grade.

5.3.3 Integer Linear Programming Model

We formulate the problem as follows which is similar as the FSTR pre-configuration problem: **Given one or more spanning trees established in a network and a set of end-to-end connections, assign each connection to a proper working spanning tree, and select the best *reconnect-link*s to reconnect each spanning tree to satisfy the protection grade for each connection, such that the backup capacity reserved in the network is minimized.**

To calculate the backup capacity required for a particular failure on each link, we first get the working capacity on each link based on the working spanning tree assignment. Then the traffic of each link on the reconnected spanning tree upon the particular failure can be obtained, and the backup capacity required on the link is the additional traffic when the spanning tree is reconnected. Several parameters need to be computed before solving the model, including *reconnect-link* set R_f^k for each link f, *reconnect-path* RP_{fl}^k for each link f and its *reconnect-link* l, and reconnected spanning tree $T_{f_i,l}^k$. The variables to be determined by the ILP solver are:

α_d^k : binary variable, $\alpha_d^k = 1$ if connection d uses spanning tree k as working

spanning tree

$\gamma_{f_i l}^{k,0}$: binary variable, $\gamma_{f_i l}^{k,0} = 1$ if on spanning tree k, link l is the primary *reconnect-link* upon failure of link i

$\gamma_{f_i l}^{k,1}$: binary variable, $\gamma f_i l^{k,1} = 1$ if on spanning tree k, link l is the secondary *reconnect-link* upon failure of link i to handle **Case 3** failure.

The objective function of the ILP model can be expressed as:

$$\min \sum_{m \in E} r_m$$

Similar to the constraints for the FSTR, we have the constraints for working spanning tree assignment and *reconnect-link* pre-configuration. Only one working spanning tree is assigned to each connection. Only one primary and one secondary *reconnect-link* are assigned to each protected link on the spanning tree.

$$\sum_{k} \alpha_d^k = 1 \qquad \forall d \in \mathcal{D}$$

$$\sum_{l \in R_f^k} \gamma_{f_i l}^{k,0} = 1 \qquad \forall k, \forall i \in T^k$$

$$\sum_{l \in R_f^k} \gamma_{f_i l}^{k,1} = 1 \qquad \forall k, \forall i \in T^k$$

The backup capacity required for single link failure is calculated as follows. It is also the backup capacity required for failure as in **Case 2**. We first calculate

the working traffic on each link, which is the traffic without any failure. Then the traffic on the reconnected tree upon a particular failure can be obtained. The backup capacity reserved for the failure should be the additional traffic traversing the link after the spanning tree is reconnected.

$$w_m^{f_0} = \sum_k \sum_{d \in D} \alpha_d^k P_d(T^k, m) c_d \qquad \forall m \in E$$

$$w_m^{f_i} = \sum_k \sum_{d \in D} \sum_{l \in R_{f_i}^k} \alpha_d^k \gamma_{f_i l}^{k,1} P_d(T_{f_i,l}^k, m) c_d \quad \forall m, i \in E$$

$$r_m^{f_i} = (w_m^{f_i} - w_m^{f_0})^+ \qquad \forall m, i \in E$$

To handle failures as in **Case 3**, the backup capacity using the secondary *reconnect-link* should be reserved independently. Since the primary and secondary *reconnect-link* cannot fail concurrently, the backup capacity reserved in the network when the secondary *reconnect-link* is activated can be shared with the backup capacity for the primary *reconnect-link*. Similar equations for single link failure as above are used. The variable $\gamma_{f_i l}^{k,0}$ is replaced by $\gamma_{f_i l}^{k,1}$, and one additional condition ensures that the *reconnect-link* and the secondary *reconnect-link* of a protected link

are not the same.

$$w_m^{f_{ij}} = \sum_{k;j\in R_{f_i}^k} \sum_{d\in D} \sum_{l\in R_{f_i}^k} \alpha_d^k \gamma_{f_i l}^{k,1} P_d(T_{f_i,l}^k, m) c_d \quad \forall m,i,j \in E$$

$$\gamma_{f_i l}^{k,0} \gamma_{f_i l}^{k,1} = 0 \qquad \forall l \in E, i \in T^k$$

For the failure as in **Case 4**, the backup capacity required for a particular failure can be calculated as follows.

$$w_m^{f_{ij}} = \sum_{k;i,j\in T^k} \sum_{d\in D} \alpha_d^k \rho_d(f_{ij}) P_d(T_{f_{ij},lt}^k, m) c_d \qquad \forall m,i,j \in E$$

$$r_m^{f_{ij}} = (w_m^{f_{ij}} - w_m^{f_0})^+ \qquad \forall m,i,j \in E$$

Further, we have the constraints that the conditions for double-link failures in Theorem 5.1 should be satisfied. When the conditions are not satisfied, $\rho_d(f_{ij}) = 0$ which means that connection c cannot be protected upon failure f_{ij}.

$$\rho_c(f_{ij}) \leq 1 - \sum_{k;i,j\in T^k} \alpha_c^k \Big(\sum_{m\in R_{f_i}^k; j\in RP_{f_i m}^k} \gamma_{f_i m}^{k,0} \sum_{n\in R_{f_j}^k; i\in RP_{f_j n}^k} \gamma_{f_j n}^{k,0} \Big)$$

$$\forall d \in D \quad \forall i,j \in E$$

The backup capacity reserved on a particular link should be the maximum value among all the failure scenarios.

$$r_m = max\{r_m^{f_i}, r_m^{f_{ij}}\} \qquad \forall m \in E$$

Finally, we have the constraint that the protection grade of each connection must meet a given requirement:

$$\sum_{f_{ij} \in F} \pi(f_{ij})\rho_d(f_{ij}) + \sum_{f_i \in F} \pi(f_i) + \pi(f_0) \leq A_d \quad \forall d \in D$$

5.4 Performance Evaluation

We study the efficiency and characteristics of backup capacity provisioning when the proposed mechanism is used to handle double-link failures in Metro Ethernet. We used 4×4 grid network and select a number of end-to-end connections randomly in the network. We assume all the connections have the same traffic demand of 1Gbps.

