Survivable Multipath Provisioning Schemes in Mesh Transport Networks

by

Guo Li

A Thesis Presented to Lakehead University

In Partial Fulfillment of the Requirement for the Degree of

Master of Science in Electrical and Computer Engineering

Thunder Bay, Ontario, Canada

April 21st, 2011

Lakehead
UNIVERSITY

OFFICE OF GRADUATE STUDIES

NAME OF STUDENT: Guo Li

DEGREE AWARDED: Master of Science in Engineering

ACADEMIC UNIT: Graduate Program in Electrical and Computer Engineering

TITLE OF THESIS: Survivable Multipath Provisioning Schemes in Mesh Transport Networks

This thesis has been prepared

under my supervision

and the candidate has complied

with the Master's regulations.

Signature of Supervisor

5/4/2011

Date

Abstract

Back to the 1960s, Internet originated from the military research of survivable, fault-tolerant and distributed computer networks. Half-century later, quarter of Earth's population use Internet services. International business transactions are promoted by the accelerated information interaction across the world. With the prosperity Internet brought to our society, its survivability and capacity become an important issue.

We investigate the problem of efficient multipath provisioning in backbone mesh networks by employing pool sharing and traffic decentralization techniques such as working and backup traffic split and mix-routing. With Multipath provisioning, a connection request can be split and diversely routed on to several paths. A feature that has two improvements over single-path provisioning: 1) load balancing among working paths, and 2) reducing the bandwidth of backup path. However, these diversely routed paths are subject to the limitation of network topology, which limit the number of available paths that can be found inside the network.

We introduce a new mix-routing algorithm referred as *YinYang* algorithm (named after the well-known Chinese concept in science and philosophy), with which a connection request's working and backup (spare) bandwidth can be divided by the number of available paths between the source and destination nodes. Each path will be composed of an equal portion of the working traffic and the corresponding backup traffic. These decentralized backup paths enhance the possibility of different demands sharing the backup paths which will lead to the

reduction in the backup bandwidth. Using simulation methodology, we show that if YinYang algorithm is augmented with shared restoration policy, spare capacity for recovering single link failure is minimized. We also show that for multi-link failures, YinYang algorithm with shared back up can achieve lower demand blocking ratio than the Dedicated Backup Path Protection algorithm and the Simple Pool Sharing algorithm investigated in the literature.

Acknowledgements

I would like to thank my supervisor Dr. Hassan Naser for his guidance and advising while completing my thesis. Not only because this work is based on his Simple Pool Sharing algorithm, I followed his book chapter to write my simulation model, but also because his encouragement in finding my new idea.

I would like to extend my thanks to Gregory Toombs, my friend. He helped me with my simulation model and guided me to create my own state machine in order to apply my algorithm.

I would also like to extend my thanks to my family and friends. Without all your encouragement and support, this work would not be possible.

Table of Contents

List of Figures

List of Symbols

Symbol	Meaning
C	The required bandwidth of demand r
J	The total number of links in a network
N	The total number of nodes in a network
M_i	The total bandwidth of link i
A_i	The available bandwidth on link i
W_i	The allocated working bandwidth on link i
B_i	The allocated backup bandwidth on link i
T_j	The maximum backup bandwidth reserved on link j if the other link fails
$C_{P1}(i)$	The cost of link I when the 1st path is computed
$C_{P2}(j)$	The cost of link j when the 2nd path is computed
$S_{P1}(r)$	The set of links on the 1st path
$S_{P2}(r)$	The set of links on the 2nd path
n_d	The number of demands accepted
$S_w(r)$	Set of links along the working path of demand r
$S_b(r)$	Set of links along the backup path of demand r

Φ	The matrix of shared backup bandwidth

List of Abbreviations

Abbreviation	Meaning
DPP	Dedicated Path Protection
SPR	Shared Path Restoration
SPS	Simple Pool Sharing
MPP	Multi-Path Provisioning
WPRC	Working Path Resource Consumption
BPRC	Backup Path Resource Consumption

Chapter 1

Introduction

1.1 Motivation

Ancient Chinese people ignited firewood on the beacon on the Great Wall to notify the enemy's invasion. For every beacon on the great wall, at least two beacons were within the sight from the direction from which the message came. Modern communication networks also apply this kind of redundant design to enhance reliability. Mesh networks across the world play an important role in globalization. All of society depends on communication networks to function on a daily basis. Managing network resources to guarantee efficiency and reliability is becoming more and more important to the overall economy.

This work treats the network survivability problem as a problem of managing network resources. In other words, link bandwidth is allocated for demands on the network with the consideration of efficiency and fairness. *YinYang* theory (a well-known ancient Chinese concept in science and philosophy) is applied in this work to optimize the overall network performance instead of emphasizing the efficiency of single demands. By routing a single demand with the consideration of other demands, more demands can be accepted.

1.2 Survivability

The origin of the modern internet is ARPANET. It was intended to ensure that a communication network should be available after some sections of the network fail from nuclear war. Network survivability is the ability of a network to withstand failure events. It

demonstrates the resilience of the network against failure events and is measured in terms of the reliability, restorability and end-to-end availability of the light path [1].

There are link failures and node failures in the network. Link failure may occur because of earthquake, flood or improper construction. Node failure may occur because of equipment malfunctions. This kind of failure can be solved by providing redundant hardware. The restoration method in this thesis focuses on link failure. The proposed algorithm and benchmark algorithm restore all single link failures. In order to test the capability of these algorithms this thesis simulates multi-link failure.

1.3 The Physical Requirement for Survivable Network

The fundamental requirement for a survivable network is that an alternative connection can be found when a link fails in the network. Tree-like networks (Figure 1-1) can't meet the fundamental requirement. Two bidirectional network connections are needed for every node in the network. The ring topology used in SONET networks is a typical survivable network which meets this requirement (Figure 1-2). Ring network topology is a closed path, which consists of consecutive nodes connected by point-to-point links [2-4]. Every node on the ring is connected to its left-side and right-side nodes.

Ring topologies are widely used in backbone networks. There are two paths between each pair of nodes in a ring topology. One path is used as the working ring and the other as backup path. Thus, the ring topology is survivable in single link failure scenarios.

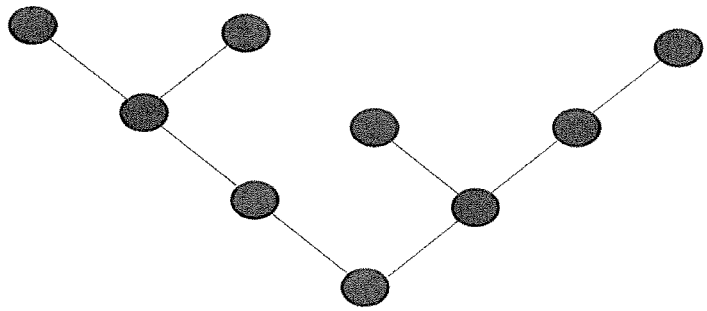


Figure 1-1 Network Types: Tree

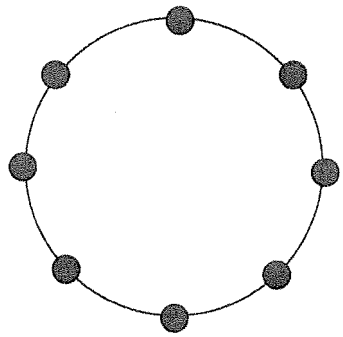


Figure 1-2 Network Types: Ring

1.4 Mesh Network

In a mesh network, every node is connected to at least two other nodes. Compared with a ring topology, a mesh topology provides better flexibility for adjustment and is more robust against link failure.

In a "full mesh" network, there is a direct link between every pair of nodes in the network. All the nodes are connected to the other nodes in a full mesh network (Figure 1-3). In a partial mesh network (Figure 1.4), not all the nodes are connected to the other nodes. The

networks present in this thesis are partial mesh networks, in which every node is connected to two or more nodes, but not to all.

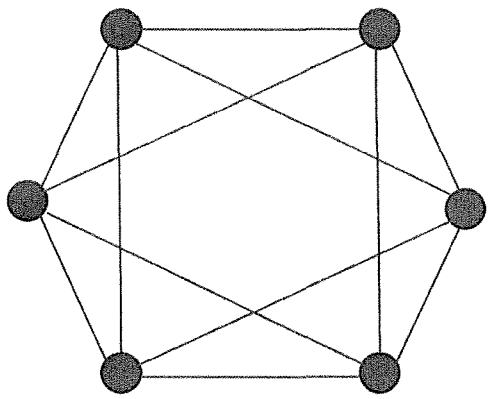


Figure 1-3 Partial Mesh

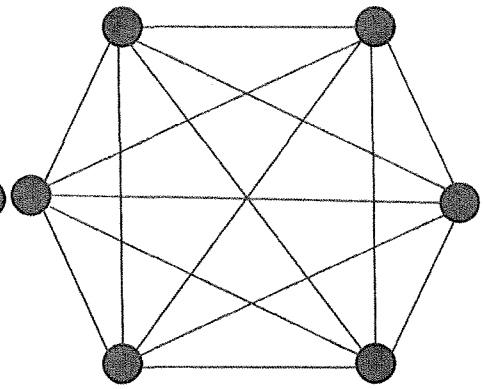


Figure 1-4 Full Mesh

1.5 Survivability in Mesh Networks

Figure 1-5 shows many different schemes that can provide survivability to a mesh network. There are two major types of schemes: protection and restoration. Protection is a proactive scheme in which backup path is calculated in advance, and bandwidth is allocated. Restoration is a reactive scheme. There are total reactive and semi-reactive restoration schemes. In a total reactive scheme, the backup path is calculated only after the working link fails. While in semi-reactive restoration schemes, backup paths are calculated at the same time as calculating the working path, but bandwidth will only be allocated to the backup paths at the time the link fails.

This thesis focuses on a semi-reactive restoration scheme. The reason for this is that proactive protection is inefficient because the backup bandwidth is dedicated to the demand,

and is only used when link failure occurs. A total reactive scheme cannot always restore single link failure, because it does not guarantee to find a backup path for the demand failure.

In path and link protection schemes the backup paths are pre-computed and bandwidth is pre-allocated at the time of demand arrival. Path and link restoration schemes are semi-reactive schemes, where the backup path is calculated at the time of demand arrival, but bandwidth is allocated at the time of link failure.

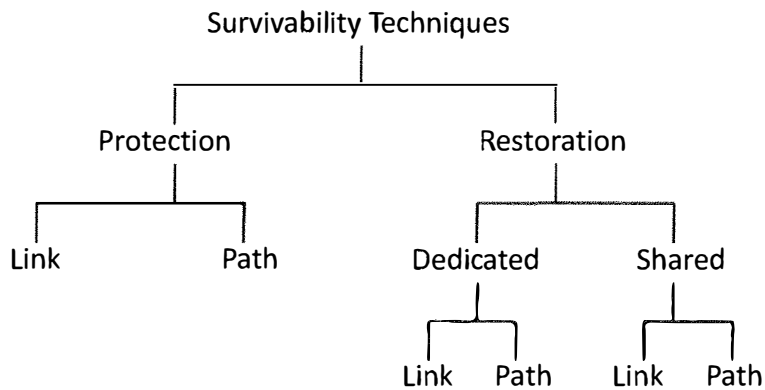


Figure 1-5 Survivability Techniques

1.6 Link and Path Protection and Restoration in Survivable Network

Protection and semi-proactive restoration paradigms can be further divided into Link Protection / Path Protection and Link Restoration / Path Restoration. In link protection and restoration, backup traffic is rerouted by the end nodes of the failed link (Figure 1-6), while in path protection and restoration (Figure 1-7), the backup traffic is rerouted by the source and destination nodes of each failed connection. Link protection and restoration are also considered "local" whereas path protection and restoration are considered "end-to-end".

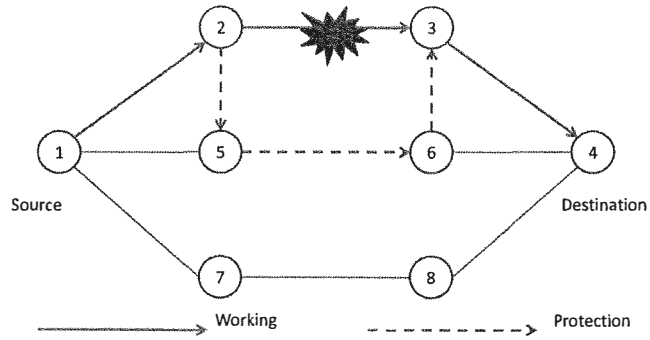


Figure 1-6 Link Protection/Restoration

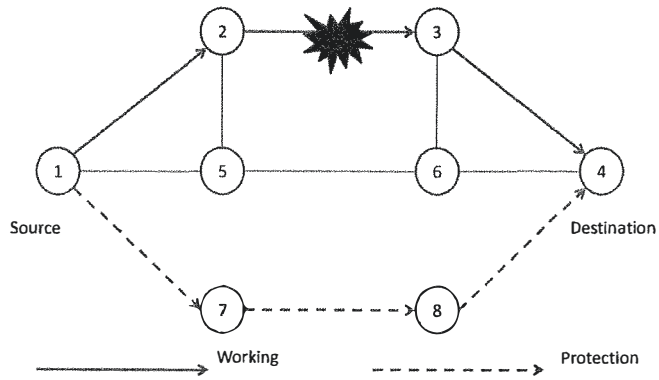


Figure 1-7 Path Protection/Restoration

1.7 Dedicated Path Protection versus Shared Path Restoration

There are two schemes for backup bandwidth allocation: dedicated path protection and shared path restoration. In both of these two schemes a pair of link-disjoint working and backup paths is found between the source and destination nodes. The difference is the allocation of backup bandwidth.

In the dedicated path protection scheme, the link capacities on the backup path are reserved exclusively for every demand. The example in Figure 1-8 shows the working path and backup path (dashed arcs) of two dedicated path protection demands A-C and A-D. It is assumed that both of them require one unit of bandwidth. Demand A-C's backup paths are links AB and BC, whereas demand A-D's backup path travels links AB, BC and CD. One unit of bandwidth for both demands A-C and A-D is exclusively reserved for each demand on the backup links AB and BC, so there are two units of bandwidth reserved on link AB and link BC.

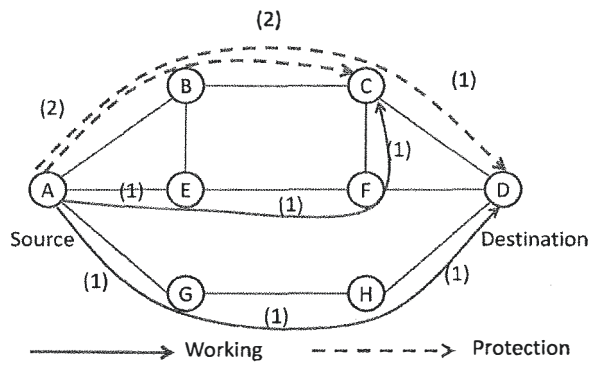


Figure 1-8 Dedicated Path Protection

In the shared path restoration scheme, the link capacities on the backup path are shared between different demands as long as their working paths are failure-disjoint. The example in Figure 1-9 shows the working path and backup path (dashed arcs) of two shared path restoration demands A-C and A-D. Demand A-C's backup path travels link AB and BC, whereas demand A-D's backup path travels links AB, BC and CD. One unit of bandwidth for both demands A-C and A-D is shared for these two demands, so there is one unit of

bandwidth reserved on link AB and link BC. If one link among these two working paths fails, the bandwidth reserved on the link AB and link BC will restore the demand. If both of these working paths fail, shared bandwidth reserved can only restore the first failed demand.

