OPTIMUM PATHS-FINDING ALGORITHMS FOR α+1 PARTIAL BANDWIDTH PATH PROTECTION

By

GAN MING LEE

A thesis submitted to the Department of Computer Science,
Faculty of Information and Communication Technology,
Universiti Tunku Abdul Rahman,
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Computer Science
January 2013

ABSTRACT

OPTIMUM PATHS-FINDING ALGORITHMS FOR α+1 PARTIAL BANDWIDTH PATH PROTECTION

Gan Ming Lee

Path protection ensures continuity of network services against link failure by assigning a link-disjoint secondary path to protect the primary path. Existing path protection schemes are classified into dedicated- and shared-path protections. However, these schemes either require high redundancy or substantial response time to resolve link failure. The end-to-end partial bandwidth protection scheme, denoted as α+1 protection, offers an alternative where only critical real-time information of the primary path is duplicated over the secondary path. The parameter α is defined as the ratio of the protection bandwidth (of secondary path) to the full bandwidth (of primary path). The challenge of α+1 protection is to identify a pair of primary-secondary paths with lowest total cost comprising the optimal solution.

Furthermore, an interesting optimal solution, which can neither be identified by the existing link-disjoint paths-finding algorithms, is found to occur when $0<\alpha<1$ in some cases. The occurrence of such an optimal solution has not been discussed in the literature, and it is referred to as a mid-optimal (MO) solution. In this thesis, the optimality conditions of the solutions with respect to any given value of α are explored, and new-efficient optimum paths-

finding algorithms for the α+1 protection are devised. The results derived from this study are supported by 5 refereed publications: 1 journal paper and 4 conference proceedings, with another forthcoming journal manuscript pending submission.

ACKNOWLEDGEMENTS

To God be the glory for the things HE has done…..

I would like to express my thanks and gratitude to my family who have given me their unconditional support and love, and for constantly remembering me in prayer. May God bless this family.

To my supervisors, Dr Liew Soung Yue and Dr Ng Yen Kaow. Many thanks for your guidance, wisdom, time and effort to supervise me towards the completion of this study. It is a privilege to have the opportunity to pursue this study under your supervision. Thank you once again.

To my friends and colleagues, thank you for your company, care, concern, and for just being there. I am grateful to have friends like you.

Trust in the LORD with all your heart and lean not on your own understanding; in all your ways submit to HIM, and HE will make your paths straight. (Proverbs 3:5-6)

APPROVAL SHEET

This thesis entitled "**OPTIMUM PATHS-FINDING ALGORITHMS FOR α+1 PARTIAL BANDWIDTH PATH PROTECTION**" was prepared by GAN MING LEE and submitted as partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science at Universiti Tunku Abdul Rahman.

Approved by:

(Dr. LIEW SOUNG YUE)
Date: 2/1/13
Supervisor
Department of Computer and Communication Technology
Faculty of Information and Communication Technology
Universiti Tunku Abdul Rahman

(Dr. NG YEN KAOW)
Date: 2/1/2013
Co-supervisor
Department of Computer Science
Faculty of Information and Communication Technology
Universiti Tunku Abdul Rahman

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

UNIVERSITI TUNKU ABDUL RAHMAN

Date: 2/1/2013

SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS

It is hereby certified that ***GAN MING LEE*** (ID No: ***09 ACD 02329***) has completed this thesis entitled "*OPTIMUM PATHS-FINDING ALGORITHMS FOR α+1 PARTIAL BANDWIDTH PATH PROTECTION* " under the supervision of Dr. LIEW SOUNG YUE (Supervisor) from the Department of COMPUTER AND COMMUNICATION TECHNOLOGY, Faculty of INFORMATION AND COMMUNICATION TECHNOLOGY, and Dr. NG YEN KAOW (Co-Supervisor) from the Department of COMPUTER SCIENCE, Faculty of INFORMATION AND COMMUNICATION TECHNOLOGY.

I understand that University will upload softcopy of my thesis in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,

(*GAN MING LEE*)

DECLARATION

I hereby declare that the dissertation is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at UTAR or other institutions.

Name GAN MING LEE

Date 2/1/2013

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AO	Alpha(α) Optimum
LB	Link-disjoint Bhandari
LPR/SFP	Linear Programming Relaxation / Single Flow Relaxation
MO	Mid-Optimal
PPP	Partial Path Protection
RAO	Reversed Alpha(α) Optimum
RF	Remove-Find
SALC	Shortest-Available-Least-Congested
SB	Suurballe
SOPS	Sub-Optimal Path-Solver
TDC	Toggling Dual Cost

CHAPTER 1

INTRODUCTION

1.1 Survivable Networks and Their Applications

The emergence of next generation network technologies such as wavelength division multiplexing (WDM) (Cisco 2000; Mukherjee 2006; Charbonneau and Vokkarane 2011) have considerably accelerated the growth of data traffic in physical communications network by allowing vast bandwidth provision on a single optical fiber (Mukherjee 1992; Benhamiche et al., 2010). As such, the risk of connection disruptions due to fiber cuts or node failure could potentially cause huge amount of data loss severely affecting the quality of service as well as productivity (Luo et al., 2006; Song et al., 2007; Rak 2012). The issue of network survivability thus becomes a fundamental requirement for high capacity networks (Haider and Harris 2007). Survivability in communications networks is defined as the capability of providing services even in the presence of failures (Zhou et al., 2000).

Two aspects are generally focused in survivable networks, mainly fault tolerance and redundancy (Ellison et al., 1999; Sterbenz et al., 2010). Fault tolerance refers to the threshold of the network to accommodate the occurrence of faults without compromising the overall system performance.

Redundancy employs a backup system in the network which offers equivalent functionality that circumvents the fault.

Network survivability in terms of connectivity looks on the measures adopted to enhance the reliability of the network connection against physical links or node failures (Kerivin and Mahjoub 2005; Todd and Doucette 2011). The setup of alternate paths (redundancy) is a common technique employed to improve the connection reliability (fault tolerance).