We assume that the failure of a link in the network is independent of other link failures. We let the failure probability of a particular link i is $Pr(i)$ which is randomly selected from $\{0.01, 0.001, 0.0001\}$. Since only the failure of single and double link are considered, We can get $\pi(f_i) = Pr(i) \prod_{j \in E-\{i\}} (1 - Pr(j))$ and

$\pi(f_{ij}) = Pr(i)Pr(j)\prod_{t\in E-\{i,j\}}(1-Pr(t))$.

We establish four spanning trees in the network according to the spanning tree algorithm used in Chapter 3, and let all the connections have the same protection grade requirement of 99% to get the reserved backup capacity. Then the connection grade requirement for all the connection is set to 99.9%.

Figure 5.4 shows the backup capacity reserved in the 16-node grid network with different number of connections. When the protection grade requirement of the connections increase, the total backup capacity reserved in the network increases, due to the fact that more double-link pairs need to be protected to meet the requirement of protection grade requirement.

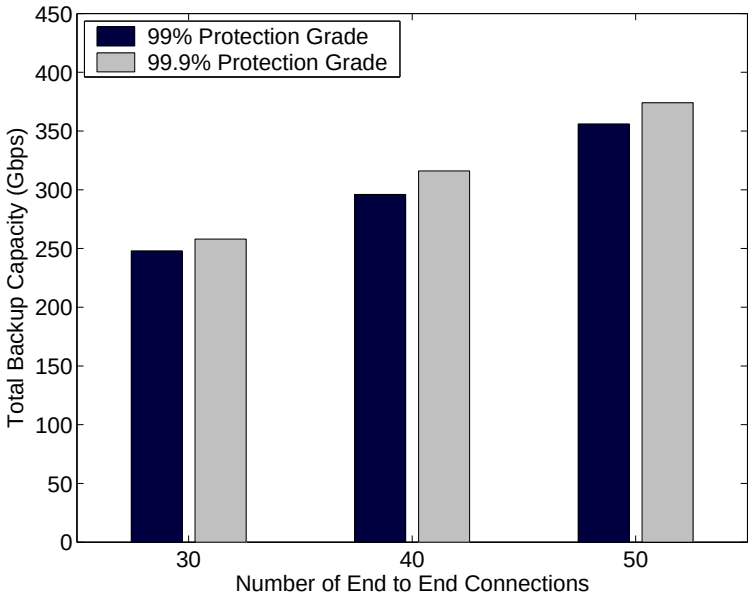


Figure 5.4: Backup capacity with different protection grade constraints.

The cumulative distribution function (CDF) of the protection grade of all connections under the constraints that protection grade of each connection must be larger than 99.9% is shown in Fig. 5.5. It can be seen that the distribution of

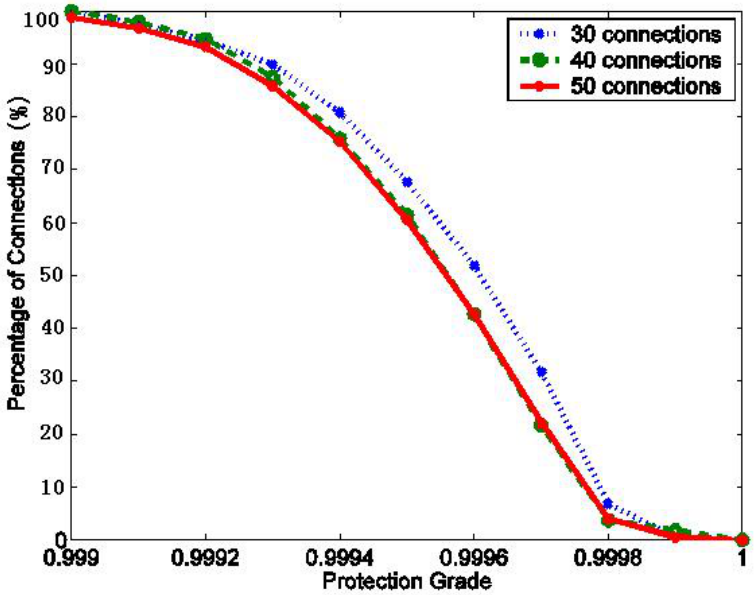


Figure 5.5: CDF of protection grade with different number of connections under the 99.9% protection grade constraints.

the protection grade is almost the same regardless of the number of connections routed in the network. About 50% of the connections can have a protection grade higher than 99.95%, and 25% of the connections have protection level higher than 99.97%. It indicates that even though the protection grade requirement is 99.9%, most of the connections are provided with much higher protection grade.

5.5 Conclusions

By carefully pre-configuring *reconnect-link*s, Fast Spanning Tree Reconnection mechanism can solve the loop problem when handling double-link failures. We developed a *reconnect-link* pre-configuration model to provide a specified protection grade for the connections. The numerical results show that our model has the capability to pre-configure the *reconnect-link*s such that the protection requirement for each connection can be satisfied with efficient utilization of the backup capacity.

Chapter 6

Survivability in Ethernet over WDM Optical Networks

6.1 Introduction

With optical interfaces and wavelength division multiplexing technologies, Ethernet can provide tens of Gbps or even higher transmission rate at low price. This makes Ethernet over Wavelength Division Multiplexing (WDM) networks a suitable candidate for metro and carrier grade networks.

The introduction of Ethernet over WDM results in a traditional two layer model, optical layer and Ethernet layer. An arbitrary logical Ethernet topology consisting of one or multiple logical spanning trees is built based on lightpahts. These logical spanning trees are independent of the underlying physical topology. Failure

handling can be done at the optical layer or Ethernet layer. Survivability in Ethernet over WDM networks differs from that in IP over WDM networks that has been widely studied. Since the logical topologies in Ethernet over WDM networks are spanning trees which are 1-connected, any single logical link failure will disconnect the logical spanning trees.

In Ethernet over WDM networks, a single physical fiber cut may result in multiple logical link failures at the Ethernet layer that cannot be resolved merely at the Ethernet layer. Hence lightpath routing of logical spanning trees over a physical topology should coordinate with Ethernet layer survivability mechanisms to ensure that any single fiber cut on a physical topology does not lead to unresolved failures at the Ethernet layer. In this chapter, we develop Fast Virtual Spanning Tree (FVSTR) mechanism for Ethernet over WDM optical networks. We illustrate the constraints of survivable routing of logical spanning trees over a physical topology. We present the integer linear programming model for computing the lightpath routing and FVSTR configuration. We use the terms logical link, virtual link, and lightpath interchangeably in this thesis.