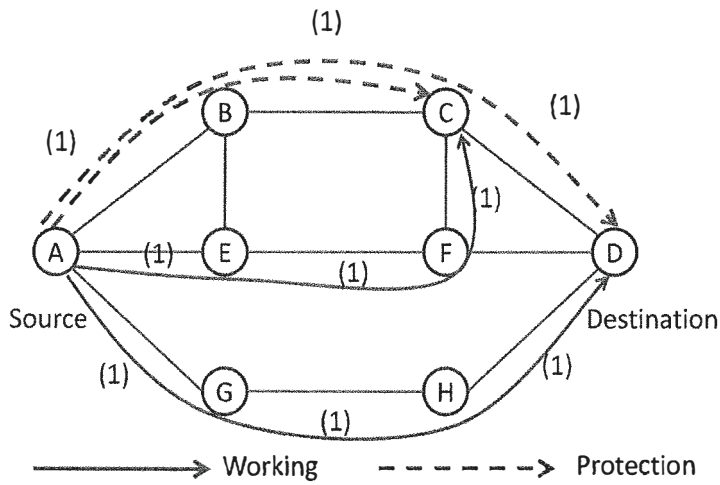


Figure 1-9 Shared Path Restoration

Shared restoration is a kind of mesh restoration scheme that shares the backup capacity between different demands when the working paths of these demands are failure disjoint. So this scheme can restore demands from a single link failure in the network. Previous research studies indicated that shared restoration is the most promising (a very efficient) scheme for saving the spare capacity while fully restoring any single link failure [5]. In this thesis, shared restoration scheme is used as the benchmark algorithm.

1.8 Load Balancing

Ensuring an even workload helps to eliminate congestion in network links [6]. Achieving traffic balance is important in communication networks. The objective of this thesis is to improve load balance and eliminate link congestions. Load balancing in this thesis refers to distributing the network traffic evenly among network links so links will not so easily be overloaded and become a bottleneck.

Figure 1-10 and Figure 1-11 show the advantage of taking load balancing into consideration during routing. Figure 1-10 shows the result of a routing scheme without load balancing. Each link has 3 units of bandwidth. Three demands A-E, A-C and A-D have been routed in this network. When applying an algorithm without a load-balancing mechanism, the link on the shortest path link A-E can easily become overloaded. Link A-E allocated all 3 units of bandwidth and can't accommodate a new demand.

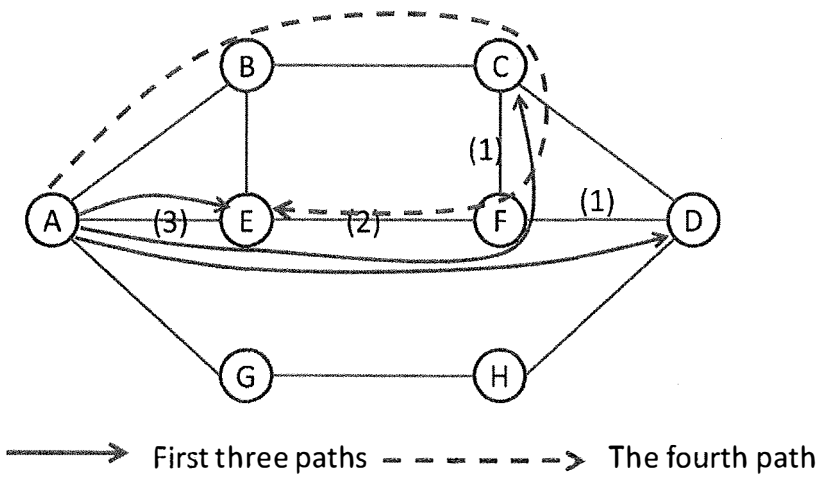


Figure 1-10 Routing Scheme without Load Balancing

Figure 1-11 shows the result of a routing scheme with load balancing. When applying an algorithm with a load-balancing mechanism (in Figure 1-11), paths of the demand are evenly distributed among the network. On working paths of three demands A-E, A-C and A-D, there is no link that is overloaded. When a new demand between A-E arrived on the network, this demand needs 4 units of bandwidth in Figure 1-10, in order to bypass the overloaded link, while needing only one unit in Figure 1-11.

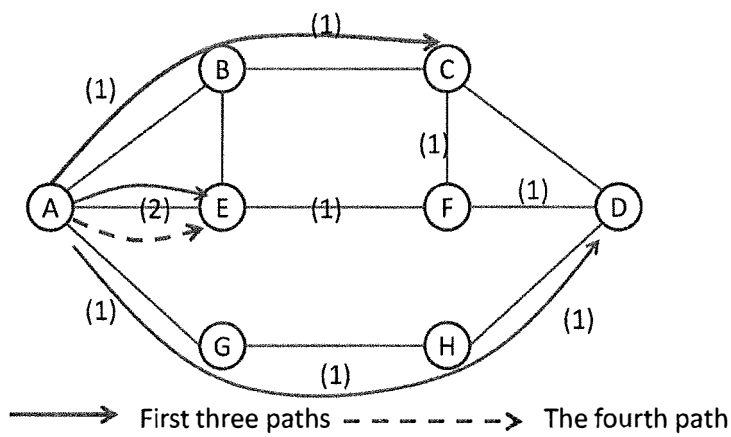


Figure 1-11 Routing Scheme with Load Balancing

1.9 Contribution of this thesis

This thesis proposes the YinYang algorithm which is the first algorithm that breaks the boundary between working paths and backup paths to route them in the same path. The YinYang algorithm uses the same mechanism as shared path restoration to enhance the efficiency by reusing shared capacity. It splits the working traffic between given source and destination nodes into multiple paths which yields a more even bandwidth distribution

amongst network links than the benchmark algorithms. This advantage leads to much lower congestion.

This thesis also introduces a simulation model that evaluates the performance of the YinYang algorithm. In this simulation model, we compare the YinYang algorithm with 1+1 dedicated protection and simple pool sharing. The YinYang algorithm has the lowest demand blocking ratio.

1.10 Outline

Chapter 1 introduces the background knowledge for an understanding of network survivability. Chapter 2 introduces the related research for protection and restoration algorithms. Presented benchmark algorithms are the dedicated path protection and the simple pool sharing.

Chapter 3 introduces the YinYang algorithm. It starts by introducing the concept of YinYang, and then shows the appearance of the two-path YinYang algorithm. It also shows the routing method. It ends by developing the YinYang algorithm into multi-path cases.

Chapter 4 introduces the metrics for evaluating the performance of YinYang algorithm. Then it explains working principles of the simulation model. The results are shown and the explanation for the results is also given.

Chapter 5 lists the contributions for the YinYang algorithm. Potential further improvements are also mentioned in this chapter.

Chapter 2

Related Research

2.1 Existing Algorithms for Protection and Restoration

There are many algorithms for network protection and restoration. Some of the related algorithms will be presented in this section.

2.1.1 Dedicated Path Protection (DPP)

With one working path and one link-disjoint backup path, 1+1 dedicated path protection algorithm achieves high network availability[5]. In [7], by increasing the number of link-disjoint backup paths from one to N ($N \geq 2$), 1+N dedicated protection schemes can achieve extremely high availability.

In [8], if the link is highly reliable, the working and backup paths no longer need to be link-disjoint. A demand can share these highly reliable links among both of its working and backup paths. Better capacity utilization can be achieved.

Dedicated path protection is used as a benchmark algorithm in this thesis, and it is used to compare with other algorithms, because of its high network availability.

2.1.2 Shared Path Restoration (SPR)

Shared Path Restoration (SPR) assumes that not all working paths will fail at the same time. The links on the backup path can be shared between different demands. Shared path restoration achieves high efficiency in reusing the backup bandwidth resource.

A mathematical formula for calculating availability of shared path restoration was proposed in [9]. It indicates that the availability of shared path restoration is lower than that of dedicated path protection.

In order to improve the availability of shared path restoration, the availability of shared path restoration in a multiple link failure scenario is addressed in [10]. The major reason for poor performance of shared path restoration is that the backup resource is shared among different demands. In the case that multiple demands are affected by working path link failures, a multitude of these demands may rush to the same backup resource, and the backup resource is not designed to serve all of these failures.

A heuristic routing algorithm is proposed in [10] to improve the availability of shared path restoration. This algorithm measures the likelihood that a backup link needs to restore more than one demand. It adjusts the backup bandwidth on that link in order to provide adequate bandwidth for restoring multiple demand failure.

Quality of Service (QoS) in shared restoration is discussed in [11]. It focuses on end-to-end delay time along the primary and backup paths. Shared path restoration algorithms that improve delay performance were proposed. In [12] traffic load balancing is emphasized. It

ensures even workload distribution, and helps to eliminate congestion on network links. A shared path restoration algorithm with an improved load balancing routing method was proposed.

In [13] a One-step Delay-constrained Pool Sharing (ODPS) algorithm was proposed to minimize the delay between the source and destination and eliminate the trap topology. Trap topology is a network topology where the algorithm cannot find a pair of link-disjoint paths for the connection request even though diverse paths for the connection request actually exist on the topology. This One-step Delay-constrained Pool Sharing (ODPS) scheme eliminates the trap topology problem by finding the link that causes a trap. Then the two link-disjoint paths will be found in the rest of the links.

Shared path restoration is highly efficient in reusing network resources. The Simple pool sharing is a kind of shared path restoration scheme that minimizes the amount of bandwidth on the shared backup path. It will be described in detail in section 2.3.2. It is used as a benchmark algorithm in this thesis, because it's highly efficient in reusing backup capacity.

2.1.3 Multi Path Provisioning (MPP)

Multi Path Provisioning (MPP) finds N (the number of paths) shortest paths between source and destination nodes of a given demand. $N-1$ paths are used as working paths, and one path is used as the backup path. Each working path takes $\frac{1}{N-1}$ of the working bandwidth needed by the demand. It is assumed that only one link fails at a time. The backup path can

handle one path failure. The bandwidth of the backup path equals the bandwidth of one of the working paths, i.e. it is equal to $\frac{1}{N-1}$ times the bandwidth requested by the demand.

In [7] a new notation $M : N$ for multi-path provisioning was proposed. M is the number of working paths and N is the number of backup paths. M shortest link-disjoint working paths are found one-by-one and then the N shortest link-disjoint backup paths are found. The method of exactly calculating availability was proposed. A link capacity adjustment scheme (LCAS), which migrates bandwidth between different paths to eliminate the congestion on overloaded links, is also proposed to further improve the performance.

In multi-path provisioning, network traffic can be split, and diversely routed. The traffic is then subject to the differential delay among different paths due to the different length. Differential Delay Constraint (DDC) is addressed in [14]. DDC was considered as a constraint in finding multiple working paths in [14]. A routing algorithm based on DDC was proposed.

Reference [15] studies the routing problem in the framework of constrained optimal control theory. Solving the routing problem becomes designing a controller. An optimal state feedback routing controller was designed in this article to solve the routing problem.

Multi Path provisioning splits and diversely routes the working traffic. The proposed YinYang algorithm applies splitting in its working traffic, to get better load balancing.

2.1.4 p-Cycle Protection Schemes

With p-Cycle protection schemes, protection paths are preconfigured in the network. The spare capacity for link protection is preconfigured in cycles. These spare capacities are also shared among the on-cycle links (the links that are on the preconfigured protection paths) and the straddling links (the links that are not on the preconfigured protection paths but are protected by the preconfigured protection paths) of the cycle [16]. The p-Cycle scheme achieves the real-time restoration speed associated with ring protection while achieving the high efficiency of mesh restoration [17].

In [18] Integer Linear Programming (ILP) is applied in planning static traffic. Integer Linear programming refers to a broad class of mathematical programming techniques characterized by a large number of variables which interact within boundaries imposed by some constraints [19]. Optimal solutions to different network resource allocation problems can be acquired by applying this technique. In reference [18], first the minimum backup capacity against any single link failure is set up. The next step generates the most likely candidate routes for each demand. These candidate routes are used to formulate the overall problem as an ILP problem to minimize the total capacity required.

The planning of the reconfigured protection path has direct influence on the performance of the algorithm. In [20] protection paths are optimized to improve capacity efficiency and recovery delay by extending and reshaping their structures.

The calculation of this scheme takes relatively a long time comparing with heuristic algorithms. So the p-Cycle protection schemes are applied at the design stage for static traffic. In this thesis the proposed YinYang algorithm and other benchmark algorithms are heuristic algorithms, which calculate the paths at real-time for random traffic.

2.2 Other related topics about Protection and Restoration Algorithms

2.2.1 Availability in Protection and Restoration Algorithms

Different demands require different network availabilities. Reference [21] presents a provisioning framework to satisfy different network availability requests. An integer linear programming (ILP) approach is developed in [21] to provide static traffic with different availability requirements. The article achieves different requirements by applying different protection schemes (unprotected, dedicated, or shared protected).

Reference [7] and [8] apply multi-backup paths to achieve high availability. In [8], a $1+N$ protection scheme is used to satisfy the availability requirement of a demand with extremely high availability. In [7], an $M+N$ multi-path provision scheme was proposed to satisfy service-level agreement (SLA) of different demands.

2.2.2 Protection and Restoration Algorithms in Multi-domain Networks

In [22], shared path restoration algorithms between multiple domains were proposed. Only certain nodes on the boundary of different domains exchange limited information. These nodes in different domains are aggregated information as a virtual single-domain network.

The routing algorithm in this virtual path network was evaluated and simulated in reference [22].

In [23], a demand's end-to-end quality of service (QoS) was investigated between multiple domains. A border Gateway Protocol (BGP) to exchange QoS information between different domains was developed. A QoS-based routing algorithm was proposed.

2.3 Benchmark Restoration Algorithms

2.3.1 1+1 Dedicated Path Protection (1+1 DPP)

The network is represented as $G = (V, E, LC)$, where V represents the set of nodes in the network, E represents the set of links and LC represents the capacity of all links. A Connection is represented as $t = (S, D, C)$, where S and D represent the Source and Destination nodes, and C represents the bandwidth required by a new demand. There are two steps to finding working and backup paths for the demand requiring a dedicated path protection.

In the first step: find a shortest path with enough bandwidth to accommodate the working path. Dijkstra's algorithm is used to compute the least-cost path. The cost assigned to link i to calculate the working path is:

$$Cost\ P_{Working}\ (i) = \begin{cases} \infty & C > A_i \\ \varepsilon & C \leq A_i \end{cases} \qquad (2.1)$$

A_i is the available bandwidth on link i. ε is a small positive integer, set to 1 in this thesis.

In the second step: find a link-disjoint path from the working path found in the first step, with enough bandwidth to accommodate the same bandwidth as the working path. Dijkstra's algorithm is used to compute the second least-cost path. The cost assigned to link j to calculate the backup path is:

$$Cost\ P_{Backup}\ \ (j) = \begin{cases} \infty & j \in P_{Working} \\ \varepsilon & C \leq A_j \\ \infty & C > A_j \end{cases} \qquad (2.2)$$

In order to find a link-disjoint backup path, the cost of a link on the working path will be set to infinite. A_i is the available bandwidth on link j. ε is a small positive integer (set to 1 in this thesis).

If both paths can be found, C units of bandwidth will be allocated to the demand on all links along the computed working path and the backup path.

2.3.2 Simple Pool Sharing (SPS)

"Pool sharing ensures that the amount of backup bandwidth required on a link to restore the failed connections that result from any single link failure will not be more than the total amount of backup bandwidth reserved on that link." [24].

The network is represented as $G = (V, E, LC)$. There are J links in the network. The Connection is represented as $t = (S, D, C)$. With pool sharing, the working and backup paths for every demand are computed in two steps.

In the first step: find a shortest path with enough bandwidth to accommodate the working path. Dijkstra's algorithm is used to compute the least-cost path. The cost assigned to link i to calculate the working path is:

$$Cost\ P_{Working}\ (i) = \begin{cases} \infty & C > A_i \\ \varepsilon & C \leq A_i \end{cases} \qquad (2.3)$$

The second step is finding a link-disjoint backup path, whose bandwidth can be shared by other demands. In pool sharing, the records of the backup bandwidth reserved on links are represented in the following matrix:

$$\Phi = \begin{bmatrix} 0 & \varphi_{12} & \varphi_{13} & \cdots & \varphi_{1J} \\ \varphi_{21} & 0 & \varphi_{23} & \cdots & \varphi_{2J} \\ \varphi_{31} & \varphi_{21} & 0 & \cdots & \varphi_{3J} \\ \cdots & \cdots & & & \cdots \\ \varphi_{J1} & \varphi_{J2} & \varphi_{J3} & \ldots & 0 \end{bmatrix} \qquad (2.4)$$

φ_{ij} is the amount of backup bandwidth needed on link j if link i fails.