Generally, the survivability mechanisms (Gerstel and Ramaswami 2000) at this level can be classified under two categories: protection (Ramamurthy and Mukherjee 1999; Song and Mukherjee 2009) and restoration (Ramamurthy and Mukherjee 2002; Li 2011). Protection is understood to be a proactive approach whereby the backup route along with spare capacity is reserved at the initial connection setup phase. Restoration on the other hand is a reactive approach in which only after the fault is detected, would the network initiate a connection recovery attempt utilizing the current available spare capacity.

Communication network is the primary backbone for various connection, communication and network applications. The Internet for example has become vital for many industries including banking and finance, logistics, and commerce (Li et al., 2008; Sterbenz et al., 2010). High volumes of transactions involving various currencies and amounts are made over the Internet every day across time zones (Galati 2000). The Internet is also used in

administrations, communications, obtaining products and services, information access, as well as in numerous other fields.

The telecommunication networks consist of transmission and switching systems (McGorman 2002) which provide telephony service over mobile and fixed lines. The telephone network is the fundamental medium for voice communication. The network has since evolved to support packet data traffic (Liao et al., 2011). The growth and dependence on telecommunication service has raised the issue of the network architecture's survivability. This has become a crucial aspect especially for vendors, service providers and government agencies (Liu et al, 2004).

Military communication networks are developed specifically for the demands of modern battlefield (Fu-Li et al., 2011; Shi et al., 2012). National defense and efficient warfare coordination depends on the ability to execute network centric warfare and operations through a robust military network (Sterbenz et al., 2010). The concept of network centric warfare is to ensure a robust military network to enhance information sharing, situation awareness, collaboration, synchronization, sustainability, speed of command and mission effectiveness (Department of Defense 2005).

In the utility industry, for instance power generation and distribution, smart grid communication network is fast becoming the next emerging advancement by integrating the applications of telecommunication and information technology onto the electrical power grid (Luan et al., 2010; Hunt

2012). This enables the monitoring, control and communication on various aspects of the power grid to customers and operators, thus providing an opportunity to respond in real-time towards the changes of the grid's condition (Aalamifar et al., 2012).

Healthcare is another field which is beginning to see a growing reliance in communication network (Ghassemi and Wunnava 2002; Graves et al., 2005). Communication network in healthcare such as e-health and telemedicine (Pinciroli et al., 2011) is a critical component for communicating to a data centre to retrieve patient information, personnel authentication, inter- and intra-hospital communication, aid in disaster recovery, emergency response and patient health monitoring. Since healthcare is itself a critical field, it is vital that a reliable communication network be used in support of their important responsibility.

It is obvious that these critical network infrastructures are not only essential but they also support the functions of other systems (Rinaldi et al., 2001). The failure to any of these critical networks could cause catastrophic consequences. Therefore the study of network survivability is crucial to build a resilient and robust communication network.

1.2 Project Motivation and Objectives

Protection schemes are measures implemented to ensure the overall network's survivability in the event of unforeseen incidents, for instance link

failures. A link failure, such as a fiber cut is the most common cause of connection failure (Long 2010; Wu et al., 2011) that could be due to natural disasters, accidents, human error or intentional sabotage. In a network, the main connection between the source and destination is known as the active path or primary path. The objective of protection schemes is thus to maintain the connectivity between the source and destination nodes even if a link failure occurs along the primary path. These protection schemes can, in general, be categorized into link protection, path protection and partial path protection. They each function to provide varying levels of reliability for the connectivity between the source and destination nodes.

Two important aspects are basically used to evaluate the feasibility of protection schemes, namely the resource utilization and recovery speed (Li et al., 2001; Guo et al., 2008). Resource utilization can be measured by the amount of redundant backup resource being reserved to protect a particular connection. Conversely, the recovery speed is affected by the time needed by the protection scheme to detect a fault and resume the transmission. In the existing protection schemes, there is a tradeoff between resource utilization and recovery speed. Some protection schemes allow several connections to share the same pool of backup resource thus resulting in higher resource utilization but at the expense of recovery speed and vice versa. The architectures of the following protection schemes are discussed in the context of resource utilization and recovery speed.

Link protection (Ramamurthy and Mukherjee 1999; Zhou et al., 2000; Mukherjee 2006) protects individual links that make up the primary path. If a link on the primary path fails, link protection reroutes all traffic over the failed link while the rest of the links remain on the path. The architecture of link protection enables fast protection-switching time (Mukherjee 2006), because the fault detection and recovery are managed by the end nodes of the failed link without the involvement of the source and destination nodes (Zhou et al., 2000). This results in a desirable recovery performance. However, since there is no way to predict which link will fail, all the links along the primary path have to be considered in order to realize this protection scheme. This becomes impractical to implement as the size of the network increases. Consequently, the resource utilization for link protection is less efficient compared to path protection (Mukherjee 2006).

Implementing path protection to a network between a source and destination involves a primary path and a secondary path (or a protection path), where the two paths must be link-disjointed. Suppose a link in the primary path is broken, the secondary path would ensure the continuity of service by assuming the responsibility of data transfer that was handled by the primary path.

Existing path protection schemes are generally classified under two major categories: dedicated path protection (e.g., 1+1 protection, 1:1 protection) and shared path protection (e.g., 1:N protection, M:N protection) (Ramamurthy and Mukherjee 1999; Zhou et al., 2000; Assi et al., 2003;

Haider and Harris 2007). In the 1+1 protection, the same information signals are transmitted simultaneously by the source through both primary and secondary paths to the destination. Since the destination receives a full set of duplicate data, fast connection recovery can be ensured in the case of link breakdown on the primary path (recovery time ≤ 50ms) (Koster et al., 2005; Haider and Harris 2007). However, it is less efficient in resource utilization as it requires 100 percent redundancy.