6.2 Ethernet over WDM model

The introduction of Ethernet over WDM network results in a two layer-optical layer and Ethernet layer-vertical model shown in Fig. 6.1. We consider that optical layer

consists of optical fibers and optical switches. Lightpaths with a certain capacity (e.g. 1Gbps or 10Gbps) are established on the physical topology. Each lightpath can start at, terminate at, or bypass the optical switches. Ethernet switches are interconnected via lightpaths. Hence one or multiple logical spanning trees can be built by using the lightpaths. Note that the structures of these spanning trees are arbitrary and independent of the underlying physical topology. We assume that the Ethernet over WDM network architecture uses peer model, hence the optical layer and Ethernet layer can exchange information at both layers.

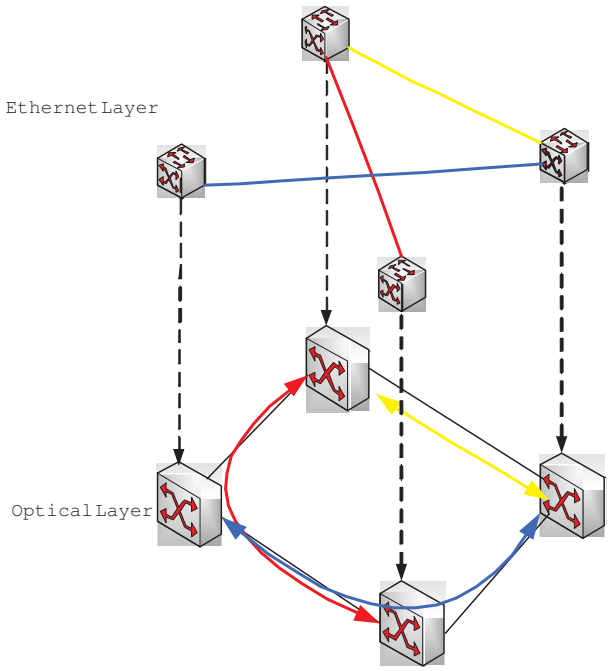


Figure 6.1: Illustration of Ethernet over WDM model

Survivability schemes for Ethernet over WDM networks can be applied at the optical layer, Ethernet layer, or both layers. Survivable routing in Ethernet over

WDM networks is defined as lightpath routing with which spanning trees are still connected upon any single fiber cut. When survivability schemes at the optical layer are used, two link-disjoint lightpaths for each logical spanning tree link should be established. When there is a fiber cut on the primary lightpath of the logical spanning tree, the backup lightpaths take the responsibility to carry the traffic on the primary lightpath. Survivability schemes at the optical layer have fast recovery speed but use more network resources. When survivability schemes at the Ethernet layer are implemented instead, Ethernet switches should rebuild the spanning trees upon logical link failures on the logical spanning trees resulting from physical fiber cut. Survivability schemes at the Ethernet layer has the drawback of low speed spanning tree re-convergence. In addition, logical links not on a spanning tree need to be established for spanning tree reconnection.

6.3 FVSTR Mechanism for Ethernet over WDM Networks

Fast Virtual Spanning Tree Reconnection (FVSTR) mechanism works in a similar way as FSTR mechanism at the Ethernet layer. It can solve the problem of long recovery time inherent in previous Ethernet layer survivability schemes. At the same time, it uses less network resources, including capacity and transceivers,

than optical layer survivability schemes. To use Fast Virtual Spanning Tree Reconnection mechanism, the optical network must create new lightpaths which are logical *reconnect-link*s currently not on the failed logical spanning tree. Ethernet switches should be configured to use these logical *reconnect-link*s against failures of the corresponding logical links on logical spanning tree. Figure 6.2 shows an example of the FVSTR implementation on Ethernet over WDM networks. The solid lines represent logical links on a logical spanning tree, and the dash lines represent logical *reconnect-link*s. In the optical layer, the routing of each logical link is shown with the same line. FVSTR configuration in the figure shows how the spanning tree is reconnected after failure. For example, "$A - B - A - D$" represents that logical link $A - D$ would be used as the logical *reconnect-link* to reconnect the spanning tree upon failure of link $A - B$.

When FVSTR mechanism is used in Ethernet over WDM networks, a single fiber cut at the physical topology may lead to multiple logical link failures at the Ethernet layer due to the fact that these multiple logical links might traverse the same single fiber at the optical layer. Multiple logical link failures pose two major problems at the Ethernet layer when the FVSTR mechanism is used:

1. A spanning tree is unable to be reconnected if the logical link on the spanning tree and its logical *reconnect-link* fail simultaneously.

2. Loops can be generated if multiple logical links on the spanning tree fail

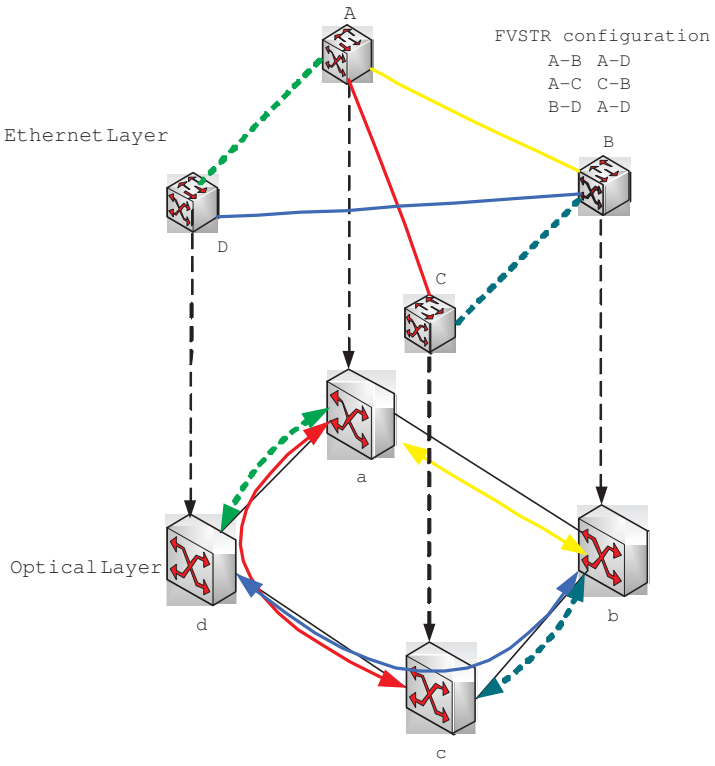


Figure 6.2: Illustration of FVSTR mechanism on Ethernet over WDM networks: $A-D$ is the logical *reconnect-link* upon failure of $A-B$ or $B-D$; $B-C$ is the logical *reconnect-link* upon failure of $A-C$

simultaneously.