The total amount of backup bandwidth needed on link j (B_j) is the maximum of all elements in column j:

$$B_j = max\,[\varphi_{ij}]\ \ 1 \leq i \leq J \qquad (2.5)$$

Assuming a working path has been found in step 1. Now define the following additional notations:

$S_w(r)$: set of links along the working path of demand r.

$S_b(r)$: set of links along the backup path of demand r.

Set $S_w(r)$ is known from step 1. Let T_j denote the maximum amount of backup bandwidth required on link j if a link in $S_w(r)$ fails.

$$T_j = C + max\,[\varphi_{ij}]\ \forall i \in S_w(r) \tag{2.6}$$

So the bandwidth of T_j is enough to restore current routing demand's working path failure (with C units of bandwidth) and accepted demands' working path failure (with $max\,[k_{ij}]\ \forall i \in S_w(r)$ units of bandwidth).

In order to compute a backup path for demand r, the following cost function is used to set the cost of every link j:

$$\theta_j = \begin{cases} \infty & if & j \in Sw(r) \\ \varepsilon & elseif & T_j \leq B_j \\ T_j - B_j & elseif & T_j - B_j \leq R_j \\ \infty & otherwise & \end{cases} \tag{2.7}$$

In this cost function:

Line 1: Set the cost of links on the working path to infinite, so that Dijkstra's algorithm will not choose these links for the backup path. In this way the backup path will be link-disjoint from the working path.

Line 2: In this case, the demand t can be restored on the link j without needing to reserve additional backup bandwidth on this link. This is where the "Sharing" occurs. The cost on link j: "ε" is a very small number, so the routing algorithm will prefer to choose this kind of "Sharing" link for the backup path.

Line 3: $T_j - B_j$ will be the cost of additional backup bandwidth needed to be reserved on the link if the residual capacity on link j is large enough to accommodate this request. The backup path will prefer to use links without the need to reserve any additional backup bandwidth on this link.

Line 4: If the residual capacity on link j is not adequate to accommodate this additional bandwidth, the cost of link j will be set to infinite. This link will not be used as a link on the backup path.

Two steps are applied here. In step 1, the minimum hop working path for demand r is found. In step 2, the cost function of (2.7) is used to compute a shared backup path. If both paths are found then the bandwidth is allocated on the links along these paths. The demand is accepted according to the scheme below.

The bandwidth allocation follows two steps. First update the total reserved working bandwidth on the working path. The bandwidth of the newly arrived demand is simply added to the total working bandwidth already reserved on that link.

For the second step, update the backup bandwidth. The total reserved backup bandwidth on every link along the backup path is updated by updating the corresponding elements in

matrix Φ. Bandwidth C is added to element φ_{ij} for every link i along the computed working path and every link j along the backup path. That is,

$$\forall i \in S_w(r), \forall j \in S_B(r) : \varphi_{ij} = \varphi_{ij} + C. \quad (2.8)$$

After updating matrix Φ, the total reserved backup bandwidth on the backup links in S_B can be obtained from:

$$B_j = max [\varphi_{ij}] \ 1 \leq i \leq J \quad (2.9)$$

On some of these links, B_j may not change: in this case T_j must have been less than or equal to B_j when computing the backup path.

2.3.3 2:1 Multi Path Provisioning (2:1 MPP)

2:1 Multi-path provisioning splits a typical working path into two working paths each with half of the bandwidth of the original path

In Figure 2-1, there is a demand A-D with 2 units of working bandwidth and 2 units of backup bandwidth. When applying multi-path provisioning in Figure 2-2, the working path is split into two working paths. Each of them has one unit of bandwidth. The backup path only needs 1/2 of the bandwidth of the working path.

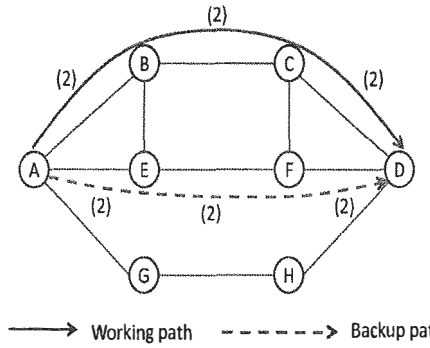


Figure 2-11+1 Dedicated Path Protection

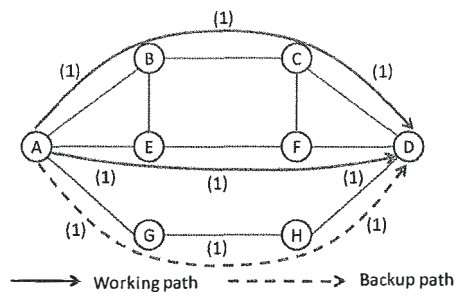


Figure 2-2 2:1 Multi-Path Provisioning

The backup path can accommodate one of the working paths. In the single link failure case, $\frac{1}{N}$ of the working traffic will travel though the backup path. The backup path along with the rest of the working paths will accommodate the traffic for the demand.

In Figure 2-3, link B-C fails, and path A-G-H-D replaces path A-B-C-D as the working path. This together with the path A-E-F-D will carry the entire traffic of the demand. Figure 2-4 shows that the failure happened between E-F, so path A-G-H-D and A-B-C-D accommodate the traffic.

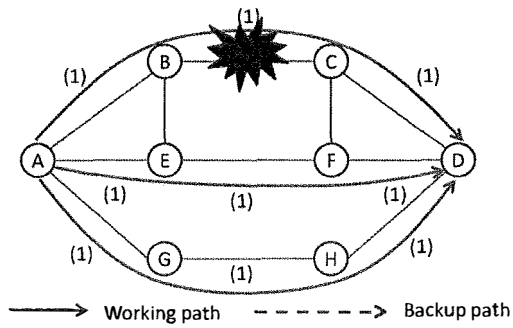


Figure 2-3 Protection Scheme (a)

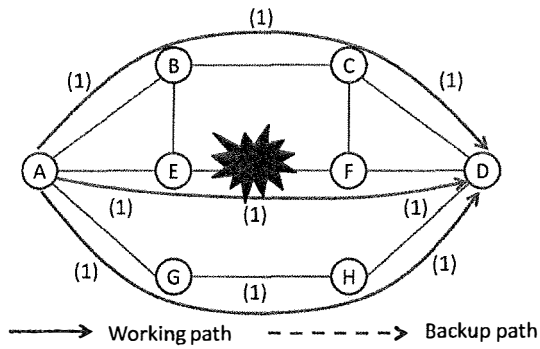


Figure 2-4 Protection Scheme (b)

The description of the 2:1 Multi Path Provisioning is:

The network is represented as $G = (V, E, LC)$. V represents all nodes, E represents all links and LC represents the link capacity. The Connection is represented as $t = (S, D, 2C)$, where S and D represent source and destination. $2C$ represents capacity required by the new demand. With 2:1 multi-path protection, there are three steps to compute the two working paths and the backup path.

In the first step: find a shortest path with enough bandwidth to accommodate the first working path with C units of bandwidth. Dijkstra's algorithm is used to compute a least-cost path. The cost assigned on link i to calculate the working path is:

$$Cost\ P_{W1}\ \ (i) = \begin{cases} \infty & C > A_i \\ \varepsilon & C \leq A_i \end{cases} \tag{2.10}$$

A_i is the available bandwidth on link *i*. ε is a small positive integer, equal to 1 in this thesis. Assume that the first working path has been found. Now define the following additional notation:

$S_{w1}(r)$: set of links along the working path just found.

In the second step: find a second shortest path with enough bandwidth to accommodate the second working path with *C* units of bandwidth. Dijkstra's algorithm is used to compute this path. The cost assigned to link *i* to calculate the second working path is:

$$Cost\ P_{W2}\ \ (i) = \begin{cases} \infty & C > A_i \\ \infty & i \in Sw1(r) \\ \varepsilon & C \leq A_i \end{cases} \tag{2.11}$$

A_i is the available bandwidth on link *i*. ε is a small positive integer, equal to 1 in the simulation model of this thesis. Assume that the second working path has been found. Now define the following additional notation:

$S_{w2}(r)$: set of links along the second working path just computed.

In the third step: find a link-disjoint path from the working paths found in the first and second step, with enough bandwidth to accommodate the same bandwidth as each working path. Dijkstra's algorithm is used to compute this least-cost path. The cost assigned on link *j* to calculate the backup path is:

$$Cost\ P_{Backup}\ \ (j) = \begin{cases} \infty & j \in S_{W1}(r) \\ \infty & j \in S_{W2}(r) \\ \varepsilon & C \leq A_j \\ \infty & C > A_j \end{cases} \tag{2.12}$$

If all three paths can be found, C units of bandwidth among the two working paths and backup paths will be assigned, or the demand will be rejected.

For a demand in the 2:1 Multi-path provisioning, the link capacities on the backup path are reserved exclusively for different demands.

Chapter 3

Proposed Two-Path YinYang Restoration Schemes

3.1 Introduction of YinYang concept

YinYang is a Traditional Chinese philosophical concept that is used to explain how contrary powers are interconnected and interdependent in the natural world, and they give rise to each other in turn. The symbol of YinYang is shown in Figure 3-1.

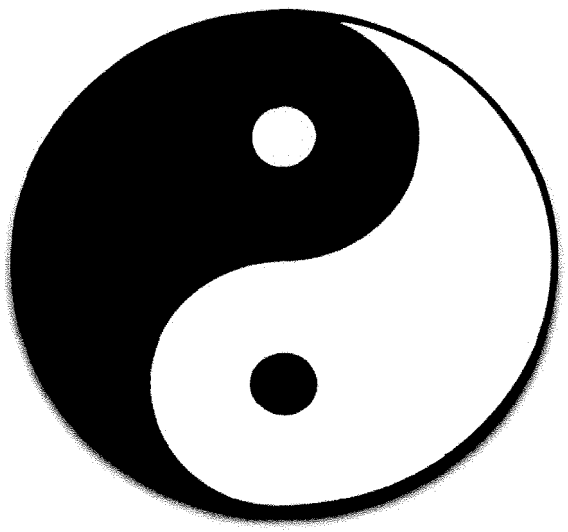


Figure 3-1 The symbol of YinYang

3.2 Premise of the Two-Path YinYang algorithm

This thesis proposes the YinYang algorithm based on two fundamental principles of the YinYang concept: "Treat contrary things as a whole" and "The contrary things keep changing".

Firstly, the YinYang concept believes in complementary opposites that interact within a greater whole, as part of a dynamic system. The YinYang algorithm treats the working path and backup path as a whole.

Take 1+1 Dedicated Path Protection as an example. In the first step, it finds the first shortest path for the working path. In the second step, it finds the second shortest link-disjoint path as the backup path. When more demands arrive, repeat this process. When finding the first path, 1+1 DPP only considers finding the lowest-cost path for the first path, and doesn't consider the backup path or other demands that will arrive after it. So the backup path has to find the shortest path from the links left from the working path. The connection demands arriving later have to find their path from the leftover network resources of the already arrived demands.

The working path and backup path are inseparable in protection and restoration schemes. The YinYang algorithm considers the working and backup paths as a whole. Instead of finding the best path for the working path first and then for the backup, it finds the best two paths for the working and backup paths at the same time.

Secondly, the YinYang concept believes contrary powers give rise to each other in turn and prefer to achieve a kind of balance. The YinYang algorithm starts with one working path and one backup path found by Shared Path Restoration. After several iterations, the working path and backup path are mixed together, and each path will be composed of an equal portion of the working traffic and the corresponding backup traffic. This new fundamental traffic planning scheme splits the traffic without physically finding an additional path to achieve the same result. Simulation results show significant improvement for relieving congested links and improving demand acceptance ratio.

The YinYang algorithm treats its working and backup paths as a whole. It also prefers sharing backup bandwidth with existing demands at the routing stage. It treats all the demands in the network as a whole. By slightly sacrificing the efficiency (with longer paths) of every demand, the network resources are more fairly utilized by different demands, so the total acceptance ratio is improved.

3.3 Two-Path YinYang restoration appearance

Figure 3-2 shows a demonstration network with a connection demand from source A to destination F, requesting 100 units of bandwidth. Assume two paths are found for this demand. The first path is ABCF. The second path is from nodes AEGF, through links, 4, 5 and 6.

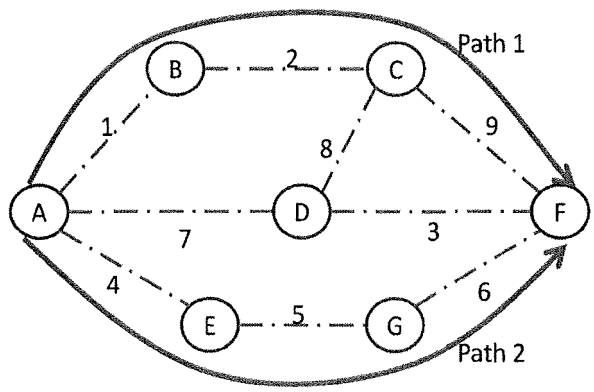


Figure 3-2 Demonstration Network

To better demonstrate Two-Path YinYang restoration, shared path restoration is shown here. Shared path restoration will assign traffic on this network as in Figure 3-3. The working path is ABCF with 100 units of bandwidth, and the shared backup path is AEGF, also with 100 units of bandwidth.

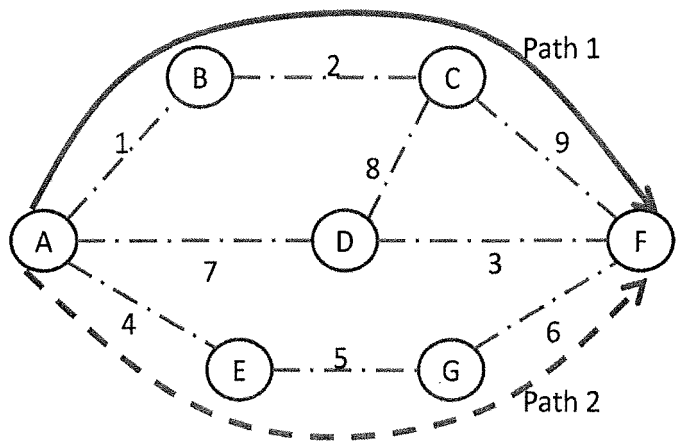


Figure 3-3 Shared path Restoration traffic assignment

3.3.1 Two-Path YinYang restoration traffic assignment

Two-Path YinYang restoration will assign traffic on this network as in Figure 3-4. The first path is ABCF with 50 units of working bandwidth and 50 units of backup bandwidth. The second path is AEGF, also with 50 units of working bandwidth.

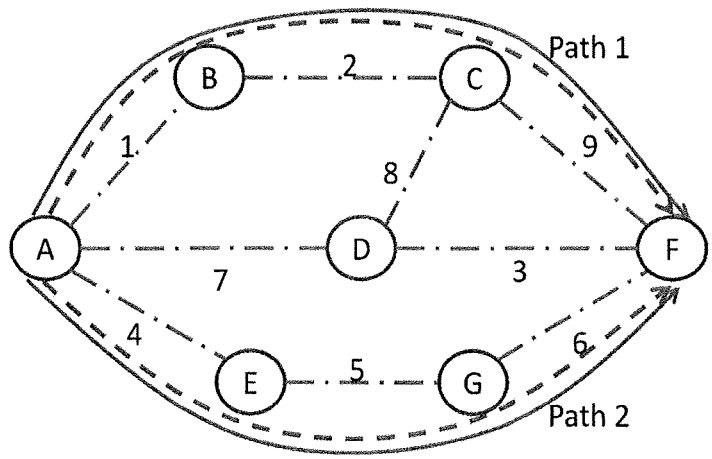


Figure 3-4 Two-Path YinYang restoration traffic assignment

3.3.2 Two-Path YinYang restoration failure operation

Two-Path YinYang restoration can recover from single link failure among its two paths. Figure 3-5 shows when link failure happened at link 2 (between node B and node C). Path 1 with 50 units of working traffic from ABCF will not work. The backup capacity reserved on Path 2 will accommodate the failed working traffic of Path 1.