The 1:1 protection, on the other hand, is a special kind of dedicated path protection (Schupke and Prinz 2004; Saeedinia 2011). It initially transmits data only on the primary path, and the reservation of bandwidth on the secondary path is rather a soft one so that low-priority traffic can utilize the reserved bandwidth when the protection detail of the secondary path has not been activated (Haider and Harris 2007). Only when the primary path fails, the transmission of data is resumed by switching the connection to the secondary path. However, the 1:1 protection still reserves the same amount of bandwidth to be for protecting high-priority traffic, thus it does not have significant saving in protection bandwidth because of its dedicated nature. Moreover, in the 1:1 protection, only after a failure is detected on the primary path, the secondary path can then resume the transmission, which leads to a higher recovery time compared to the 1+1 protection (Haider and Harris 2007).

The 1:N protection allows a single secondary/protection path to be shared by N primary/active paths, thus resulting in a more efficient resource utilization compared to the 1+1 and 1:1 protections owing to such a shared

nature. The drawback however is the limited protection coverage. That is, once the reserved backup resource has been used in the event of a path failure, all other paths that rely on this backup resource will be unprotected. To resolve this problem, M secondary/protection paths are used to protect N primary/active paths in the M:N protection. Nevertheless, it still suffers from less tolerance for multi link failures compared to the dedicated path protection (Zhou et al., 2000). In addition, both the 1:N and M:N protections notice significant increase in the response time to detect and recover from link failure due to the similarities in architecture with the 1:1 protection.

Another protection scheme known as the partial path protection (PPP) (Wang et al., 2002; Xue et al., 2005) assigns an 'end-to-end' protection path for each link of the primary path. Since the protection paths in PPP function to only protect certain link(s) on the primary path, it makes this protection scheme more flexible to implement as the protection path needs not be completely link-disjoint with the primary path. It is shown in Wang et al., (2002) that PPP outperforms the path protection in terms of blocking probability. But, the disadvantage of the PPP is fairly obvious. Apart from the detection at a higher layer for the primary path failure, local information at the lower layers for the location of the failed link on the primary path must also be identified. Both pieces of information are needed by the network manager in order to activate the necessary protection path associated to the particular failed link, compromising further the recovery time.

In most cases, when the primary path fails, real-time network applications could still be functioning at a satisfactory level over the secondary backup path that transmits only mission-critical data (Roy and Mukherjee 2008; Huang et al., 2010a; Huang et al., 2010b). In other words, the primary path should be able to be protected by a secondary link-disjoint path whose protection bandwidth is less than the primary bandwidth. The concept is to provide sufficient protection to ensure continuity of service with lower backup bandwidth, and thus reducing the network cost while the recovery time is not compromised. Such a concept is labeled as the partial bandwidth path protection scheme (Fang et al., 2005) or the α+1 path protection (Gan and Liew 2011).

1.2.1 α+1 Path Protection

Similar to the 1+1 protection, a solution in the partial bandwidth path protection comprises a pair of link-disjoint primary and secondary paths between the source and destination nodes. Unlike the 1+1 protection, however, the bandwidth requirement by the secondary path to protect the primary path can be flexibly adjusted. This protection scheme is denoted as the α+1 protection, where α is defined as the ratio of the bandwidth required on the secondary path over that required on the primary, and $0 \leq \alpha \leq 1$. This scheme is in fact applicable in optical networks as well as connection oriented networks such as IP/MPLS and ATM.

The value for parameter α is user defined, and it essentially specifies the level of protection required by respective services. As the value α varies, it may result in a different optimal solution with minimum total cost in implementing the α+1 protection. For instance, when α is equal or very close to zero, it is obvious that the shortest path, serving as the primary, coupled with another link-disjoint shortest secondary path would be an optimal solution because no or only very little protection bandwidth is required. Such a solution can be identified by a link-disjoint paths-finding algorithm called the Remove Find (RF or two-step method) (Bhandari 1994). When α is equal or very close to one, on the other hand, the optimal solution can be identified by another algorithm called the Link-disjoint Bhandari (LB), which was proposed in Guo et al., (2003) for the 1+1 protection. Detail examples describing the Remove Find and Link-disjoint Bhandari algorithm are presented in Chapter 3.3.

An interesting optimal solution, referred to as a mid-optimal (MO) solution occur when $0<\alpha<1$ in some cases. Such unique optimal solutions could not be identified by the RF and LB paths-finding algorithms nor has it been discussed in the literature. Further discussions on the MO solution are presented in Chapter 3.5. The fundamental objective of this study is therefore centered on the occurrence of the MO solution and the research towards an improved and enhanced optimum paths-finding algorithm for the α+1 protection scheme.

1.3 Research Framework and Contributions

The study presented in this thesis has the following framework as shown in Figure 1.1. The general research area is categorized under network routing. The subject area is then focused towards routing protection schemes and the existing path protection schemes such as the 1+1, 1:1, 1:N, M:N and PPP are reviewed accordingly. The discussion continues with the partial bandwidth path protection or α+1 protection scheme as the ideal protection scheme due to its features which addresses the disadvantages of the other existing path protection schemes. The problem formulation with regards to the α+1 protection is subsequently introduced and the discussion looks into the approaches from various studies.