To solve the above two problems, coordination of lightpath establishment at the optical layer and FVSTR configuration at the Ethernet layer is required. Since we assume that Ethernet over WDM networks are implemented using peer model, optical layer and Ethernet layer can exchange routing and FVSTR configuration information with each other for coordination.

At the optical layer, the routing of lightpaths should ensure that each logical

link on the spanning tree and its *reconnect-link*s are not routed through the same fiber, otherwise the failure of a logical link cannot be recovered by FVSTR. At the Ethernet layer, the FVSTR configuration must make sure that each logical link on the spanning tree has a *reconnect-link*. If several logical links on the spanning tree fail concurrently due to the single fiber cut, the reconnection by using their logical *reconnect-link*s should not generate any loop. This can be ensured by selecting the *reconnect-link*s according to Theorem 5.2 stated in the last chapter. In addition, FVSTR configuration should select the logical *reconnect-link* for each logical link on the spanning tree such that they do not fail simultaneously.

Figure 6.2 shows an example of successful lightpath routing and FVSTR configuration in Ethernet over WDM networks, including a spanning tree and two *reconnect-link*s. It can be seen that if fiber $d-c$ fails, logical link $A-C$ and $B-D$ will fail simultaneously. This failure scenario can be recovered by using the logical *reconnect-link* $A-D$ for the failure of $B-D$ and $B-C$ for failure of $A-C$. Note that these two logical *reconnect-link*s are not routed over the physical fiber $d-c$. From the example we can observe that, in addition to lightpath routing of logical links on the spanning tree, logical *reconnect-link*s should be configured at the Ethernet layer, and their routing must be established through the optical layer.

Note that the lightpath routing at the optical layer and FVSTR configuration at the Ethernet layer have an impact on each other. Therefore, they need to be

considered as a joint optimization problem. In the next section, we would the problem and give the mathematical model.

6.4 Problem Formulation

We formulate the problem as follows: **Given one or more logical spanning trees, find logical *reconnect-link*s corresponding to the failure of each logical link on the spanning tree, and find lightpath routing of logical spanning tree links and *reconnect-link*s to protect against any single fiber cut, such that the total number of wavelengths on the physical fibers in the network is minimized.**

We assume that all the logical links are bi-directional, and a logical link between nodes i and j is represented as (i, j). We assume each bi-directional physical link consists of two arcs in the opposite direction and has the same capacity. Arc from node i to j is represented as $\vec{p}(i, j)$. We also assume that the logical spanning tree covers all the nodes in the network, which means that at least one lightpath terminates and starts at each node. Since logical *reconnect-link*s for spanning tree can be built between any two nodes, we pre-compute the *reconnect-link* set R_{st} for each logical link (s, t) on the spanning tree including all the possible *reconnect-link*s of (s, t). Then, for each *reconnect-link* (s', t') in R_{st}, we pre-compute its corresponding *reconnect-path* $RP_{st}^{s't'}$. Two variables need to be determined:

$\gamma_{st}^{s't'}$: binary variable, $\gamma_{st}^{s't'} = 1$ if logical link (s', t') is built as logical *reconnect-link* upon failure of logical link (s, t)

λ_{ij}^{st} : binary variable, $\lambda_{ij}^{st} = 1$ if logical link (s, t) is routed on physical arc $\vec{p}(i, j)$

The objective function of the ILP model is to minimize the total number of wavelengths used on the fibers expressed as:

$$\min \sum_{p(i,j)\in E_p} \sum_{k,(s,t)\in T_k} (\lambda_{ij}^{st} + \sum_{(s',t')\in \overline{T_k}} \gamma_{s,t}^{s',t'} \lambda_{ij}^{s't'})$$

The constraints of the model include two types: Ethernet layer constraints and Optical layer constraints. Ethernet layer constraints ensure that logical *reconnect-link*s are configured such that all the logical link failure scenarios can be recovered and no loop will be caused by the spanning tree reconnection.

$$\sum_{(s't')\in R_{st}} \gamma_{st}^{s't'} \geq 1 \qquad \forall (s, t) \in T^k$$

The above constraints ensure that any logical link on the spanning tree should have at least one logical *reconnect-link*. It can have multiple *reconnect-link*s, since there is a possibility that the logical link on the spanning tree fails simultaneously with one of its logical *reconnect-link*s. In this case, the end nodes of the failed *reconnect-link* should notify another *reconnect-link* to reconnect the spanning tree.

Ethernet layer is constraint to prevent loops after the spanning tree reconnection upon multiple link failures as shown below. Define $\mathfrak{F}_{st}^{ij} = 1$ if link (s,t) is on logical routing of node i and node j. The first two constraints ensure that there is a logical routing between any two end nodes of a fiber, node i and node j, after the failure of the fiber $p(i,j)$. The next two constraints ensure that the logical routing between node i and node j includes *reconnect-link*s and the number of *reconnect-link*s equals to the number of logical links on the protected fiber $p(i,j)$.

$$\lambda_{ij}^{st}(\sum_{(s',t')} \mathfrak{F}_{s't'}^{ij} - \sum_{(t',s')} \mathfrak{F}_{t's'}^{ij}) = \begin{cases} 1, & s' = i; \\ -1, & t' = j; \\ 0, & otherwise \end{cases}$$

$$\forall \vec{p}(i,j) \in E_p,\ \forall (s,t) \in T_k$$

$$\mathfrak{F}_{s't'}^{ij} \leq 1 - \lambda_{ij}^{s't'} \qquad \forall \vec{p}(i,j) \in E_p,\ \forall (s',t') \in T_k$$

$$\mathfrak{F}_{s't'}^{ij} \geq \gamma_{st}^{s't'} \quad \forall (s,t) \in T_k$$

$$\sum_{(s,t)\in T_k} \sum_{(s',t')} \lambda_{ij}^{st} \gamma_{st}^{s't'} = \sum_{(s,t)\in T_k} \lambda_{ij}^{st} \quad \forall \vec{p}(i,j) \in E_p$$

The constraints ensure that if multiple logical links on the spanning tree are routed over the same physical fiber, the *reconnect-link*s of these links form a spanning tree to connect all the separated sub trees.