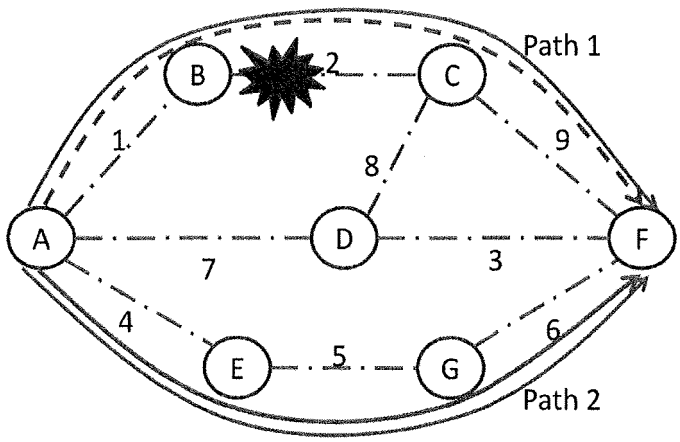


Figure 3-5 Two-Path YinYang restoration failure operation (1)

Figure 3-6 shows when link failure happened at link 5. Path 2 with 50 units of working traffic from AEGF, will not work. The backup capacity reserved on Path 1 will accommodate the failed working traffic of Path 2.

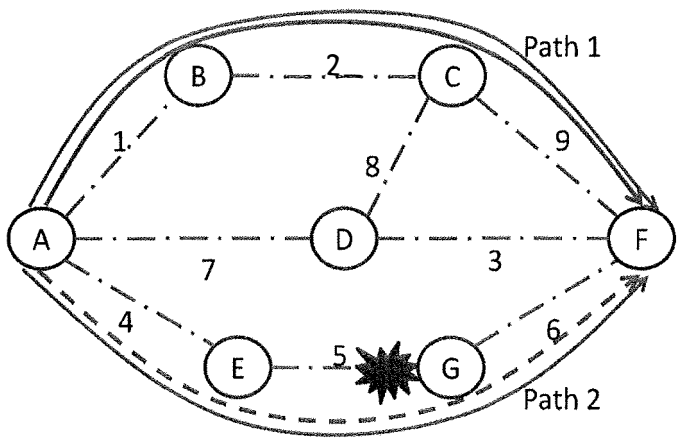


Figure 3-6 Two-Path YinYang restoration failure operation (2)

3.3.3 Demonstration of load balancing

Section 2.3.3 introduced Simple Pool Sharing. It is used here as a benchmark algorithm to show the load balancing performance of the Two-Path YinYang restoration.

Here we use the same example (same network, same demand and same paths) as section 3.3.2. In Two-Path YinYang restoration, two paths are found as in Figure 3-7:

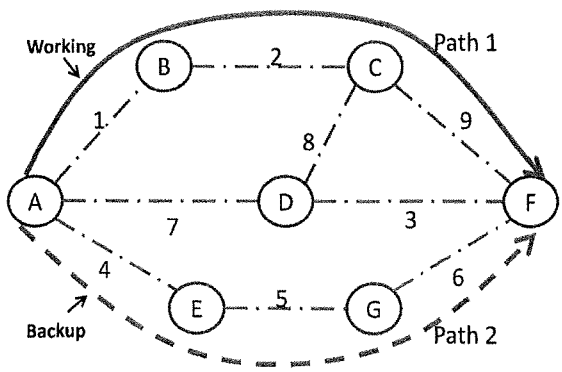


Figure 3-7 Two paths for simple pool sharing

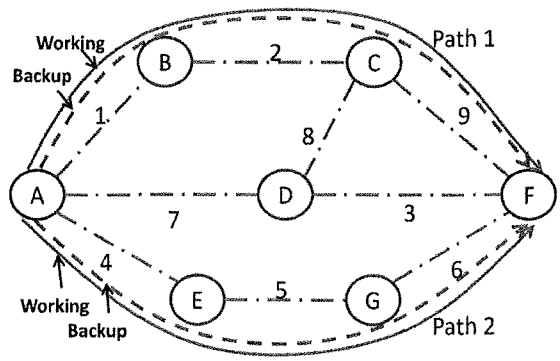


Figure 3-8 Paths for YinYang restoration

Now the backup bandwidth for simple pool sharing is calculated. The records of the backup bandwidth reserved on links are represented in the following matrix:

$$\alpha = \begin{bmatrix} 0 & 0 & 0 & 100 & 100 & 100 & 0 & 0 & 0 \\ 0 & 0 & 0 & 100 & 100 & 100 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 100 & 100 & 100 & 0 & 0 & 0 \end{bmatrix}$$

Figure 3-9: α, the Backup matrix for SPS

$$\beta = \begin{bmatrix} 0 & 0 & 0 & 50 & 50 & 50 & 0 & 0 & 0 \\ 0 & 0 & 0 & 50 & 50 & 50 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 50 & 50 & 0 & 0 & 0 & 0 & 0 & 0 & 50 \\ 50 & 50 & 0 & 0 & 0 & 0 & 0 & 0 & 50 \\ 50 & 50 & 0 & 0 & 0 & 0 & 0 & 0 & 50 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 50 & 50 & 50 & 0 & 0 & 0 \end{bmatrix}$$

Figure 3-10: β, the Backup matrix for YinYang

In Figure 3-9, the backup bandwidth is allocated in three columns, while in Figure 3-9, backup bandwidth is allocated in six columns. The total amount of backup bandwidth is the same in these two matrices.

Define: the backup bandwidth matrix for SPS in Figure 3-9 as α, and the backup bandwidth matrix for YinYang in Figure 3-9 as β:

$$\beta = \alpha/2 + (\alpha/2)^{T} \tag{3.1}$$

Equation (3.1 shows how the YinYang scheme reduces the maximum value of the backup matrix by a factor of two (the first term on right side) and allocates these values more evenly by transposing half of the backup matrix α to different columns (the second term on the right). This mathematical transform in the backup matrix causes a significant backup bandwidth allocation change.

The total amount of backup bandwidth needed on link j (B_j) is the maximum of all elements in column j:

$$B_j = max\,[\varphi_{ij}] \quad \forall i \in I \tag{3.2}$$

So the backup bandwidth needed on the links in this network is in Table 3-11 (for SPS) and Table 3-12 (for YinYang):

Link:	1	2	3	4	5	6	7	8	9
Bandwidth:				100	100	100			

Table 3-11 Backup bandwidth allocation for SPS

Link:	1	2	3	4	5	6	7	8	9
Bandwidth:	50	50		50	50	50			50

Table 3-12 Backup bandwidth allocation for YinYang restoration

To better illustrate the load-balancing improvement, bandwidth distributions of the two paths for simple pool sharing (SPS) and the Two-Path YinYang restoration are shown side-by-side, in Figure 3-13, Figure 3-14 and Figure 3-15.

Firstly backup sharing bandwidth is shown in a bar graph in Figure 3-13. In SPS, the backup bandwidth occupies 100 units on links 4, 5 and 6, while in YinYang, the backup bandwidth occupies 50 units on links 1, 2, 4, 5, 6 and 9. In YinYang, the backup bandwidth is evenly distributed on more links instead of concentrated on few links.

The data for the working bandwidth is in a bar graph (Figure 3-14). Similar to sharing backup bandwidth, working bandwidth is distributed more evenly with YinYang.

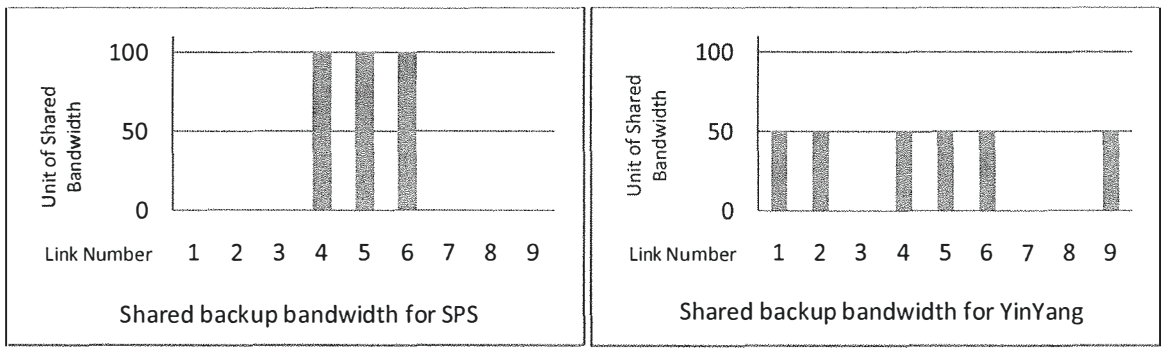


Figure 3-13 Comparison for backup sharing bandwidth of SPS and YinYang

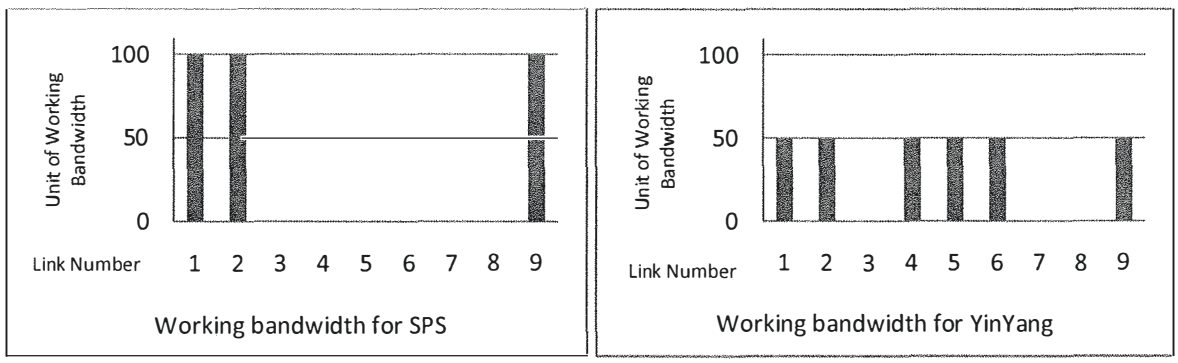


Figure 3-14 Comparison for working bandwidth of SPS and YinYang

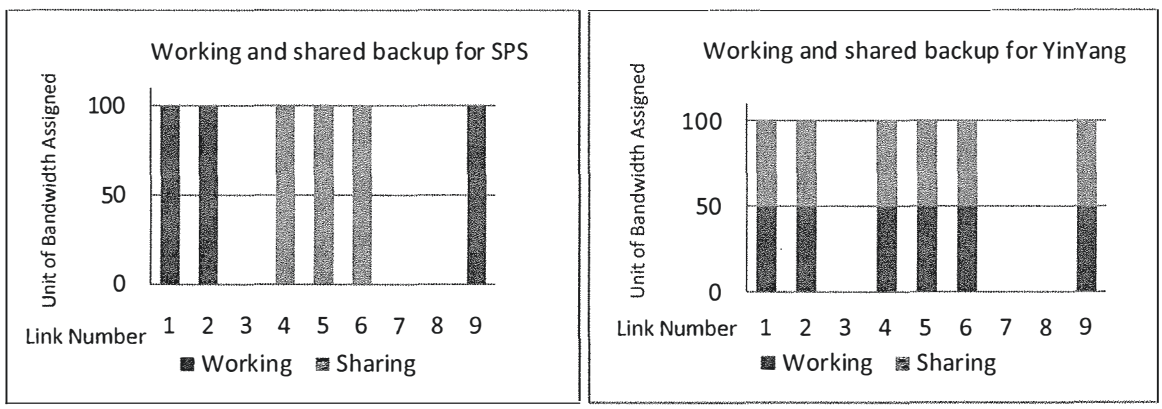
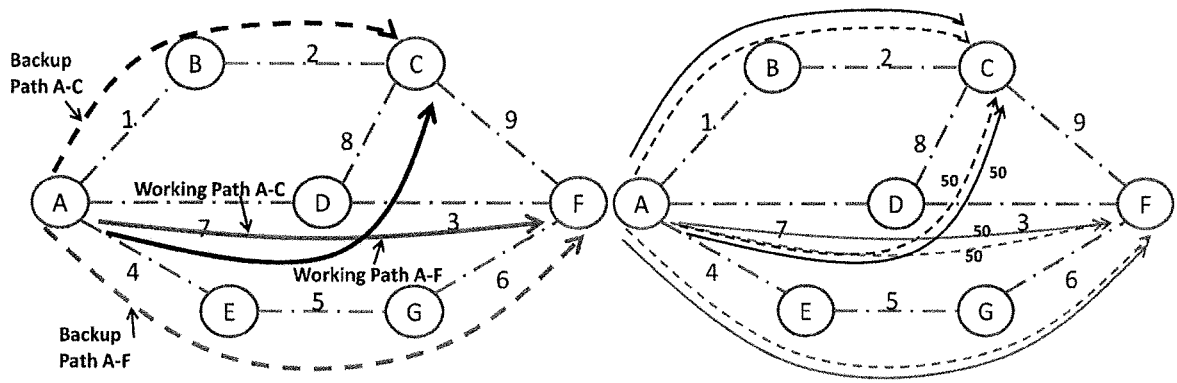


Figure 3-15 Comparison for working and sharing backup bandwidth of SPS and YinYang

When we put the working and shared backup bandwidth together in a bar graph (Figure 3-15), we can get the following conclusions:

Firstly, YinYang doesn't have large working bandwidth like 100 units. This kind of bandwidth can easily cause congestions on an overloaded link. Secondly, in the YinYang algorithm, sharing backup bandwidth is on the same link as the working bandwidth. This bandwidth may be shared with other demands. This is a mechanism that relieves the congestions of overloaded links. Thirdly, this distributed backup bandwidth enhances the possibility of different demands sharing the backup bandwidth.



(A) Two-path simple pool sharing (B) Two-Path YinYang restoration

Figure 3-16 Congestion relief example

The bandwidth sharing mechanism is shown in Figure 3-16. Two demands A-C and A-F in the same network, both requesting 100 units of bandwidth. In Figure 3-16(a), two working paths of simple pool sharing are allocated on link 7. These two working paths will take 200 units of bandwidth. In Figure 3-16(b), two paths of YinYang restoration are allocated on link

7. Because the backup bandwidth can be shared on link 7, they will share 50 units of bandwidth. The working bandwidth on link 7 for demands A-C and A-F will take 50 units each, so the total bandwidth needed on link 7 is 50 units for backup and 2×50 units for working, giving 150 units in total.

3.4 Two Path YinYang Algorithm Routing Problem

In the first three sections, we introduced Two-Path YinYang Algorithm's traffic assignment, failure operation and load balancing performance. This section will introduce the routing technique that can be used to compute the paths for every demand. These paths were already demonstrated for the examples shown in the previous section.

3.4.1 Routing Challenge for the Two-Path YinYang Algorithm

The simple pool sharing scheme (mentioned in section 2.3.2) finds the working path first. Based on the working path and other demands already in the network, a backup path that can accommodate working path will be found. The newly-found working path and the accepted demands determine the cost for a link to be chosen as the backup path. Then the least-cost backup path is found. Without a working path, the amount of bandwidth needed on the backup path cannot be calculated.

But in the Two-Path YinYang algorithm both of its paths are working and backup path at the same time. One path cannot be found without the other. The YinYang algorithm faces a logical paradox at the routing stage.