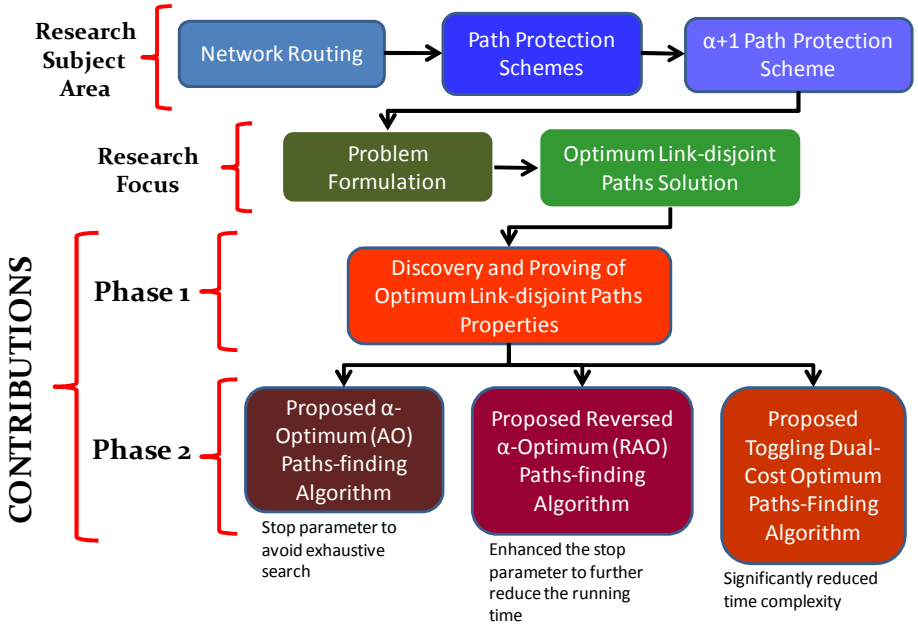

Figure 1.1: Project framework

The main contributions presented in this thesis are in two phases. In phase 1, the research is centered on the characteristics of the optimal solutions which comprised of link-disjoint paths. The properties inherent in the mid-optimal (MO) solution are particularly analyzed. The occurrence of such an optimal solution has not been discussed in the literature. In phase 2, the properties derived in phase 1 are incorporated to construct three novel optimum link-disjoint paths-finding algorithm for the α+1 protection schemes. The three proposed α+1 optimum paths-finding algorithms are labeled as the α-optimum (AO), reversed α-optimum (RAO) and the toggling dual cost (TDC) algorithm. The link-disjoint path-finding algorithms could also be realized as node-disjoint algorithms by utilizing the concept of 'node-splitting' (Guo et al., 2003; Suurballe and Tarjan 1984).

The AO algorithm identifies the optimal solutions utilizing the Yen's algorithm (*K*-shortest path algorithm). The *K*-shortest path algorithm functions to discover not just the shortest path in the network, but also the second shortest path, third shortest path and so forth. The Yen's algorithm (Yen 1971) is a well known *K*-shortest path algorithm and has a complexity of $O(KV(E+V \log V)$ (Martins and Pascoal 2003), where *K* is the number of paths to be found in ascending cost order. Among existing *K*-shortest path algorithms, the Yen's algorithm is selected because it is shown to have the lowest complexity (Hershberger et al., 2007a; Hershberger et al., 2007b).

Apart from that, the AO algorithm incorporates the derived properties of the MO solution which enables it to run efficiently without performing an

exhaustive search, as compared with other existing algorithms. The RAO algorithm suggests a slight modification to the original graph before executing the Yen's algorithm to identify the optimal solution. This slight modification allows the RAO to have a more effective stop parameter and reduces the procedures involved in each iterative cycle in comparison with the AO algorithm. The TDC algorithm functions with a modified version of the Bellman-Ford algorithm to find the optimal solution without the use of the *K*-shortest path algorithm. Generally, it is found that the largest contributor to the complexity of the α+1 optimum algorithms is due to the *K*-shortest path algorithm. As such, the TDC is shown to have a significantly lower complexity compared to the AO and RAO algorithms. A comparison between these new proposed α+1 optimum algorithm is reviewed in Chapter 5.4.

1.4 Thesis Overview

In this PhD thesis, findings will be presented to address the challenges mentioned with reference to the α+1 path protection along with the proposed approach to resolve the optimal solution. The results derived from this study are supported by 5 refereed publications: 1 journal paper and 4 conference proceedings, with another forthcoming journal manuscript pending submission.

The remaining chapters of the thesis are organized as follows. Chapter 2 discusses in detail the mechanism, advantages and applications of α+1 path protection (partial bandwidth path protection) including a review on the relevant literatures concerning this scheme. Chapter 3 looks into the problem

formation for the α+1 path protection solution and review studies with similar optimization function. The occurrence of the MO solution in α+1 path protection is subsequently highlighted together with the derivation of the MO solution properties. Chapter 4 presents two proposed optimum paths-finding algorithm (AO and RAO algorithm) for the α+1 path protection that is based on the utilization of common *K*-shortest path algorithm. Chapter 5 presents another optimum paths-finding algorithm (TDC algorithm) which applies a graph transformation process known as Suurballe graph transformation, instead of the *K*-shortest path algorithm to obtain the optimal solution with lower time complexity. Chapter 6 concludes the thesis.

CHAPTER 2

PARTIAL BANDWIDTH PATH PROTECTION (α+1 PROTECTION)

2.1 Mechanisms of α+1 Protection

The concept of α+1 protection is categorized as a path protection scheme. A connection between a source and destination node implemented with the path protection scheme will consist of a primary path which is protected by a secondary link-disjoint path. The secondary path ensures that connection between the source and destination node is maintained when the primary path fails. While the amount of bandwidth provided along the primary path enables the actual intended service operation, the concept of α+1 protection propose that the resource (bandwidth) allocation along the secondary path need not be the same as that of the primary path. It is found that many applications can provide a basic but nonetheless satisfactory service under minimum resource constraint (Santos et al., 2009; Vadrevu et al., 2012). Therefore, this basic form of service is made available through the secondary path in the event the actual comprehensive service could not be provided via the primary path. In view of this concept, the resource allocation along the secondary path can be significantly reduced compared to the resource allocated on the primary path.

The character α is used in this context corresponds to, in ratio, the amount of bandwidth allocated in the secondary path over that allocated in the primary path where $0 \leq \alpha \leq 1$. Changing the value α proportionally adjust the amount of bandwidth to be allocated in the secondary path to protect the primary path. When $\alpha=0$, it indicates that no bandwidth is allocated on the secondary path. This is a hypothetical case where it implies that the primary path is unprotected. When $\alpha=1$, the amount of bandwidth assigned on the secondary path is the same as that on the primary path. This protection configuration is now actually the same as the 1+1 path protection. It is apparent that a varying grade of protection can be offered in this manner. By adjusting the value of α, protection levels can range from zero protection ($\alpha=0$) to full protection ($\alpha=1$). This allows the flexibility for various applications to specify their desired levels of protection according to individual requirements and specifications.