Optical layer constraints include routing constraints and survivable routing

constraints. Survivable routing constraints ensure that any single fiber cut does not result in disconnection of the spanning tree. In other words, the condition that lightpath routing of the logical link on the spanning tree and those of its logical *reconnect-link*s do not traverse the same fiber must be satisfied.

$$1 + \sum_{(s',t') \in R_{st}} \gamma_{st}^{s't'} \geq \lambda_{ij}^{st} + \sum_{(s',t') \in R_{st}} \lambda_{ij}^{s't'} \gamma_{st}^{s't'}$$

$$\forall (s,t) \in T_k, \forall \vec{p}(i,j) \in E_p$$

Finally, there are also routing constraints at the optical layer for logical links on the spanning trees and *reconnect-link*s shown as below:

$$\sum_{j;\vec{p}(i,j) \in E_p} \lambda_{ij}^{st} - \sum_{j;\vec{p}(j,i) \in E_p} \lambda_{ji}^{st}$$

$$= \begin{cases} 1, & s = j;\ (s,t) \in T_k \ or \sum\limits_{(s',t') \in T_k} \gamma_{s't'}^{st} \geq 1 \\ -1, & t = i;\ (s,t) \in T_k \ or \sum\limits_{(s',t') \in T_k} \gamma_{s't'}^{st} \geq 1 \\ 0, & otherwise \end{cases}$$

6.5 Performance Evaluation

In this section, we study the effectiveness and network resource utilization of the FVSTR mechanism applied to Ethernet over WDM networks using an ILP solver.

We randomly generate various topologies with different sizes and density as the physical topologies. Two randomly generated spanning trees are placed as the logical spanning trees over the physical topology. We randomly select 5 different physical topologies with each network size and density, and also randomly select 5 groups of spanning trees. The experiments are then run with these network configurations, and the average values are reported.

We compare the FVSTR mechanism with two optical layer survivability schemes: dedicated backup path (DBP) protection and shared backup path (SBP) protection. Dedicated backup path protection builds two link-disjoint lightpaths for each logical link on the spanning tree. Upon failure of a primary lightpath, the backup lightpath is used as the logical link for traffic transmission. The dedicated backup path protection requires dedicated wavelength for each lightpath, while a wavelength of backup lightpaths can be shared in shared backup path protection.

Table 6.1 shows the total number of wavelengths used on the links for 9-node and 12-node physical topologies with different degrees, including the wavelengths used for the logical links on the spanning tree and logical *reconnect-link*s.

It can be observed from the table that Fast Virtual Spanning Tree Reconnection (FVSTR) mechanism can save around 15% of wavelengths used in the network than shared backup path (SBP) protection. Even though SBP allows wavelength sharing among backup lightpaths, there is more capacity sharing within FVSTR

Table 6.1: Total Number of Wavelengths

	DBP	SBP	FVSTR
9-node deg=2	62	44	38
9-node deg=2.5	44	40	33
9-node deg=3	35	29	25
12-node deg=2	122	95	74
12-node deg=2.5	75	66	52
12-node deg=3	61	53	45

mechanism at the Ethernet layer. It is because FVSTR mechanism allows multiple logical links on the spanning tree share the same *reconnect-link* as long as these logical links will not fail simultaneously. It can also be seen that when physical network density increases, the performance gap between FVSTR mechanism and SBP decreases. This is because, in denser networks , SBP can have more routing choices for each lightpath at the optical layer. In all the cases, DBP requires the most number of lightpahts due to the fact that this mechanism does not support sharing of backup wavelengths.

Table 6.2: Maximum Number of Wavlengths Used on a Link

	DBP	SBP	FVSTR
9-node deg=2	7	5	5
9-node deg=2.5	6	4	3
9-node deg=3	4	3	3
12-node deg=2	10	8	7
12-node deg=2.5	7	4	3
12-node deg=3	5	3	2

Table 6.2 shows the maximum number of wavelengths used on a link in the

network. With the FVSTR mechanism, the maximum number of wavelengths used on one link is minimum which indicates that the network load is more balanced with the FVSTR than DBP and SBP mechanisms.

6.6 Conclusions

In this chapter, we developed a survivability mechanism for Ethernet over WDM networks. At the Ethernet layer, Fast Virtual Spanning Tree Reconnection protocol is used to reconnect the logical spanning tree upon any single or multiple logical link failures on the spanning tree caused by a single physical fiber cut. At the optical layer, lightpath routing is coordinated with the FVSTR configuration at the Ethernet layer to ensure that any single fiber cut on the physical topology can be resolved. We developed an integer linear programming model to solve the survivability problem of lightpath routing jointly with FVSTR configuration in Ethernet over WDM networks. The numerical results show that FVSTR mechanism leads to better network resource utilization and performs more efficiently than optical layer protection mechanisms.

Chapter 7

Conclusions and Further Research

7.1 Conclusions

Metro Ethernet technology is becoming a promising candidate for the future metropolitan area networks. In this thesis, we have developed survivability schemes which can meet the requirements of Metro Ethernet networks.

We proposed a distributed local restoration mechanism for Metro Ethernet networks. Upon failure of a link, a switch restores traffic to pre-configured backup spanning trees locally. We presented the implementation details of the mechanism and formulated the working spanning tree assignment and backup spanning tree configuration problem as an ILP problem based on different backup tree selection strategies (connection-based and destination-based). The problem was proved to be NP-complete. We developed heuristic algorithms for each backup tree selection

strategy. It is observed that both connection-based and destination-based local restoration mechanisms have fast recovery speed and efficient bandwidth utilization. The efficient bandwidth usage of the mechanism should be attributed to backup capacity sharing in the mechanisms. Simulation results show that the gap between the results from our heuristic algorithms and the optimal results is less than 10%. It indicates that our heuristic algorithms perform closely to the optimal one, which should be attributed to the way of link cost computation of the heuristic algorithms. The simulation results on large scale and more realistic network shows that a small number of spanning trees is sufficient to achieve good performance. Therefore the local restoration mechanism can achieve the goal of efficient capacity utilization and scalability.