The inspiration for solving this problem is also from the YinYang concept. The YinYang concept believes contrary powers give rise to each other in turn and prefers to achieve a kind of balance. The YinYang algorithm starts with path 1 (working path) and path 2 (backup path) found by Shared Path Restoration, as in Figure 3-7. Then in several rerouting iterations, the portion of backup bandwidth in path 1 increases from 0% to 50%, while at the same time, the portion of backup bandwidth in path 2 decreases from 100% to 50%. At the end, each path will be composed of an equal portion of the working traffic and the corresponding backup traffic, as in Figure 3-7. The demand will be rejected if at the initial stage or rerouting stages any of these paths cannot be found.

3.4.2 Two-Path YinYang Restoration Algorithm

Input: Network topology *G= (V, E, Capacity).* Connection request *t= (S, D, 2C).*

V represents all the nodes, *E* represents all the links and *Capacity* represents the capacity for all the links. The Connection is represented as *t = (S, D, 2C). S* and *D* represent Source and Destination. *2C* is the bandwidth requested by the demand.

Output: two paths, P_1 and P_2, that are link-disjoint. The bandwidth request for $P_1=2C$, and the bandwidth request for $P_2=2C$. On the links along P_1 and P_2, *C* units of bandwidth are used as the working path and another *C* units of bandwidth are used as the backup path.

Step 1: *[Initialization]:* In this step, the working and backup paths are found in a manner identical to Single Pool Sharing. The connection request will be rejected if any of these two paths cannot be found.

Step 1.1: Find the shortest path as the working path $P_1^{(0)}$. The bandwidth of this path is $2C$.

$$Cost\ P_1^{(0)}(i) = \begin{cases} \infty & 2C > A_i \\ \varepsilon & 2C \leq A_i \end{cases} \tag{3.3}$$

Dijkstra's algorithm is used with the above cost assignment in order to compute a least-cost path.

Step 1.2: Find one backup path $P_2^{(0)}$. The bandwidth of this path is $2C$.

The backup bandwidth reserved on links is in the following matrix:

$$\Phi = \begin{bmatrix} 0 & \varphi_{12} & \varphi_{13} & \cdots & \varphi_{1J} \\ \varphi_{21} & 0 & \varphi_{23} & \cdots & \varphi_{2J} \\ \varphi_{31} & \varphi_{21} & 0 & \cdots & \varphi_{3J} \\ \cdots & \cdots & & & \cdots \\ \varphi_{J1} & \varphi_{J2} & \varphi_{J3} & \ldots & 0 \end{bmatrix} \tag{3.4}$$

φ_{ij} is the amount of bandwidth needed on link j if link i fails. ($1 \leqslant i, j \leqslant J$). J is the number of links in the network.

The total amount of backup bandwidth needed on link j (B_j) is the maximum of all elements in column j:

$$B_j = max\,[\varphi_{ij}] \quad 1 \leqslant i \leqslant J \tag{3.5}$$

Denote: $S_w(r)$, the set of links along the $P_1^{(0)}$ path of demand request t.

Denote: $T_j^{(0)}$, the maximum amount of backup bandwidth required on link j if a link in $S_w(r)$ fails.

$$T_j^{(0)} = 2C + max\,[\varphi_{ij}] \quad \forall i \in S_w(r) \tag{3.6}$$

The cost assigned on link j to find $P_2^{(0)}$ is:

$$Cost\ P_2^{(0)}(j) = \begin{cases} \infty & j \in \boldsymbol{P_1}^{(0)} \\ \varepsilon & T_j^{(0)} \leq B_j \\ T_j^{(0)} - B_j & 0 < T_j^{(0)} - B_j \leq A_j \\ \infty & otherwise \end{cases} \tag{3.7}$$

Dijkstra's algorithm is used with the above cost assignment in order to compute a least-cost path.

Step 2: In this step, the working and backup paths are mixed. When we arrive at this step, we already have one working path ("Path 1") and one shared backup path ("Path 2"), as in Figure 3-17 (a). Then we delete Path 1 (Figure 3-17 (b)) and assume that 1% of Path 2 is working and 99% is still backup (Figure 3-17 (c)). We re-route Path 1 with a path that can provide protection for Path 2 (Figure 3-17 (d)). This new Path 1 will be 1% backup and 99% working. We then delete Path 2 (Figure 3-17 (e)) and assume that Path 1's ratio is 2% backup and 98% working (Figure 3-17 (f)), re-route Path 2 (Figure 3-17 (g)), and so on until the ratio is 50%/50% (Figure 3-17 (h)). In general, the paths may change from one iteration to the next iteration. However, for ease of discussion, the paths shown in this example do not change. The connection request will be rejected if any of these two paths cannot be found during the rerouting stages.

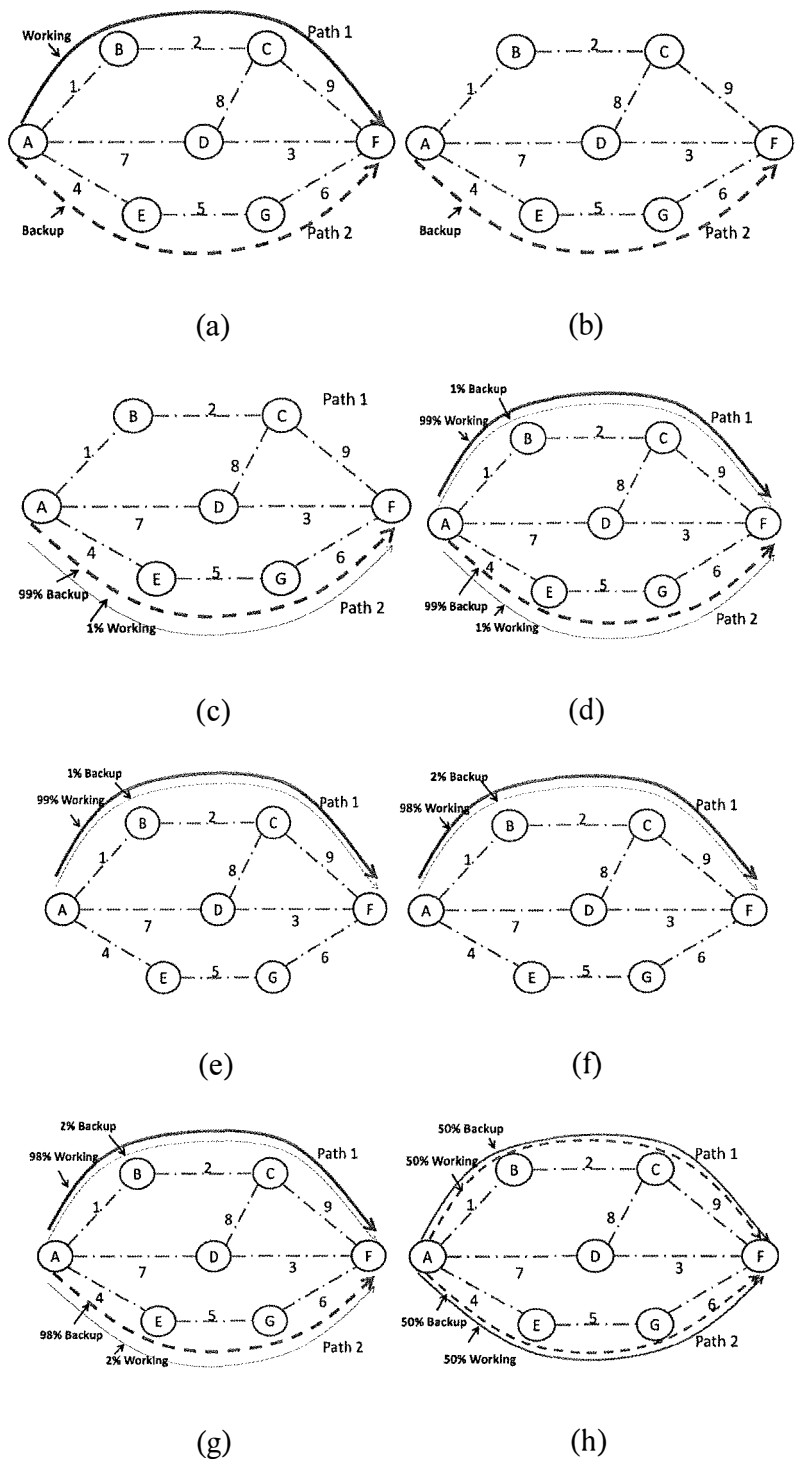


Figure 3-17 Two-path YinYang rerouting iterations

Denote n as the index for re-routing iteration, where $\frac{n}{100}$ is the portion of the backup bandwidth in P_1. In step 2, n changes from 1 to 50, so the portion of the backup bandwidth will go from 1% to 50%.

For (n = 1; n <= 50; n ++)

{

Step 2.1: Keep $P_2^{(n-1)}$, Find the path $P_1^{(n)}$ to replace $P_1^{(n-1)}$.

$$Cost\ P_1^{(n)}(j) = \left(1 - \frac{n}{100}\right) * CostWork_{p1}^{(n)}(j, n) + \left(\frac{n}{100}\right) * CostBackup_{p1}^{(n)}(j, n) \quad (3.8)$$

Step 2.2: Keep $P_1^{(n)}$, Find the path $P_2^{(n)}$ to replace $P_2^{(n-1)}$.

$$Cost\ P_2^{(n)}(j) = \left(\frac{n}{100}\right) * CostWork_{p2}^{(n)}(j, n) + \left(1 - \frac{n}{100}\right) * CostBackup_{p2}^{(n)}(j, n) \quad (3.9)$$

}

Formulas (3.8) and (3.9) show the cost function for Path 1 and Path 2 during the 50 re-route iterations. Take Formula (3.8) for example: the first line is the weight of working bandwidth in Path 1 multiplied by the cost of working bandwidth. The cost of working bandwidth of Path 1 (3.10) is similar to simple pool sharing, but with an index n to change the proportion according to the number of iterations. The second line is the weight of backup bandwidth in

Path 2 multiplied by the cost of backup bandwidth. The cost of backup bandwidth of Path 2 (3.11) is similar to simple pool sharing, but with an index n to change the proportion according to the number of iterations. Formula (3.9) shows the cost function for Path 2 following a similar pattern.

$$CostWork_{p1}^{(n)}(j,n) = \begin{cases} \infty & \left(1-\frac{n}{100}\right) * 2C > A_j \\ \varepsilon & \left(1-\frac{n}{100}\right) * 2C \leq A_j \end{cases} \tag{3.10}$$

$$CostBackup_{p1}^{(n)}(j,n) = \begin{cases} \infty & j \in \boldsymbol{P_2^{(n-1)}} \\ \varepsilon & T'_j{}^{(n)} \leq B_j - \left(1-\frac{n}{100}\right) * 2C \leq A_j \\ T'_j{}^{(n)} - B_j & 0 < T'_j{}^{(n)} - B_j \leq A_j - \left(1-\frac{n}{100}\right) * 2C \\ \infty & otherwise \end{cases} \tag{3.11}$$

$$T'_j{}^{(n)} = \frac{n}{100} * 2C + max\,[\varphi_{ij}];\ \forall i \in P_2^{(n-1)} \tag{3.12}$$

$$CostWork_{p2}^{(n)}(j,n) = \begin{cases} \infty & \frac{n}{100} * 2C > A_j \\ \varepsilon & \frac{n}{100} * 2C \leq A_j \end{cases} \tag{3.13}$$

$$CostBackup_{p2}^{(n)}(j,n) = \begin{cases} \infty & j \in \boldsymbol{P_1^n} \\ \varepsilon & T''_j{}^{(n)} \leq B_j - \frac{n}{100} * 2C \leq A_j \\ T''_j{}^{(n)} - B_j & 0 < T''_j{}^{(n)} - B_j \leq A_j - \frac{n}{100} * 2C \\ \infty & otherwise \end{cases} \quad (3.14)$$

$$T''_j{}^{(n)} = \left(1 - \frac{n}{100}\right) * 2C + max[\varphi_{ij}];\ \forall i \in P_1^{(n)} \quad (3.15)$$

Step 3: Assign bandwidth to the two paths.

3.5 Multi-Path YinYang Restoration

3.5.1 Bandwidth allocation for Multi-Path YinYang Restoration

If *N* paths can be found between the source and destination, theoretically, we can perform *N*-Path YinYang restoration as in Figure 3-18.

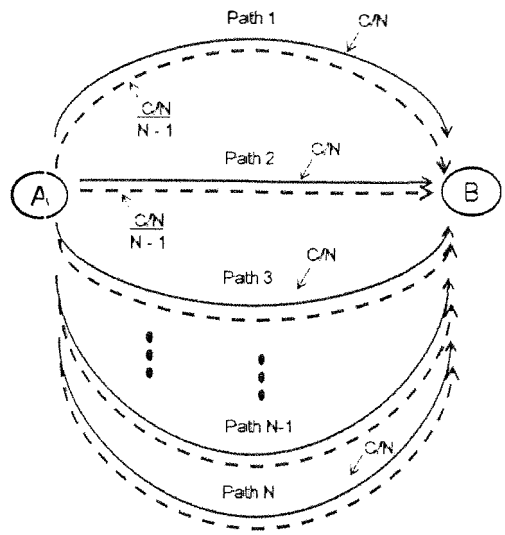


Figure 3-18 N-Path YinYang algorithm

The bandwidth for the connection request is *C*; the bandwidth requested for the working portion on a path is:

$$B_{Working} = \frac{C}{N} \tag{3.16}$$

If one path failed in one of the working paths, the backup capacity among the other *N*-1 paths will accommodate the *C/N* units of working bandwidth on the failed link. So the bandwidth requested for the backup portion on a path is:

$$B_{Backup} = \frac{C}{N(N-1)} \tag{3.17}$$

3.5.2 Bandwidth allocation for Three-Path YinYang Restoration

In Three-Path YinYang restoration, the bandwidth of the working traffic equals 1/3 of the demand's request and the bandwidth of the backup traffic equals 1/6 of the demand's request.

Assuming 150 units of bandwidth are needed between nodes A and B, the bandwidth allocation between them by the Three-Path YinYang algorithm is shown in Figure 3-19.

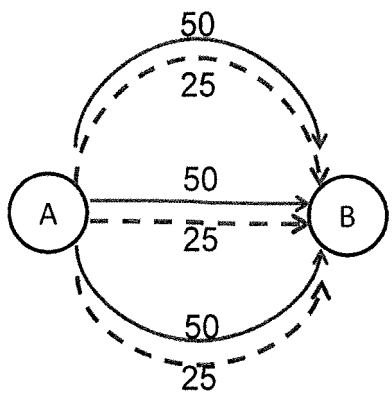


Figure 3-19 Bandwidth allocation for the Three-Path YinYang algorithm

Chapter 4

Simulation Results

4.1 Introduction

In this section, we will compare the performance of the Two-Path YinYang restoration scheme proposed in section 3.4.2 with benchmark algorithms in section 2.3.1 (1+1 Dedicated Path Protection) and 2.3.2 (The Simple Pool Sharing).

4.2 Performance Evaluations

In-house simulations based on C++ were used to study the proposed YinYang algorithm on the existing North-American backbone networks, the Global Crossing network (GCN) and a modified version of the Global Crossing network (MGCN). In order to represent the performance of the Two-Path YinYang restoration, two benchmark algorithms (Simple Pool Sharing and 1+1 Dedicated Path Protection) are simulated at the same time. These two algorithms represent the trade-off between reliability and bandwidth sharing efficiency.

Four metrics are used in this work to measure the performance of the two-path YinYang algorithm and benchmark algorithms. These metrics show the reliability, availability, efficiency and the ability to recover from failure. They are defined separately in the following sections.

4.2.1 Demand Blocking Ratio

The demand blocking ratio is the metric for the number of demands that will be rejected. It is an important metric that will be improved (compare with benchmark algorithms) by the two-path YinYang restoration scheme. It shows the usability of an algorithm. If demand blocking ratio for an algorithm is lower than that of another algorithm, it means more demand requests will be accepted.