2.2 Advantages of α+1 Protection over Existing Path Protection Schemes

Existing path protection schemes as mentioned in the previous chapter are the 1+1 protection, 1:1 protection, 1:N / M:N protection (Ramamurthy and Mukherjee 1999; Zhou et al., 2000; Schupke and Prinz 2004; Haider and Harris 2007) and partial path protection (PPP)(Wang et al., 2002; Xue et al., 2005). The differences and disadvantages of these existing schemes was discussed and subsequently the α+1 protection and its features are highlighted. One key feature of the α+1 protection is the fast recovery response time

similar to that of dedicated path protection. This positive attribute makes the α+1 protection an attractive scheme compared to the 1:1 protection, 1:N / M:N protection and PPP. Apart from that, the α+1 protection has the flexibility to offer a range of protection levels through specifying the value of α. This allows the α+1 protection to reduce the amount of redundancy required and consequently overcome the disadvantage faced by the 1+1 protection scheme which is notorious for having high redundancy.

From the network administrator's perspective, the α+1 protection enhances the network utilization. The efficient use of network resources by reducing redundancy for protection will improve the network's performance. This is reflected by the higher call admission rate (throughput) or lower blocking probability. The ensuing example illustrates this improvement by simulation over a graph topology.

A physical communications network is commonly represented in a topology comprising of nodes (which are either redistribution points or communication endpoints) interconnected by links (such as fiber optic cables). A graph is a practical mathematical representation of communications network (Laborczi 2002; Tizghadam and Leon-Garcia 2010). Nodes or vertices in the graph depicts the physical routers or switches while the graph edges are the cables or optical fiber links. A graph is expressed as $\boldsymbol{G} = (\boldsymbol{V},\boldsymbol{E})$, where $\boldsymbol{V}$ is a non-empty set consisting the elements of the graph vertices (Jungnickel 2008). The elements in set $\boldsymbol{E}$ are the graph edges. Two edges are considered to be

adjacent if they share a common node. Likewise, two nodes are adjacent when they are connected by the same edge.

A network is usually modeled as a directed graph (Laborczi 2002). Supposed set $\boldsymbol{V}$ consists of the nodes $\{a, b, c, d, e, f, g\}$ with the ordered edge set $\boldsymbol{E}$ $\{(a, b), (b, e), (c, d), (d, e), (e, f), (f, g)\}$. From this example, nodes *a* and *b* are adjacent nodes because they are connected with edge (a, b). Similarly, edge (a, b) and edge (b, e) are adjacent to each other because they have node *b* as the common node. The combination of sets $\boldsymbol{V}$ and $\boldsymbol{E}$ forms a graph (Harris et al., 2008). Additionally, the node degree indicates the number of edges incident on the node.

A path is defined as a route between a specified start/source nodes and end/destination node where the edges and nodes traversed from the source node to reach the destination node are distinct (Jungnickel 2008). A path can be expressed in a sequence of nodes. Referring to the previous example, the graph $\boldsymbol{G} = (\boldsymbol{V},\boldsymbol{E})$ consists of the nodes $\{a, b, c, d, e, f, g\} \in \boldsymbol{V}$ and edges $\{(a, b), (b, e), (c, d), (d, e), (e, f), (f, g)\} \in \boldsymbol{E}$. If the source and destination nodes are node *a* and node *f* respectively, the route from the source to destination node, denoted as path **P** may be expressed in this manner: $\mathbf{P} = (a, b, e, f)$.

The edges in the graph are often assigned with a weight value making the graph, a weighted graph. The weight value might indicate the distance, cost, time or delay, between two connecting nodes (Laborczi 2002; Jungnickel

2008). By attaching a weighted metric on the graph edges, it allows the computation to optimize the path selection based on specified criteria.

Correspondingly, the simulation in this example is conducted on a 14 node 21 bidirectional link NSFnet topology , as shown in Figure 2.1,with W = 16 unit bandwidth per link. The links, for this instance, are assumed to have equal weights where the weight represents the length of the link.

Each call is a connection request between two nodes on the network. Calls arrive in accordance to a Poisson process with rate λ. Source and destination node is randomly selected based on a uniform distribution. The call duration is exponentially distributed with a mean of $1/\mu$. Therefore the Erlang load offered to the entire network is $\rho=\lambda/\mu$. A successful call request would reserve 1 unit bandwidth on the primary path while on the secondary path the unit bandwidth reservation is proportional to α. If there is not enough capacity for either the primary or the secondary path's request, the call is blocked. For this example, the primary path is determined as the shortest-available path, while the secondary path is the shortest–available path which is link-disjoint with the primary path.

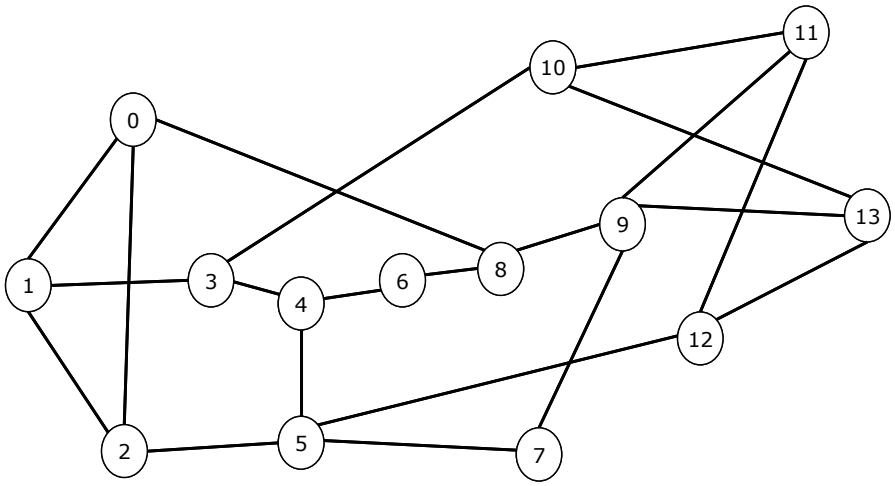

Figure 2.1: 14-node 21-bidirectional links NSFnet

Figure 2.2 shows the influence of α on the network's performance, measured in blocking probability. Each matching parameter shows the average blocking probability after 1 million calls. As α vary between 0.5 and 1, it respectively corresponds to 50% and 100% of protection bandwidth reserved along the secondary path with reference to the bandwidth utilized on the primary path. The graph in Figure 2.2 indicates that the blocking probability decreases when the protection ratio, α, is reduced. This trend is expected because by reducing the protection bandwidth, the network will generally have more spare capacity to accommodate a higher volume of calls. The results in Figure 2.2 verify that the implementation of the α+1 protection scheme does significantly improve the network's performance.