In order to satisfy the survivability requirements of Metro Ethernet with low cost Ethernet switches, we also proposed a fast spanning tree reconnection (FSTR) mechanism for Metro Ethernet networks. In the mechanism, a *reconnect-link* is activated to reconnect the two separated subtrees to a whole spanning tree in case of a single link failure. The mechanism can ensure fast failure recovery due to the fact that only Ethernet switches on the reconnect-path need to be notified and re-configured avoiding the switching table flushing and backward learning is avoided. FSTR pre-configuration problem was formulated as an ILP problem, and proved to be NP-Complete. The numerical results show that with lower implementation cost, FSTR can have similar performance as optimal RSTP on a single

spanning tree and connection-based local restoration mechanism on multiple spanning trees. The results suggest that with only single link reconnection, the FSTR mechanism can have close performance to the global optimal solution. We also developed an efficient augmentation based algorithm that obtain close approximation to the optimal solution. This algorithm was compared with the optimal model. The simulation results closely approximate the optimal results. The significance of the FSTR mechanism, in addition to its fast recovery and efficiency, is its compatibility with off-the-shelf Ethernet switch with little modification and ease of implementation.

We presented a resilient mechanism to handle double-link failures in Metro Ethernet networks. The mechanism is based on Fast Spanning Tree Reconnection mechanism, but solves the loop problem in the original FSTR mechanism upon double-link failures. We proved that with some additional constraints during pre-configuration, loops can be avoided when using the FSTR mechanism to handle double-link failures. In addition, we developed a reconnect-link pre-configuration model to provide a specified protection grade for the connections in the network. The numerical results show that our model has the capability to pre-configure the reconnect-links such that the protection requirement for each connection can be satisfied with efficient utilization of the backup capacity.

To implement the survivability schemes in Metro Ethernet built over the optical network infrastructures, we finally presented a two layer survivability architecture

of Ethernet over WDM networks. The survivability mechanisms at the optical layer and Ethernet layer coordinate with each other to provide efficient and guaranteed protection upon single fiber cut. The numerical results show that our proposal of two layer integrated survivability mechanism on Ethernet over WDM network is more resource efficient than the traditional optical layer protection mechanisms.

We have compared our local restoration and fast spanning tree reconnection mechanism with each other and mechanisms proposed by other researchers. Local restoration mechanism provides lower resource redundancy than other mechanisms (full redundancy) with multiple spanning trees. FSTR has a similar network redundancy to $RSTP_{optimal}$ on single spanning tree (5% higher) and lower network redundancy (10%) than link-based local restoration on multiple spanning trees. Meanwhile, FSTR is simpler and its recovery speed is faster than traditional RSTP.

7.2 Contributions of the Thesis

7.2.1 Local Restoration in Metro Ethernet

- A local restoration mechanism using multiple spanning trees for Metro Ethernet networks has been proposed. It handles failure in a fast and distributed way providing guaranteed 100% protection.

- An NP-complete proof of local restoration configuration problem has been given. Two heuristic algorithms (connection-based and destination-based) have been developed.

7.2.2 Fast Spanning Tree Reconnection Mechanism

- We proposed a survivability scheme called Fast Spanning Tree Reconnection (FSTR) mechanism which uses a single link to reconnect the spanning tree upon a single link failure in Metro Ethernet networks.

- We proved that the FSTR pre-configuration problem is NP-complete and developed an efficient augmentation based algorithm.

- We studied the deadlock problem of the FSTR mechanism upon double-link and multiple-link failures, and developed a theorem to provide a loop free FSTR configuration.

7.2.3 Survivability in Ethernet over WDM Optical Networks

- Based on the traditional two layer peer model, we developed an integrated survivability mechanism for Ethernet over WDM optical networks which provide guaranteed protection upon any single fiber cut through coordination of

the two layers.

7.3 Future Research

It is acknowledged that the traffic pattern used in this study is assumed to be static. In MANs and WANs, the assumption is reasonable since traffic in these networks would not change drastically in a short time. However, the survivability mechanism for Metro Ethernet Networks under dynamic traffic pattern should also be considered. Spanning tree reconfiguration should be made periodically to adapt to the traffic changes in the network. With dynamic traffic, backup capacity can be over-provisioned for guaranteed protection. Another possible solution is to reserve backup capacity based on the statistical characteristic of the traffic, and provide a certain protection grade instead of 100% protection.

It should be noted that the survivability schemes for Metro Ethernet networks proposed by us require a centralized manager to pre-configure the spanning tree structures, map traffic to each spanning tree. Since the traditional Ethernet has the features of ease of configuration and plug-and-play, a distributed pre-configuration of the survivable schemes in which a distributed protocol can be used replacing the centralized manager is another future research direction. Consider the FSTR mechanism, it is possible to let each Ethernet switch have the capability to determine its own *reconnect-link* independently. Therefore, a signaling protocol will be

needed for the topology and traffic information exchange among Ethernet switches, and a decision algorithm can be used for each Ethernet switch to determine the *reconnect-link*.

Currently, the survivable schemes and protocols for Metro Ethernet networks proposed by us are based on theoretical analysis and simulation. Since our mechanisms are compatible with current Ethernet switches, it is possible to design a testbed of the Metro Ethernet survivability system, similar to the Viking system, and study its performance in a realistic network. Some improvements should be done on Ethernet switch to support our mechanisms. With local restoration, Ethernet switch should be implemented to have the capability of changing Ethernet frame's VLAN ID. With FSTR, two tables should be configured in Ethernet switch, and a module should be added to make Ethernet switch configure its switching table based on the tables. . These algorithms can be run on commodity hardware. We expect that the FSTR protocol should have faster recovery speed and better bandwidth utilization than the widely-used RSTP even in a realistic environment.

7.4 Publications

1. J. Qiu, G. Mohan, K. C. Chua, and Y. Liu, "Local Restoration with Multiple Spanning Trees in Metro Ethernet," *in Proceedings of ONDM'08*, 2008.

2. J. Qiu, Y. Liu, G. Mohan and K. C. Chua, "Fast Spanning Tree Reconnection

in Resilient Metro Ethernet Networks," *in Proceedings of ICC'09*, 2009.