In the simulation, when a path is successfully computed and the bandwidth is allocated for a given demand, the "Counter of Accepted Demand" will be increased. If the demand cannot find its paths, the "Counter of Rejected Demand" will be increased. This metric is measured for every demand. By the end of the simulation, the average "Demand Blocking Ratio" of all the demands will be calculated for the test algorithm. The definition is as equation (4.1):

$$\text{Demand blocking ratio} = \frac{\text{The number of Rejected Demands}}{\text{Number of Rejected Demands+Number of Accepted Demands}} \quad (4.1)$$

4.2.2 Working Path and Backup Path Resource Consumption

The efficiency of an algorithm can be measured by the resources it consumes to provide survivability between nodes. In this thesis, the length of the working and backup paths in terms of number of hops is the metric to indicate the efficiency of the measured algorithm.

For shared restoration, the number of hops along a backup path is only counted for the first demand, because the later demands will share that backup path with the first demand.

In the simulation, when the paths are found and bandwidth is allocated for a given demand, the "working path and backup path resource consumption measurement" method will be invoked. By the end of the simulation, the average "working path and backup path resource consumption" of all demands will be calculated for the test algorithm. For every demand, the working and backup resources are measured according to the following definition (equation (4.2) and (4.3)):

Working Path Resource Consumption = Hop Requirement * Bandwidth Requested (4.2)

Backup Path Resource Consumption = Hop Requirement * Bandwidth Requested (4.3)

4.2.3 Restoration Failure Rate

In simulating the protection and restoration algorithm, restoration failure rate is also an important metric. An existing demand will fail only when it is not restorable. For the algorithms presented in this thesis, a single link failure will never have restoration failure. It occurs only on multiple failures. Restoration failure rate shows the probability of a demand being unrestorable when a link along its path fails. Restoration failure rate is the indicator for the effectiveness of the restoration algorithm.

When a link failure event happens in the simulation, "Try restoration" method will be invoked at all effected demands. If the demand cannot be restored, "Counter of Unrestorable Restoration" will be incremented. By the end of the simulation, the average restoration

failure rate of all demands will be calculated for the test algorithm. The definition is as equation (4.4):

$$\text{Restoration Failure Rate} = \frac{\text{Number of Unrestorable demands}}{\text{Total Number of Restoration Attempts}} \tag{4.4}$$

4.2.4 Availability

The availability is the ratio of the demand's working time to holding time. Demand working time is the difference between holding time and failure time. The definition for availability is in equation (4.5):

$$\text{Availability} = \frac{\text{Demand Holding time} - \text{Demand failure time}}{\text{Demand Holding time}} \tag{4.5}$$

4.3 Network Topologies

4.3.1 Global Crossing Network

In the simulations, two network topologies have been used. The first is the Global Crossing Network (GCN).The GCN is shown in Figure 4-1.

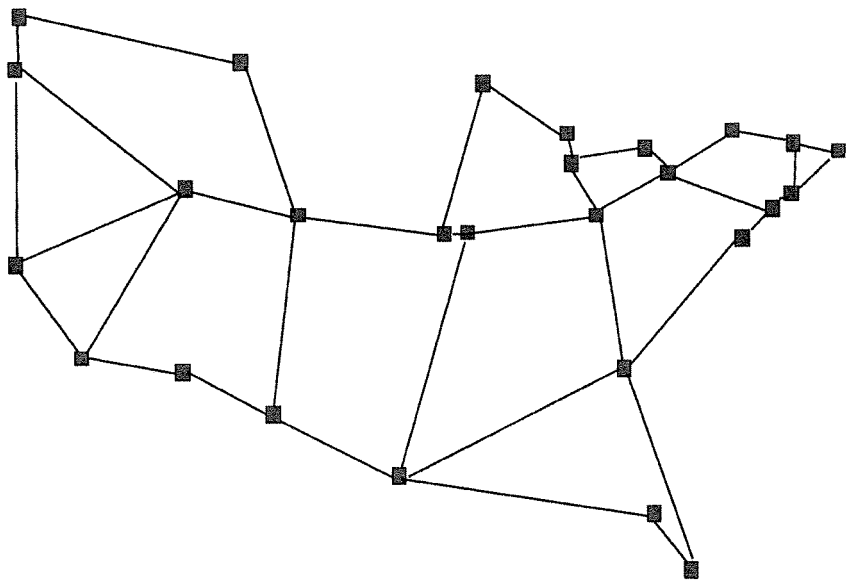


Figure 4-1 Global Crossing Network

In the GCN the average nodal degree is 2.81. There are 27 nodes and 38 bidirectional links in the GCN. The maximum nodal degree in the GCN is 4.

4.3.2 Modified Global Crossing Network

In order to get a network topology with a higher nodal degree needed to demonstrate the benefits of the YinYang algorithm, we modified the global crossing network, here referred to as MGCN. A few links have been added into the GCN in order to increase its average nodal degree. The MGCN is shown in Figure 4-2.

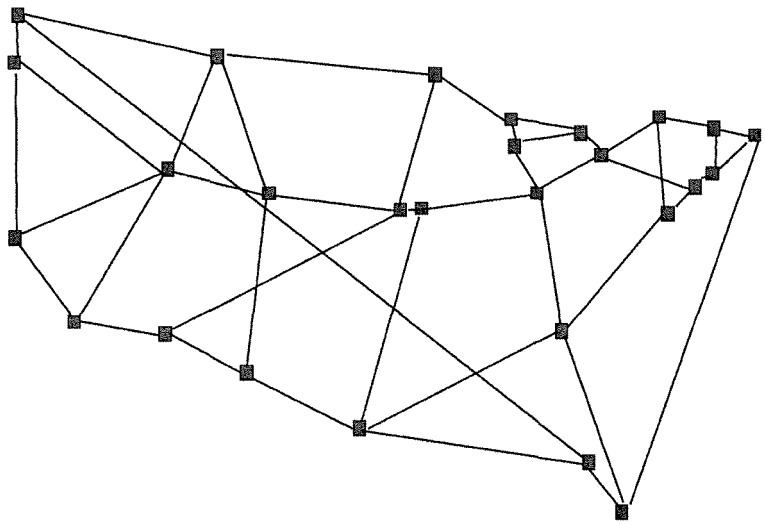


Figure 4-2 Modified Global Crossing Network

In the MGCN, the average nodal degree is 3.29. There are 27 nodes and 46 bidirectional links in the MGCN. The maximum nodal degree in the GCN is 5.

4.4 Simulation Model

The structure of the simulation model and some important simulation statistics for the simulation model will be introduced in this section.

4.4.1 Structure of the Simulation Model

The basic frame of the simulation model in this thesis follows the book chapter [24]: "Modeling and simulation of mesh networks with path protection and restoration." Object-oriented modeling based on C++ code and Unified Modeling Language techniques are used to describe the components, structure and behaviours of the network. As illustrated in Figure 4-3, bidirectional links interconnect all the nodes. The Network Agent collects the

information from all the network components and controls network resources to satisfy different demands.

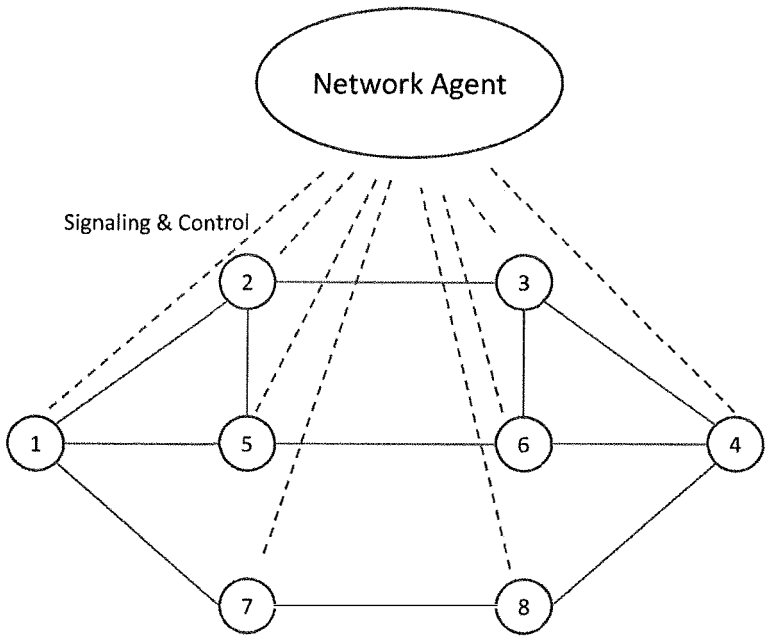


Figure 4-3 Network Model

The algorithms presented in previous chapters are methods in the "Network Agent" class. Event, Link, Node and Demand are classes that are similar to the descriptions in [24].

4.4.2 Overview of Network Simulation Behaviour

In Figure 4-4 is the state diagram of the network behaviour. The network changes through a series of discrete states during the simulation process.

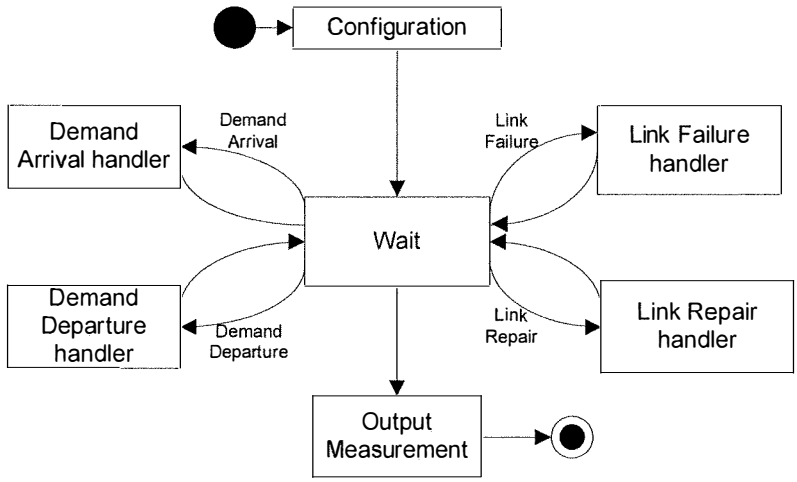


Figure 4-4 Network behavior state machine

The figure consists of the following states:

- Initial: a state denoted by a circle. It shows the starting point of the simulation.

- Configuration: represents the state that the network links, demands and nodes are configured and instantiated. During this state, simulation parameters are configured and counters for the measurement are initialized.

- Wait: represents the state in which the network waits for next event to occur.

- Demand Arrival Handler: this state is reached when the event type is DemandArrival. In this state the Network Agent compute routes for the arrived demand. The simulation returns to the wait state when this state completed.

- Demand Departure Handler: this state is reached when the event type is DemandDeparture. In this state, the Network Agent tears down the path(s) of the departing demand. The simulation returns to the wait state when this state is completed.

- Link Failure Handler: this state is reached when the event type is LinkFailure. In this state, the Network Agent initiates the repair process and restores demands affected by the failure. The simulation returns to the wait state when this state is completed.

- Link Repair Handler: this state is reached when the event type is LinkRepair. In this state, the Network Agent performs several tasks including reverting demands to their normal working route, and examining the impact of this repaired link on blocked demands. The simulation returns to the wait state when this state is completed.

- Output measurements: this state is reached when the network simulation clock (simulation time) reaches a preset finish time. When this time is reached the simulation is stopped and various measurements are outputted.

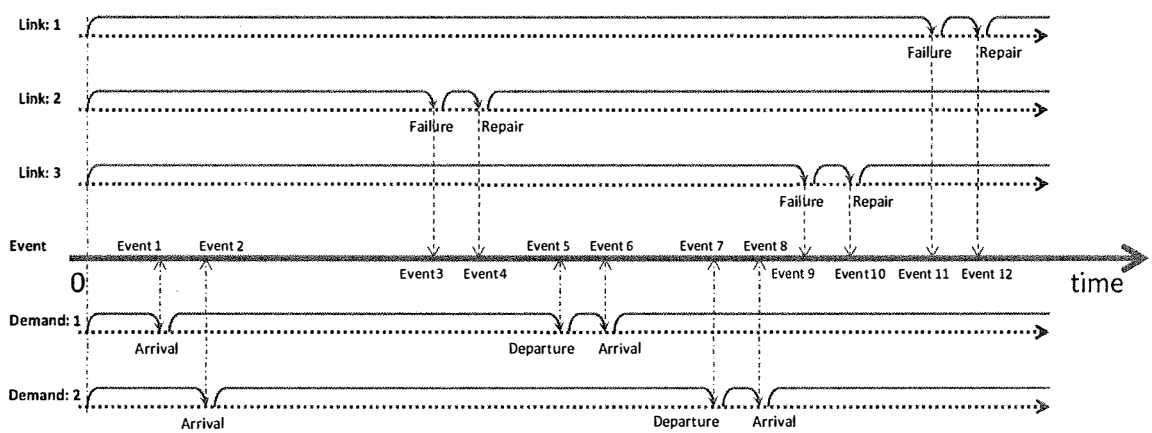


Figure 4-5 Link and demand event generation in the timeline

Figure 4-5 displays the timeline of the network simulation. In this example we assume there are 3 links and 2 demands. At the very beginning, two demand arrival events (Event 1, Event 2) are added to the simulation timeline. The arrival times are determined by the output of a

random number generator with an exponential distribution. Similarly, one link failure event for each of the links (3 links in total) is added to the simulation timeline. The time of link failure (Event 3, Event 9 and Event 11) is determined by a random number generator with an exponential distribution whose mean is the mean failure time for that link.

Then, the main simulation loop is entered. The loop iterates through events in the event timeline buffer in chronological order. For every demand arrival encountered, a new demand departure event (Event 5 and Event 7) is added to the timeline. For every demand departure encountered, a new demand arrival is added (Event 6 and Event 8).

For every link failure event (Event 3, Event 9 and Event 11), a link repair event is added (Event 4, Event 10 and Event 12). And for every link repair encountered, a link failure is added. In this way, at any given time in the simulation, there is a fixed number of pending link events and a fixed number of pending demand events. The simulation loop is terminated when a predetermined maximum time has expired.

4.4.3 The changes of demand

The entire network's behaviour is modeled by the simulation. Each demand behaviour affects the performance of the test algorithm as detailed below.

There are four events that may affect the demand condition. These four events are: DemandArrival, DemandDeparture, LinkFailure and LinkRepair. There are five different conditions for the demand. These five conditions are: Start, Fail, Work, Restored and Restored but fail.

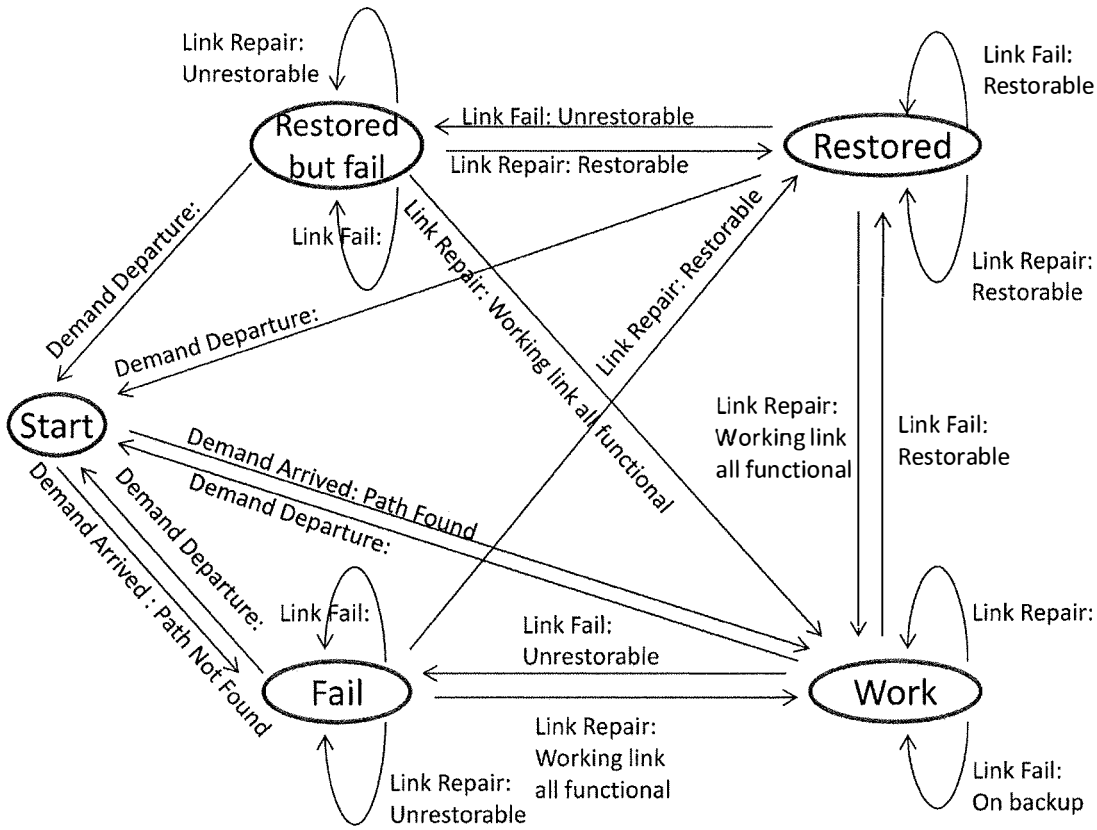


Figure 4-6 Events driven demand change

Figure 4-6 shows the demand changes driven by the above mentioned events. For easier demonstration of those conditions, we assume there is a demand with one working path and one backup path. The figure consists of the following demand conditions:

- Start: It shows the starting point of the demand.