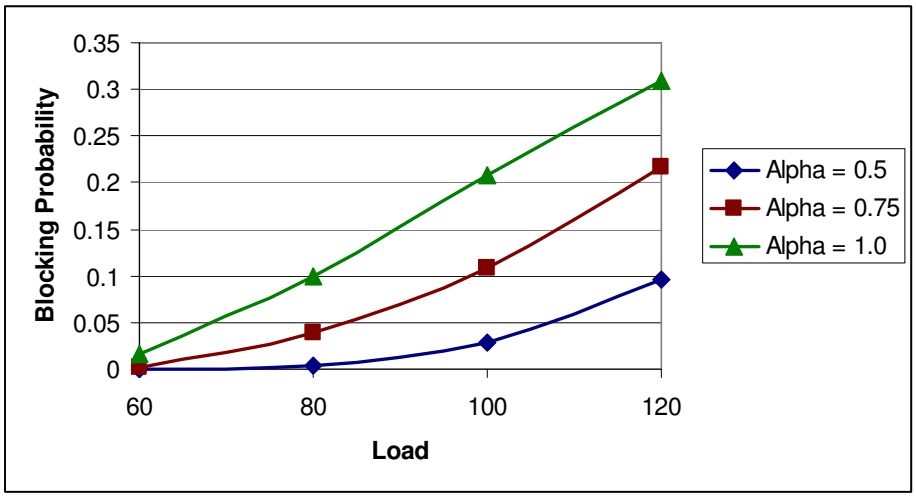

Figure 2.2: Effect of α on blocking probability

2.3 Applications for α+1 Protection

In communication networks, a connection-oriented network can be of either circuit switching or packet switching network (Veeraraghavan and Karol 1999; Veeraraghavan et al., 2006). A connection-oriented network basically requires a connection between two users to be set up by reserving network resources along the path before any actual exchange of communication occurs (Peros 2006). Examples of circuit switched networks are WDM and SONET/SDH networks, while connection-oriented packet switching networks are such as ATM, X.25 and MPLS. In circuit switching network, a channel is allocated for each connection. As for connection-oriented packet switching networks, communication is packaged in the form of packets. Network resources measured in bandwidth is reserved along the

connection path which accommodates the transmission of packets between both users.

The implementation of path protection complements the connection-oriented network by improving the transmission's path reliability. Path protection protects the transmission path by assigning a secondary link-disjoint path. Dedicated path protection will ensure a fast recovery from path failure. This feature is offered by the α+1 protection scheme. Additionally as mentioned, the α+1 protection provide the flexibility to specify the level of protection depending on respective specifications and applications. This allows the network resource to be efficiently utilized by reducing the amount of redundancy allocation.

Next generation network applications such as IPTV, VoIP, VoD, Triple Play (VoIP, video, data), IMS and P2P are fast dominating the data traffic in networks with a considerable growing number of end users (Needham and Harris 2008; Fiedler et al., 2009; Gumaste et al., 2010). Most of these applications function by streaming data in real-time (Gamage et al., 2007). The implementation of path protection for these applications would comprise a primary path, protected by a secondary link-disjoint path. To ensure minimum disruption in the event the primary path fails, the time needed for service to resume over the secondary path must be minimized. At present, the protection scheme with fast recovery time is the dedicated path protection (Haider and Harris 2007; Sukhni and Mouftah 2008). The

implementation of this scheme however, as mentioned, employs high redundancy.

In most cases, when the primary path fails, these applications could still function at a satisfactory level over the secondary path that transmits only mission-critical data. This is the rationale behind the α+1 protection scheme. It provides sufficient protection to ensure continuity of service with fast recovery time. For example, VoIP (Kim et al., 2010; Jelassi et al., 2012) involves real-time data streaming of audio feeds over IP networks for communication purposes. Suppose the α+1 protection scheme is employed in this application. A standard connection between two end-users involves a primary path and a secondary link-disjoint path. The primary path would transmit voice data of the highest quality possible while the secondary protection path transmits only the essential voice packets sufficient for satisfactory communication. In the event the primary path fails, continuity of service is assured at a reasonable level of quality.

From a network provider's point of view, services can be separated according to their protection level requirements. For example, critical services be provided with the 1+1 protection, best-effort service are not given any protection and intermediate service are offered 1:1 or shared backup protection. With the α+1 protection scheme, services can offer a more flexible range of protection level specified by α. The issue that arises is that the optimum routes may not be similar for services with different protection level. For example, the optimum routes (primary path and secondary path) for services with 1+1

protection may be different from the best effort services (no protection). Likewise, services with α level of protection might also have another optimum route. This matter would be discussed in detail in the proceeding chapters.

2.4 Existing Studies on Partial Bandwidth Path Protection

There have been substantial interests in the study of partial bandwidth path protection for the application in optical network communication. In Tracca et al., (2006), the concept of differentiated reliability (DiR) (Luo et al., 2009; Wu et al., 2010) is applied to the design of WDM networks together with the shared path protection (SPP) switching scheme. The scheme is referred to as SPP-DiR. In shared path protection, the resources along the backup paths are allowed to be shared on the condition that the primary paths do not have any common links. The proposed SPP-DiR scheme suggests that the backup resources for intersecting primary paths are still allowed to be shared provided that the computed maximum acceptable downtime ratio (MADR) is less than the tolerated value determined by the application. This enables reliability to be offered at different levels and as such reduces the necessary overall network resources while ensuring that each client application attains the necessary reliability as required and consequently resulting in cost reductions.