3. J. Qiu, G. Mohan and K. C. Chua, Y. Liu, "Handling Double-Link Failures in Metro Ethernet Networks Using Fast Spanning Tree Reconnection," *in Proceedings of Globecom'09*, 2009.

4. J. Qiu, G. Mohan, K. C. Chua, and Y. Liu, "Local Restoration with Multiple Spanning Trees in Metro Ethernet Networks," accepted by *ACM/IEEE Transaction on Networking.*

5. J. Qiu, Y. Liu, G. Mohan and K. C. Chua, "Fast Spanning Tree Reconnection Mechanism in Resilient Metro Ethernet Networks," accepted by *Computer Networks.*

Bibliography

[1] *Strategic Directions Moving the Decimal Point: An Introduction to 10 Gigabit Ethernet.* Cisco System, 2002.

[2] M. Ali, G. Chiruvolu, and A. Ge, "Traffic engineering in metro ethernet," *IEEE Network*, vol. 2, pp. 11–17, 2004.

[3] M. Golash, "Reliability in ethernet networks: Survey of various approaches," *Bell Labs Technical Journal*, vol. 11, no. 3, pp. 161–171, 2006.

[4] A. Myers, T. S. E. Ng, and H. Zhang, "Rethinking the service model: Scaling ethernet to a million nodes," in *3th Workshop on Hot Topics in Networks*, 2004.

[5] S. Halabi, *Metro Ethernet.* Cisco System, 2003.

[6] A. S. Tanenbaum, *Computer Networks.* Pearson International Edition, 2003.

[7] D. E. McDysan and D. L. Spohn, *ATM Theory and Applications.* McGraw-Hill, 1999.

[8] IEEE, *IEEE 802.1d, Standard for Local and Metropolitan Area Networks-Media Access Control(MAC) Bridges.*

[9] IEEE, *IEEE 802.1w, Standard for Local and Metropolitan Area Networks-Rapid Reconfiguration of Spanning Tree.*

[10] IEEE, *IEEE 802.1s, Standard for Local and Metropolitan Area Networks-Multiple Spanning Trees.*

[11] R. C. Sofia, "A survey of advanced ethernet forwarding approaches," *IEEE Communications Surveys and Tutorials*, vol. 11, no. 1, pp. 92–114, 2009.

[12] K. Elmeleegy, A. L. Cox, and T. S. E. Ng, "On count-to-infinity induced forwarding loops in ethernet networks," in *Proceedings of IEEE INFOCOM'06*, 2006.

[13] IEEE, *IEEE 802.1q, Standard for Local and Metropolitan Area Networks-Virtual Bridged Local Area Networks.*

[14] IEEE, *IEEE 802.1ad, Standard for Local and Metropolitan Area Networks-Provider Bridge.*

[15] R. Santitoro, "Metro ethernet services-a technical overview," *Merto Ethernet Forum, http://www.metroethernetforum.org*, 2008.

[16] L. Rodeheffer, A. Thekkath, and C. Anderson, "Smartbridge: A scalable bridge architecture," in *Proceedings of SIGCOMM'00*, pp. 205–216, 2000.

[17] K. Lui, W. C. Lee, and K. Nahrstedt, "Star: A tranparent spanning tree bridge protocol with alternate routing," *ACM SIGCOMM Computer Communicatoin Review*, vol. 32, no. 3, pp. 33–46, 2002.

[18] F. D. Pellegrini, D. Starobinski, G. Karpovsky, and B. Levitin, "Scalable cycle-breaking algorithms for gigabit ethernet backbones," in *Proceedings of IEEE INFOCOM'04*, pp. 2175–2184, 2004.

[19] M. Dannhardt, *Ethernet Over SONET*. Technology White Paper, Sierra Inc., 2004.

[20] A. Srivastava, S. Acharya, M. Alicherry, B. Gupta, and P. Risbood, "Differential delay aware routing for ethernet over sonet/sdh," in *Proceedings of IEEE INFOCOM'05*, pp. 1117–1127, 2005.

[21] P. Risbood, S. Acharya, and B. Gupta, "The best challenge for next-generation ethernet services," in *Proceedings of IEEE INFOCOM'05*, pp. 1049–1059, 2005.

[22] *MPLS: Making the Most of Ethernet in the Metro.* Tehcnology White Paper, River Stone Networks, 2006.

[23] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C. Chen-Nee, Y. Ganjali, , and C. Diot, "Characterization of failures in an operational ip backbone network," *IEEE/ACM Transactions on Networking*, vol. 16, no. 4, pp. 749–762, 2008.

[24] M. To and P. Neusy, "Unavailability analysis of long-haul networks," *IEEE Journal on Selected Areas in Communications*, vol. 12, no. 1, pp. 100–109, 1994.

[25] ATIS, *T1.Rpt24-1993 A Technical Report on Network Survivability Performance.* 2003.

[26] W. D. Grover, *Mesh-Based Survivable Networks: Options and Strategies for Optical, MPLS, SONE, and ATM Networking.* Pretince Hall, 2004.

[27] R. Murthy and G. Mohan, *WDM Optical Networks: Concept, Design and Algorithms.* Pretince Hall PTR, 2002.

[28] A. K. Somani, *Survivability and Traffic Grooming in WDM Optical Networks.* Cambridge University Press, 2005.

[29] G. Mohan and R. Murthy, "Lightpath restoration in wdm optical networks," *IEEE Network*, vol. 14, no. 6, pp. 25–32, 2000.

[30] D. Y. Xiong and C. Qiao, "Acheiving fast and bandwidth efficient shared-path protection," *Journal of Lgithwave Technology*, vol. 21, no. 2, pp. 25–32, 2003.

[31] P. H. Ho, J. Tapolcai, and T. Cinkler, "Segment shared protection in mesh communication networks with bandwidth guaranteed tunnels," *IEEE/ACM Transactions on Networking*, vol. 12, no. 6, pp. 1105–1118, 2004.

[32] R. S. Bahatia, M. Kodialam, T. V. Lakshman, and S. Sengupta, "Bandwidth guaranteed routing with fast restoration against link and node failures," *IEEE/ACM Transactions on Networking*, vol. 16, no. 6, pp. 1321–1330, 2008.

[33] L. Li, M. M. Buddhikot, C. Chekuri, and K. Guo, "Routing bandwidth guaranteed paths with local restoration in label switched networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 2, pp. 437–449, 2005.