- Fail: It shows the condition where a demand is not working at all. It may be because the working and backup path cannot be found at the initial stage, or because the working path has failed and the backup path is not available.

- Work: It shows the condition where a demand works with its working path.

- Restored: It shows the condition where a demand works with its backup path.

- Restored but fail: It shows the condition where a demand's working path had failed, it had switched to the restore condition and worked with its backup path (in the "Restored" condition), and then the backup path also failed.

4.4.4 Some important simulation statistics

In order to generate useful simulation results, a set of statistics will be set up.

- All the links in the simulation are bidirectional. The capacity of these links is 10 units of bandwidth.
- The bandwidth requested by a demand is 1 unit of bandwidth.
- All the links in the simulation are independent. Multiple link failures can happen at the same time in the network. The failure rate of all links is 500 FITs/Mile. This value is based on reference [25]. This implies 500 failures per 10^9 seconds per mile.
- In order to be compatible with previous research, 10 hours are set to fix link failure. The simulation simulates events over the span of one year.
- The number of demands in the simulation is fixed. Each survivability algorithm is simulated 13 times for each network with 20 ~ 150 demands.
- The demands are between a random source and destination. In order to let the demand stay active for most of the time, the average demand holding time is 10 days. The average interval between demand arrivals is 99 seconds.

4.5 Simulation Results

Three protection and restoration schemes are tested in this section: "Two-Path Simple Pool Sharing (SPS)", "1+1 Dedicated Path Protection (1+1 DPP)" and "Two-Path YinYang restoration (YinYang)".

Four metrics were tested in the simulations: Demand Blocking Ratio, Path Resource Consumption, Restoration Failure Ratio, and Availability.

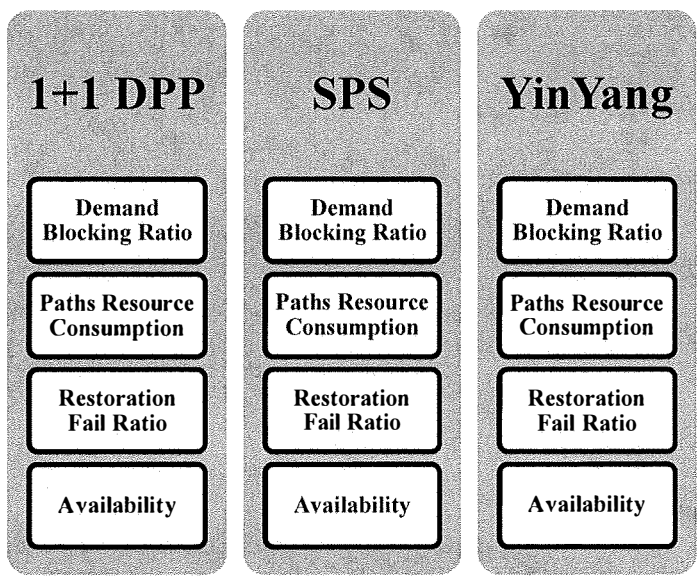


Figure 4-7 Simulation Algorithms and Metrics

Simulation algorithms and metrics were shown in Figure 4-7 Simulation Algorithms and Metrics. Simulations were conducted on the Global Crossing Network (GCN) and Modified Global Crossing Network (MGCN). The Modified Global Crossing Network (MGCN) has a

higher maximum degree than the Global Crossing Network (GCN). The number of demands in the simulation was varied from 20 to 150 in increments of 10.

4.5.1 Demand Blocking Ratio

Demand blocking ratio indicates the probability of not finding a path for a demand in the network.

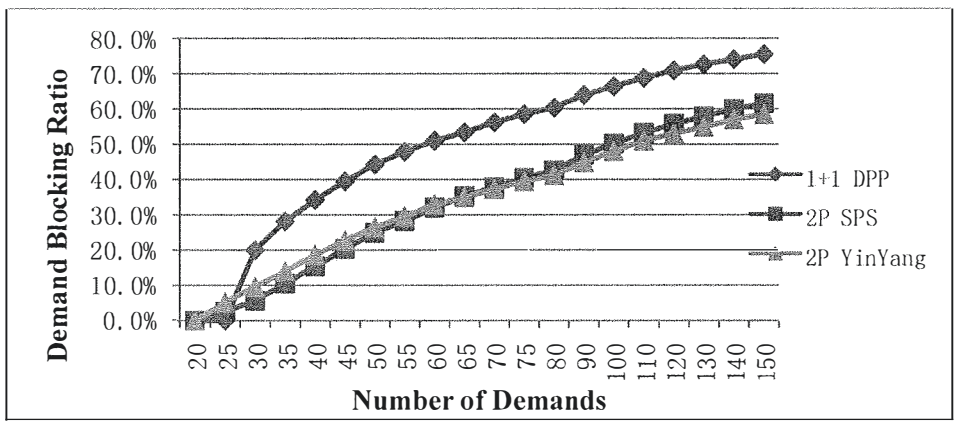


Figure 4-8 Demand blocking ratio for GCN network

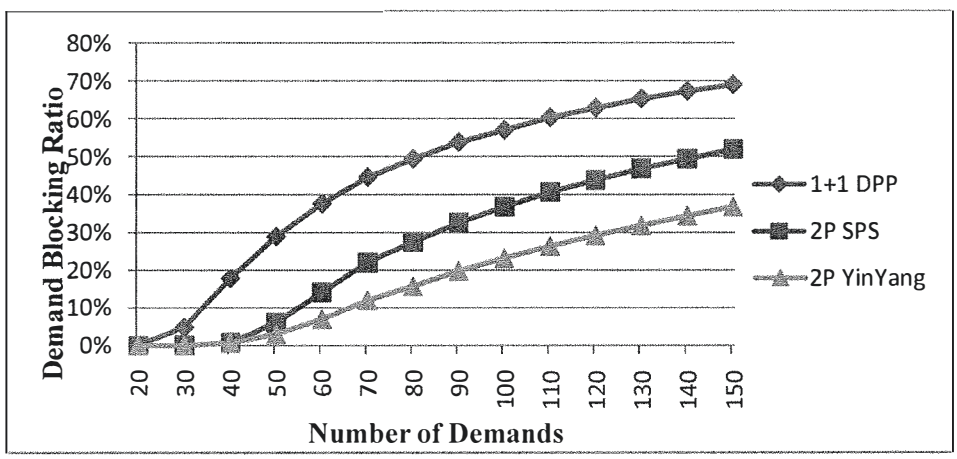


Figure 4-9 Demand blocking ratio for MGCN network

In Figure 4-8 (the lower-degree network), the two-path YinYang algorithm only slightly outperforms the simple pool sharing when the network is heavily loaded.

In Figure 4-9 (the higher-degree network), the two-path YinYang algorithm significantly outperforms the simple pool sharing when more than 50 demands arrive at the network.

In Figure 4-8 and Figure 4-9, 1+1 dedicated path protection has the highest demand blocking ratio, the reason will be explained in section 4.5.2.

4.5.2 Working Path and Backup Path Resource Consumption

This metric indicates the efficiency of an algorithm by measuring the resources i.e. (bandwidth) it consumed. The number of hops needed on the working path and backup path are measured. For shared capacity, only the first occurrence of resource consumption will be counted toward this metric.

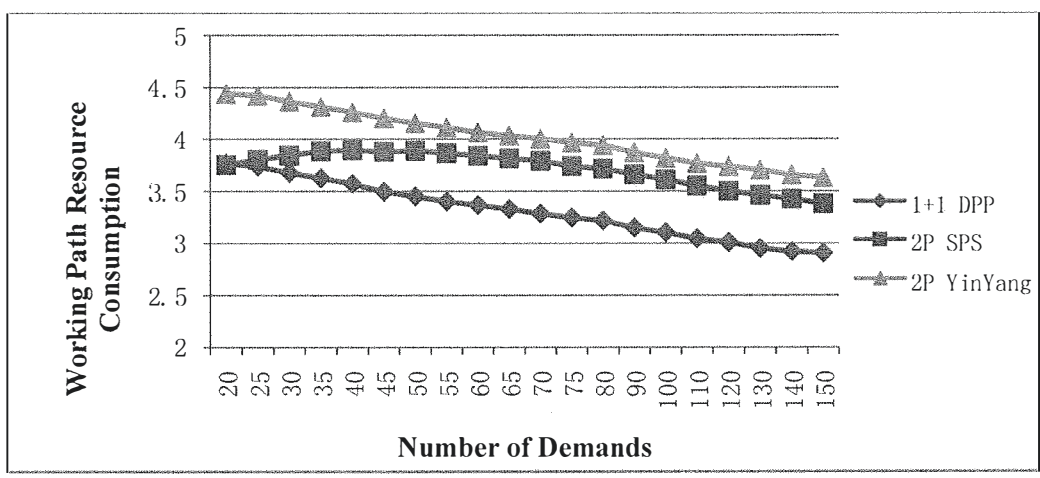


Figure 4-10 Working path resource consumption for GCN network

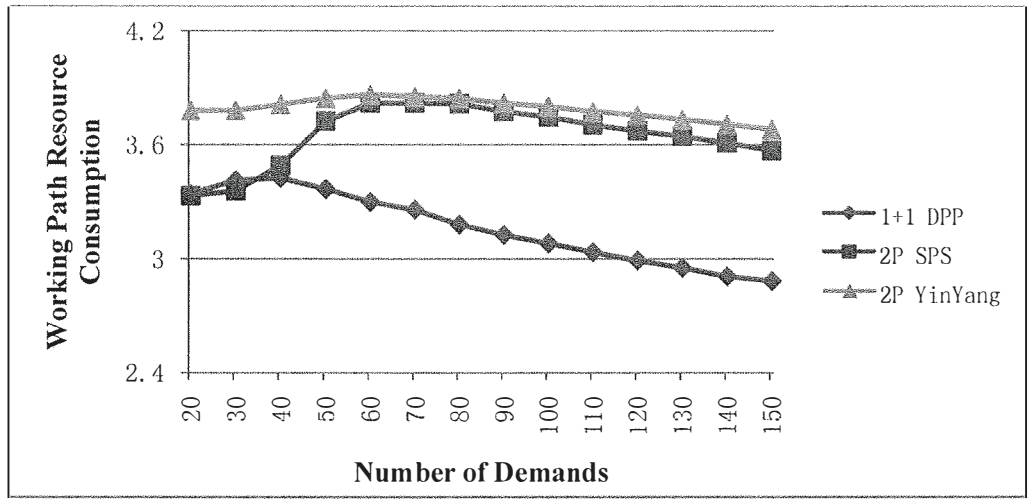


Figure 4-11 Working path resource consumption for MGCN network

In Figure 4-10 (lower degree network) 1+1 DPP algorithm has the lowest working path resource consumption. The YinYang algorithm has the highest working path resource consumption. The working path resource consumption of the SPS is between them.

In both of Figure 4-10 and Figure 4-11 the working path resource consumption of the SPS is close to the DPP at beginning. When more demands arrive at the network, the working path resource consumption of the SPS increased and approaches that of the YinYang algorithm.

The working path resource consumption for the SPS goes up when more demands arrive in the network. Both the SPS and 1+1 DPP find the first shortest path as the working path, so the resource consumption for the working path should stay the same, but it does not.

The explanation lies in Figure 4-8 and Figure 4-9 when the "demand blocking ratio" is investigated. In 1+1 DPP, when more demands arrived in the network, early arrived demands already take the resources (that will cost them the least), and this resource is dedicated so it

cannot be shared by later demands. Later arrived demands that cannot find resources will simply be rejected. In Figure 4-8 and Figure 4-9, 1+1 DPP gets the highest demand blocking ratio. The SPS algorithm uses fewer network resources and accepts more demands since the backup bandwidth is shared between different demands. The working paths of later arrival demands in SPS may be longer than those of the earlier arrival demands because there are fewer unused network resources available for the later arrival demands.

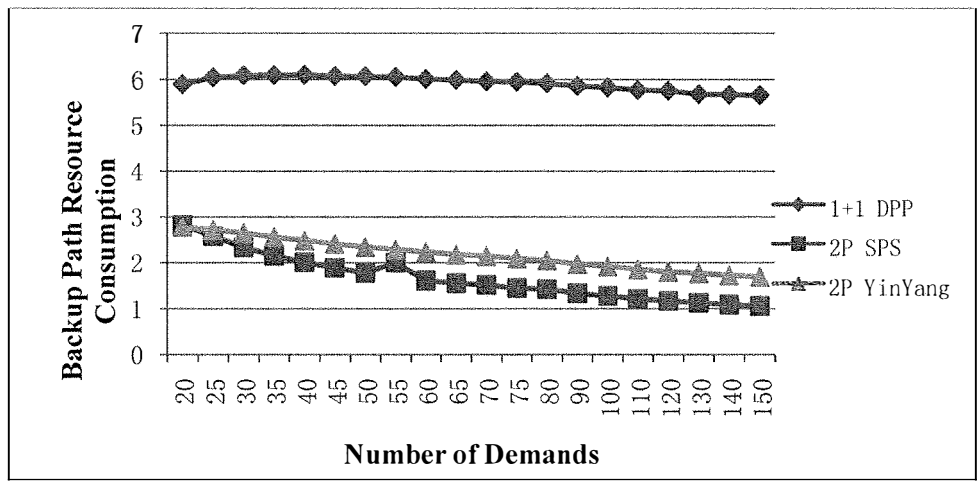


Figure 4-12 Backup path resource consumption for GCN network

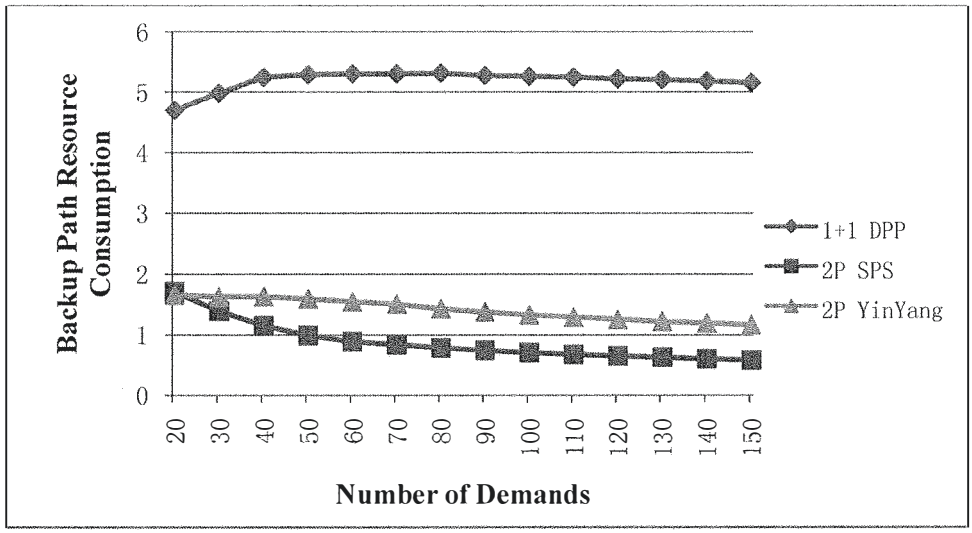


Figure 4-13 Backup path resource consumption for MGCN network

Backup path resource consumption shows the ability for the tested algorithm to share its backup path.