In another study, the quality of protection (QoP) framework is proposed (Ye et al., 2002; Fang et al., 2005) where partial protection is considered for a connection. The primary path is protected by a link-disjoint

backup path in which the bandwidth allocated along the backup path can be less than or equal to the bandwidth on the primary path. This allows the network to offer protection where each application can specify the minimum amount of bandwidth required to be reserved on the backup path. The proposed scheme is intended to reduce backup capacity allocation and as a result minimize the overall network blocking probability and connection cost.

The quality of protection framework proposed in Fang et al., (2005) is further enhanced in Sivakumar et al., (2007) and Gerstel and Sasaki (2010). As mentioned, the partial protection scheme in Fang et al., (2005) protects the primary path with a link-disjoint backup path where the reserved capacity on the backup path is less than or equal to the primary path. When the primary path fails, the connection is switched over to the backup path. Additionally, the mechanism described in Sivakumar et al., (2007) would attempt to maximize the bandwidth when the backup path is activated on top of its initially reserved capacity. This is to help minimize the effects caused by the bandwidth reduction when the connection switches to the protection path.

The partial protection model in Das et al., (2009) and Kuperman et al., (2011) suggest a slightly different perspective. A minimum fraction of the demand, denoted as q, is guaranteed between the source and destination, where q can range from 0 (no protection offered) to 1 (the service is fully protected). The scheme however views the demand and protection capacity as a single entity. Hence capacity is allowed to be distributed along multiple disjoint paths provided that the demand is satisfied, for service to be fully

functional, and the backup capacity is guaranteed when a single link failure occurs.

The multi-QoP scheme in Grover and Clouqueur (2005) presents a comprehensive policy that stipulate treatment of respective demands in a mesh-restorable network. The policy is divided into several classes which are: gold, silver, bronze and economy. Demands categorized under gold are offered full restoration protection. For those who are under the silver plan, connection restoration is of best effort depending on existing spare capacity with priority given to services that subscribe to the gold policy. Services that choose the bronze category are not offered any protection. Under the economy class, connections are also not protected and their primary resources may also be reassigned to satisfy the restoration requirement under gold. Studies with a similar concept are discussed in Giorgetti et al., (2005) and Kodian and Grover (2006).

Rak et al. (2009) considers differentiated levels of network survivability for double-node failure. Differentiated levels of network survivability are offered in the form of connection restoration time. Service demands are categorized under their respective service class based on the end-users priority requirements. As such, a lower priority demand would experience an increase in restoration time and vice versa. However, the solution produced may be only sub-optimal as it is obtained through a heuristic approach using an exhaustive trial search.

The following chapter gives the problem formulation to obtain the optimal pair of primary and secondary link-disjoint paths for the partial bandwidth path protection. A summary on existing research with similar problem formulation is presented together with discussions on the effectiveness of well-known link-disjoint path finding algorithm in identifying the optimal solution for the partial bandwidth path protection.

CHAPTER 3

ANALYTIC DISCUSSIONS ON α+1 PROTECTION

3.1 Problem Formulation

From the previous chapter, the partial bandwidth protection scheme referred to as the α+1 protection scheme has a varying parameter, α, where $0 \leq \alpha \leq 1$, depending on the application requirement. As the value of α varies from 0 to 1, it will affect the total cost of implementation. Without the loss of generality, we assume that the bandwidth requirement for the primary path is defined as 1, and that for the secondary is α. Given a valid solution that consists of a pair of link-disjoint paths for α+1 protection, let P denote the cost (per unit bandwidth) of the primary path, and S that of the secondary. The total cost can thus be defined as

$$C(\alpha) = P + \alpha S, \qquad (1)$$

where $C(\alpha)$ is the total cost of the solution, which is also a function of α. It should be noted that for any valid solution, the cost of the primary and secondary path are such that $P \leq S$. This is true when assuming the network has no capacity constraint.

The objective is therefore to find a link-disjoint primary and secondary path such that the total cost, $C(\alpha)$, is minimized. The challenge arise, as

highlighted in Gan and Liew (2010a), when the optimal solution comprising of a pair of link-disjoint path is incline to change as the value of α varies along its range.

3.2 Studies with Similar Problem Formulation

There have been several research articles that present a similar problem formulation as defined in eq. (1). The work in Laborczi et al., (2001) is described as asymmetrically weighted pair of disjoint paths. It focuses on finding and minimizing the cost function comprising a pair of node-disjoint primary and protection path between a given source and destination. This is also referred to as the node-disjoint diverse routing problem. The weight factor parameter, we denote as α_1, defines the relation between network resource consumption and utilization between the working path and the protection path. α_1 is introduced into the cost function in order that one of the link-disjoint path-pair is biased over the other with an average shorter length. The work is extended in Ho and Mouftah (2002) and Todimala and Ramamurthy (2006) to include the consideration of Shared Risk Link Groups (SLRG) when selecting the optimal solution.

In another study, the term is referred to as α-MIN-SUM 2-path problem (Yang et al., 2005a; Yang et al., 2005b). The weight factor, we denote as α_2, applies to reliable telecommunication network in which a connection between the source and destination node consist of the primary path and a link/node disjoint protection path. Here, α_2 is defined as a ratio that

is proportion to the length of the path which serve the same purpose as the parameter in Laborczi et al., (2001), Ho and Mouftah (2002) and Todimala and Ramamurthy (2006), for one link-disjoint path-pair to have an average shorter length over the other. Incorporating the weight factor is said to allow service providers to implement administrative controls such that encourages the utilization of certain routes to avoid network traffic congestion, realizing load balancing or increase the network throughput.

The shared path protection scheme in survivable traffic grooming for protection at the light path level (Yao and Ramamurthy 2005) establishes two light-paths where the primary light-path is protected by a backup light-path. The same concept is applicable for protection at the connection level (Yao and Ramamurthy 2004). The pair of primary and secondary backup path is shared risk link group (SRLG) diverse paths. The primary path is selected from the shortest available path. While the backup path has a different link weight function, weighted by the parameter, we denote as α_3. This weight factor is used, by assigning α_3 with a small value, $0 \leq \alpha_3 \leq 1$, to encourage the sharing of wavelengths with other backup paths provided that their respective primary paths do not share any common risk.