[34] R. Cohan and G. Nakibly, "Maximizing restorable throughput in mpls networks," *IEEE/ACM Transactions on Networking*, 2009.

[35] A. Kvalbein, A. F. Hansen, T.Cicic, S. Gjessing, and O. Lysne, "Fast ip network recovery using multiple routing configuration," in *Proceedings of IEEE INFOCOM'04*, 2006.

[36] O. Bonaventure, C. Filsfils, , and P. Francoisr, "Achieving sub-50 milliseconds recovery upon bgp peering link failures," *IEEE/ACM Transactions on Networking*, vol. 15, no. 5, pp. 1123–1135, 2007.

[37] M. M. Medard, S. G. Finn, , R. A. Barry, and R. G. Gallager, "Redundant trees for preplanned recovery in arbitrary vertex-redundant or edge-redundant graphs," *IEEE/ACM Transactions on Networking*, vol. 7, no. 5, pp. 641–652, 1999.

[38] *Ethernet Automatic Protection Switching (EAPS).* Tehcnology White Paper, Extreme Networks, 2006.

[39] K. K. Lee, J. Ryoo, and S. Min, "An ethernet ring protection method to minimize transient traffic by selective fdb advertisement," *ETRI Journal*, vol. 3, no. 5, pp. 631–633, 2009.

[40] J. Ryoo, H. Long, Y. Yang, M. Holness, Z. Ahmad, and J. Rhee, "Ethernet ring protection for carrier ethernet networks," *IEEE Communication Magzine*, vol. 46, no. 9, pp. 136–143, 2008.

[41] M. V. Pardmaraj, V. S. Nair, M. Marchetti, G. Chiruvolu, , and M. Ali, "Bandwidth sensitive fast failure recovery scheme for metro ethernet," *Computer Networks*, vol. 52, pp. 1603–1616, 2008.

[42] D. Jin, Y. Li, W. Chen, L. Su, and L. Zeng, "Ethernet ultra-fast switching: A tree-based local recovery scheme," *IET Communications*, vol. 4, no. 4.

[43] A. Kern, I. Moldocan, and T. Cinkler, "Bandwidth guarantees for resilient ethernet networks through rstp port cost optimization," in *Proceedings of AccessNets'07*, 2007.

[44] S. Sharma, K. Goplan, S. Nanda, , and T. Chiueh, "Viking: A multiple-spanning-tree ethernet architecture for metropolitan area and cluster networks," in *Proceedings of IEEE INFOCOM'04*, 2004.

[45] T. Cinkler, A. Kern, , and A. Moldovan, "Optimized qos protection of ethernet trees," in *Proceedings of DRCN'05*, 2005.

[46] J. Farkas, C. Antal, L. Westberg, A. Paradisi, T. R. Tronco, and V. G. Oliveira, "Fast failure handling in ethernet network," in *Proceedings of IEEE ICC'06*, 2006.

[47] M. Huynh, P. Mohapatra, and S. Goose, "Spanning tree elevation protocol: Enhancing metro ethernet performance and qos," *Computer Communications*, vol. 32, pp. 750–765, 2009.

[48] G. Baier, T. Engel, and A. Autenrieth, "Evaluation of survivable ethernet over wdm network architectures in metro ring networks," in *Proceedings of DRCN'07*, 2007.

[49] A. Kern, I. Moldovan, P. Hegyi, and T. Cinkler, "Ethernet over wdm: Optimization for resilience and scalabilitys," in *Proceedings of DRCN'07*, 2007.

[50] A. Hadjiantonis, "Evolution to a converged layer 1, 2 in a global-scale, native ethernet over wdm-based optical networking architecture," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 5, pp. 1048–1058, 2007.

[51] M. Batayneh, D. A. Schupke, M. Hoffmann, A. Kirstaedter, and B. Mukerhjee, "Lightpath-level protection versus connection-level protection for carrier-grade ethernet in a mixed-line-rate telecom network," in *Proceedings of IEEE GLOBECOM'07*, 2007.

[52] J. Farkas, C. Antal, G. Toth, and L. Westberga, "Distributed resilient architecture for ethernet networks," in *Proceedings of ACM 33rd annual on Southeast regional conference*, 2005.

[53] K. Li, "Probabilistic analysis of an approximation algorithm for maximum subset sum using recurrence relations," in *Proceedings of DRCN'05*, pp. 219–226.

[54] Y. Liu, D. Tipper, , and P. Siripongwutikorn, "Approximating optimal spare capacity allocation by successive survivable routing," *IEEE/ACM Transaction on Networking*, vol. 13, no. 1, pp. 198–211, 2005.

[55] S. Ramamurthy and B. Mukherjee, "Survivable wdm mesh networks, part i-protection," in *Proceedings of IEEE INFOCOM'99*, 1999.

[56] S. Sebbah and B. Jaumard, "A resilient transparent optical network design with a pre-configured extended-tree scheme," in *Proceedings of IEEE ICC'09*, 2009.

[57] F. Solano, T. Stidsen, R. Fabregat, and J. L. Marzo, "Label space reduction in mpls networks: How much can a single stacked label do?," *IEEE/ACM Transaction on Networking*, vol. 16, no. 6, pp. 1308–1320, 2008.

[58] B. M. Waxman, "Routing of multipooint connections," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1617–1622, 1988.

[59] G. N. Frederickson and J. Ja'Ja, "Approximation algorithms for several graph augmentation problem," *Siam Journal on Computings*, vol. 10, no. 2, pp. 270–283, 1981.

[60] A. Meddeb, "Multiple spanning tree generation and mapping algorithms for carrier class ethernets," in *Proceedings of IEEE GLOBECOM'06*, 2006.

[61] D. Santos, A. D. Sousa, F. Alvelos, M. Dzida, M. Pioro, , and M. Zagozdzaon, "Traffic engineering of multiple spanning tree routing networks: the load balancing case," in *Proceedings of IEEE NGI'09*, 2009.

[62] K. P. Eswaran and R. E. Tarjan, "Augmentation problems," *Siam Journal on Computings*, vol. 5, no. 4, pp. 653–665, 1976.

[63] H. Ma, D. Fayek, and P. Ho, "Availability-constrained multipath protection in backbone networks with double-link failure," in *Proceedings of IEEE ICC'08*, 2008.