In both Figure 4-12 and Figure 4-13, 1+1 DPP has the highest Backup path resource consumption (BPRC), because 1+1 DPP does not share its backup path at all. The BPRC performance for two-path YinYing and the Simple pool sharing are close. Both of them apply sharing technology in their routing method.

The BPRC of the two-path YinYing algorithm is more than that of the SPS, because the YinYing algorithm computes two paths with working and backup bandwidth at the same time. The cost for computing the working portion also needs to be considered. Maximizing the sharing for the backup portion is not the only consideration with the YinYang algorithm.

4.5.3 Restoration Failure Ratio

Restoration failure ratio indicates the ratio of demands that cannot be restored when a failure occurs. In the protection and restoration scheme, restoration failure rate is closely related to the availability. The restoration failure ratio indicates the ability for an algorithm to recover from failure.

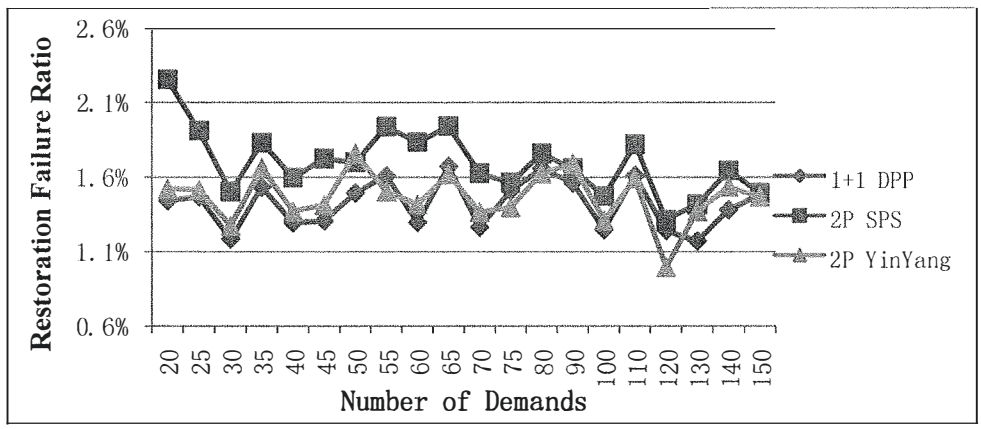


Figure 4-14 Restoration failure rate for GCN network

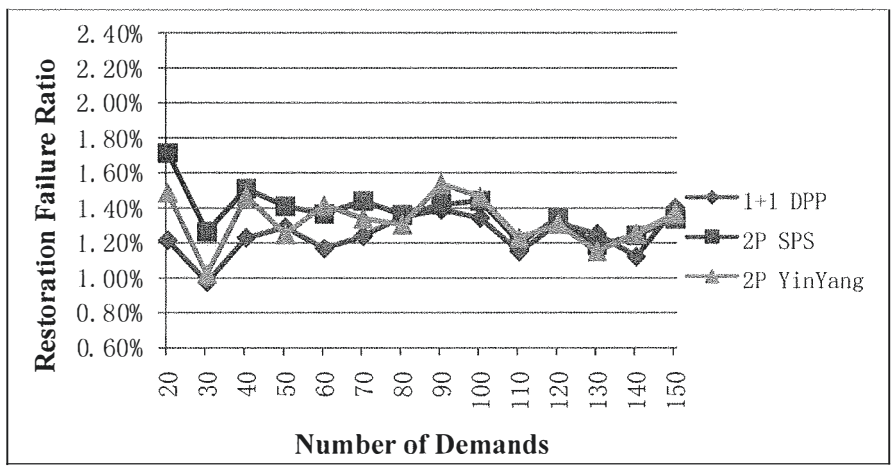


Figure 4-15 Restoration failure rate for MGCN network

Looking at Figure 4-14 (lower degree network) we can the YinYang algorithm has a similar performance as that of DPP but better than the performance of SPS.

In SPS, the backup resource is shared among different demands. In the case that multiple demands are affected by working path link failures, a multitude of these demands may rush to the same backup resource. This backup resource is not designed to serve all of these failures.

In DPP, dedicated backup bandwidths are allocated for different demands. There is no sharing between the backup bandwidths, so when multiple demands are affected by link failure, DPP performs better at restoring demands than SPS.

The Restoration failure ratio is directly proportional to sharing ability of the algorithm. For YinYang algorithm there is half working bandwidth and half shared backup bandwidth on both paths, so the YinYang algorithm's sharing ability is between that of SPS and DPP. YinYang algorithm's restoration failure ratio and the ability to restore from link failure is between SPS and DPP.

4.5.4 Availability

Availability shows the ability for an algorithm to withstand failures. The YinYang algorithm, dedicated path protection and the simple pool sharing can always recover from single link failure. In this simulation model, multi-link failure is simulated. Test algorithms were exposed to a higher failure rate than they can handle so that their robustness can be tested.

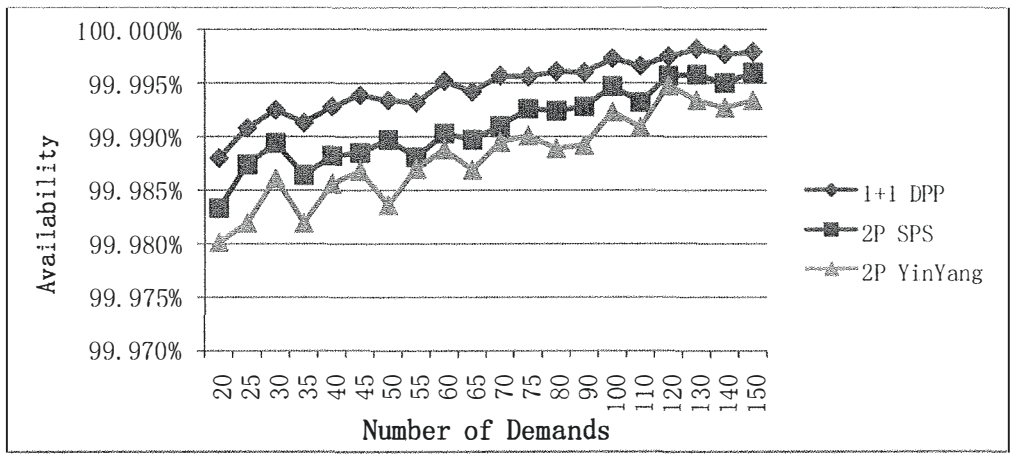


Figure 4-16 Availability for GCN network

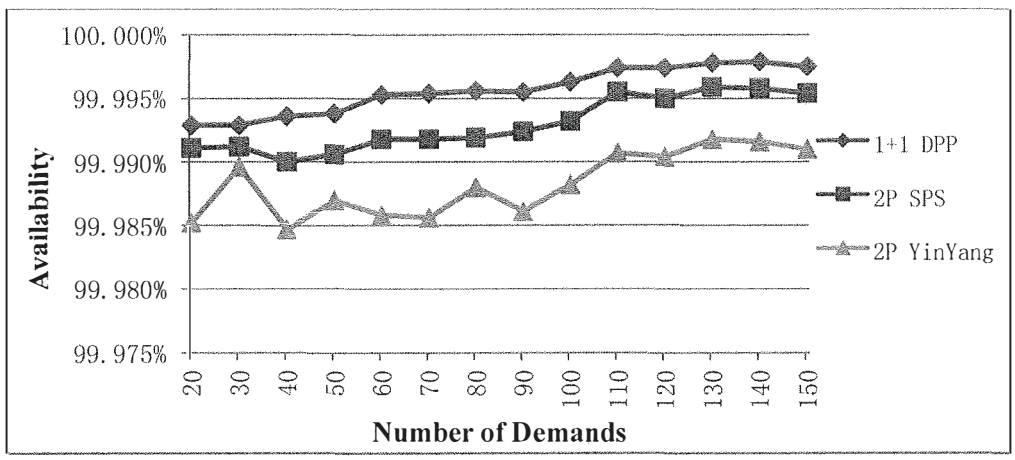


Figure 4-17 Availability for MGCN network

The result indicates that dedicated path protection has the highest availability. Next is the simple pool sharing and then Two-path YinYang algorithm.

DPP has the highest availability because it uses a dedicated backup path. The SPS has the lower availability because it uses a shared backup path between different demands. The YinYang algorithm has lowest availability because there is sharing on both of its paths, and

the hop number of these paths is larger than DPP and SPS (according to 4.5.2). Longer paths accumulate risks and cause lower availability.

4.6 Conclusion

By comparing the YinYang algorithm with two benchmark algorithms (The Simple Pool Sharing and 1+1 Dedicated Path Protection), we can make the following conclusions:

- The two-path YinYang algorithm can get lower demand blocking ratio compare with benchmark algorithm has been verified in the simulation.
- It was discovered that the two-path YinYang algorithm's demand blocking ratio will get lower in high-degree networks than in a low-degree network.
- The simulation accurately illustrates the Two-Path YinYang restoration's trade-off between reliability, bandwidth sharing efficiencies and demand blocking ratio.

Chapter 5

Conclusion and Future work

5.1 Contributions

The YinYang algorithm is the first algorithm that breaks the boundary between working path and backup path to route them together. The Two-path YinYang algorithm's routing method is the only routing method that can find two paths with sharing capacities at the same time. Compared with 1+1 dedicated protection and the simple pool sharing, the YinYang algorithm has the lowest demand blocking ratio, when the load is high and the degree of the network is high.

The YinYang algorithm uses the same mechanism as the Simple Pool Sharing to encourage reusing shared capacity in the network. It splits the working traffic between given source and destination nodes into multiple paths. This fundamental characteristic of the YinYang algorithm yields a more even bandwidth distribution amongst network links than the benchmark algorithms. This advantage leads to much lower congestion.

5.2 Future Work

The routing stage of the YinYang algorithm takes 50 iterations. Methods to decrease the routing iterations should be developed and then verified by simulation.

The three-path YinYang algorithm needs an even higher-degree network to test its performance as well as a new routing algorithm which can be the focus of future research.

References

1. Grover, W.D, *Mesh-based survivable networks: options and strategies for optical, MPLS, SONET, and ATM Networking*. Prentice Hall, 2004.
2. Tomsu, P. and C. Schmutzer, *Next Generation Optical Networks: The Convergence of IP Intelligence and Optical Technologies*. Vol. 2.Prentice Hall, 2002.
3. Drake, J. "*A review of the four major SONET/SDH rings*," in *IEEE International Conference on Communications (ICC)*, Vol. 2, pp. 878, May 1993.
4. Nonkomo, M.W. and R. Sewsunker. "*Optical backbone topology choice*," in *IEEE AFRICOM*, Vol. 1, pp. 353, Sept. 2004.
5. Ramamurthy, S., L. Sahasrabuddhe, and B. Mukherjee, "*Survivable WDM mesh networks,".*In *Journal of Lightwave Technology*, Issue:21, Vol:4, pp. 870. 2003.
6. Maach, A., G.V. Bochman, and H. Mouftah, "*Congestion control and contention elimination in optical burst switching,"* In *Telecommunication Systems*, Issue:27, Vol:2, pp. 115-131, 2004.
7. Huang, S. and B. Mukherjee. "*Adaptive Reliable Multi-Path Provisioning in WDM Mesh Networks,"* in *IEEE International Conference on Communications (ICC)*, pp. 5300, May 2008.
8. Song, L. and B. Mukherjee, "*On the study of multiple backups and primary-backup link sharing for dynamic service provisioning in survivable WDM mesh networks,"* in *IEEE Journal, Selected Areas in Communications,* Issue:26, Vol:6, pp.84-91, 2008.
9. Zhou, L., M. Held, and U. Sennhauser, "*Connection availability analysis of shared backup path-protected mesh networks,"* in IEEE International Conference on Communications (ICC) *Journal of Lightwave Technology*, Issue:25, Vol:5, pp. 1111-1119, 2007.

10. Naser, H. and H. Mouftah, "*Enhanced pool sharing: a constraint-based routing algorithm for shared mesh restoration networks,*" in *Optical Society of America (OSA) Journal of Optical Networking*, Issue:3, Vol:5, pp.203-323, 2004.

11. Naser, H. and M. Gong. "*A Delay-Constrained Shared Mesh Restoration Scheme*," in *IEEE International Conference on Communications (ICC)*, pp. 2230, Aug 2007.

12. Naser, H. and M. Gong. "*Design of Shared Mesh Restoration Schemes with Traffic Load Balancing Constraint*," in *IEEE International Conference on Communications (ICC)*, pp. 5305, May 2008.

13. Naser, H. and M. Gong. "*Link-Disjoint Shortest-Delay Path-Pair Computation Algorithms for Shared Mesh Restoration Networks,*" in *IEEE Symposium on Computers and communications (ISCC)*, pp. 269, July 2007.

14. Huang, S, B. Mukherjee, and C. Martel. "*Survivable multipath provisioning with differential delay constraint in telecom mesh networks,*" in *IEEE/ACM Transactions on Networking,* Issue:99, pp. 1, Oct 2008.

15. Retvari, G. and G. Nemeth. "*On optimal multipath rate-adaptive routing,*" in *IEEE Symposium on Computers and communications (ISCC)*, pp. 605, June 2010.

16. Mauz, C. "*p-Cycle protection in wavelength routed networks,*" *in Proc.7th Working Conf. Opt. Network Design and Modelling (ONDM'03),* Feb 2003.

17. Grover, W.D. and D. Stamatelakis. "*Bridging the ring-mesh dichotomy with p-cycles*," *in Proc.IEEE/VDE DRCN,* Munich, Germany, Apr. 2002, pp. 2761-2765.

18. Eshoul, A.E. and H.T. Mouftah,"*Survivability approaches using p-cycles in WDM mesh networks under static traffic*," in *IEEE/ACM Transactions on Networking (TON)*, Issue:17, Vol.2, pp. 671-683. April 2009.

19. Jr, W.W.T, *Operations Research Techniques, Merrill"s Mathematics and Quantitative Methods Series*, Columbus Ohio: Charles E Merrill Books Inc, 1967.

20. Sebbah, S. and B. Jaumard. "*A resilient transparent optical network design with a pre-configured extended-tree scheme,*" in *IEEE International Conference on Communications (ICC)*, pp. 1-6, June 2008.

21. Zhang, J, et al. "*A new provisioning framework to provide availability-guaranteed service in WDM mesh networks,"* in *IEEE International Conference on Communications (ICC)*, Vol. 2, pp. 1484, May 2003.

22. Gao, Z. and H. Naser. "*End-to-end shared restoration algorithms in multi-domain mesh networks*," in *IEEE Symposium on Computers and communications (ISCC)*, pp. 411, July 2008.

23. Griffin, D, et al., *Interdomain routing through QoS-class planes [Quality-of-Service-Based Routing Algorithms for Heterogeneous Networks].* in *IEEE Communications Magazine*, Issue:2, Vol.45, pp. 88-95. Feb. 2007.

24. Naser, H. and H.T. Mouftah, *Modeling and simulation of mesh networks with path protection and restoration.* Performance Tools and Applications to Networked Systems, pp. 88-117, Germany Berlin: Springer, 2004.

25. Crawford, D, *Fiber optic cable dig-ups: causes and cures.* Network reliability: a report to the nation-Compendium of technical papers, National Engineering Consortium, 1993.