It is obvious from the discussions of the various literatures that the solution to the problem formulation is the identification of the pair of optimal link/node disjoint path that minimizes the cost function. Assuming that the nodes of the network are usually located in easily accessible areas where they can be serviced and maintain periodically to ensure working reliability, hence

it is more essential for the protection scheme to focus on the network links. For the next section, well-known existing link disjoint paths-finding algorithms are reviewed to ascertain their feasibility in identifying the optimal solution for the α+1 protection problem formulation.

3.3 Existing Link-disjoint Paths-Finding Algorithms

Well-known link-disjoint paths-finding algorithms; the remove-find (RF) method (also known as the two-step method) (Bhandari 1994), the Link-disjoint Bhandari (LB) (Guo et al., 2003), the Suurballe (Suurballe and Tarjan 1984) the Bhandari (Bhandari 1994) and *K*-disjoint path algorithms (Rak 2010; Leepila et al., 2011) are discussed in this section. These link-disjoint paths-finding algorithms are mentioned because the solutions produced, consisting of a pair of link-disjoint paths, are optimal but only at specific values of α, with regards to the α+1 protection. For example, assuming valid solutions can be found by the algorithms respectively, when α=0, the solution by the remove-find RF is optimal, whereas when α=1, the solutions by LB, Suurballe and Bhandari accordingly are optimal. Properties from these solutions are analyzed and subsequently integrated to the construction of the proposed optimum α+1 paths-finding algorithm.

3.3.1 Remove-Find (Two-Step) Method

With reference to the network given in Figure 3.1, suppose a reliable communication needs to be set up between the source node ‘a’ and destination

node ‘f’, where the value on each link represents the cost (per unit bandwidth) (Younis and Fahmy 2003) incurred for data to traverse the particular link. The remove-find (RF) method consists of two steps to find a link-disjoint pair of paths. From Figure 3.2, the first step is to determine the least-cost path from the source to destination. This can be accomplished by applying familiar shortest path algorithms such as the Dijkstra’s algorithm (Dijkstra 1959).

The Dijkstra’s algorithm (Dijkstra 1959) is a well-known algorithm that is applied to find the shortest/least-cost path in the graph between a given source and destination node. It is also used to find the shortest path tree, whereby the path cost from a specified root not to all other nodes in the graph is minimal. The Dijkstra’s algorithm has a complexity of $O(|E| + |V| \log |V|)$ with V being the number of nodes and E the number of edges (Barbehenn 1998). This makes it one of the most efficient shortest path-finding algorithms. The shortcoming of the Dijkstra’s algorithm is the inability to handle negative weight values.

After the shortest-path has been identified, it is assigned to be the primary path. To find a link-disjoint secondary path, the next step of the method requires the primary path to be removed from the network and then the shortest path algorithm is applied once again to find the least-cost path from the ‘trimmed’ network, as shown in Figure 3.3. Hence, the RF solution consists of the shortest path from “a” to “f” as the primary, and the shortest link-disjoint path as the secondary. Figure 3.4 shows the primary path (path: a-

b-c-d-e-f, cost = 1+1+1+1+1 = 5) and secondary link-disjoint path (path: a-f, cost = 12) as obtained by the RF algorithm.

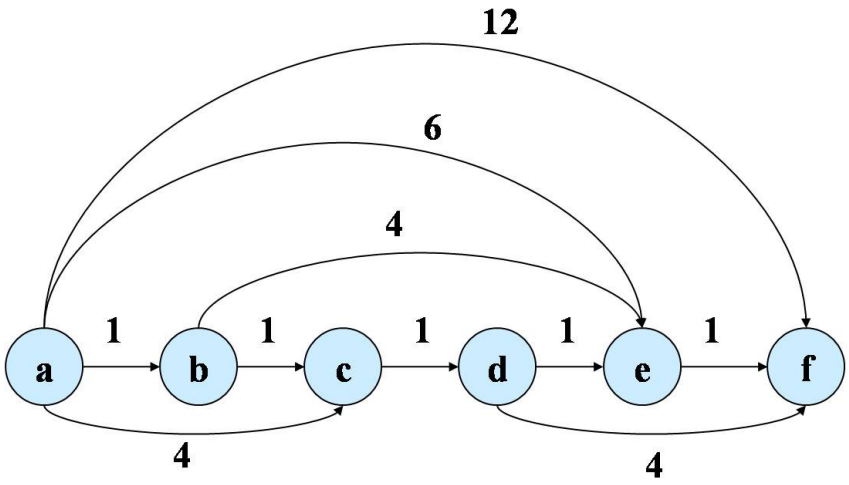

Figure 3.1: Network example

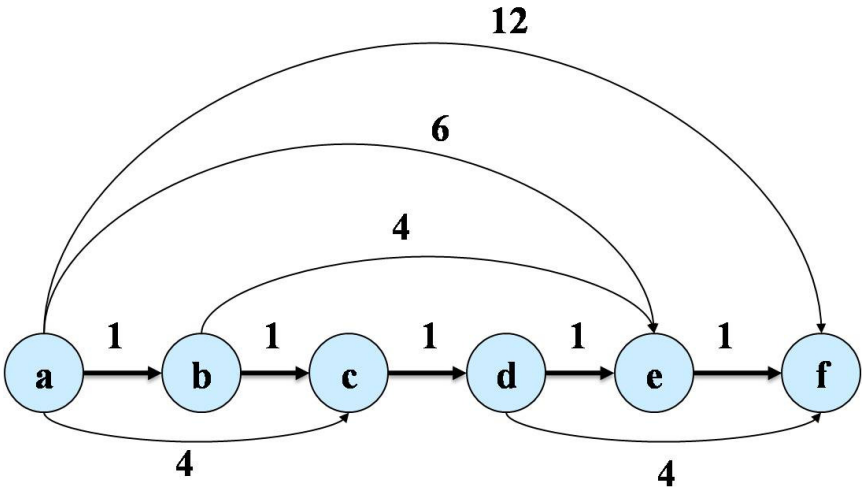

Figure 3.2: RF Step1: Determine the least cost path between node 'a' and node 'f'

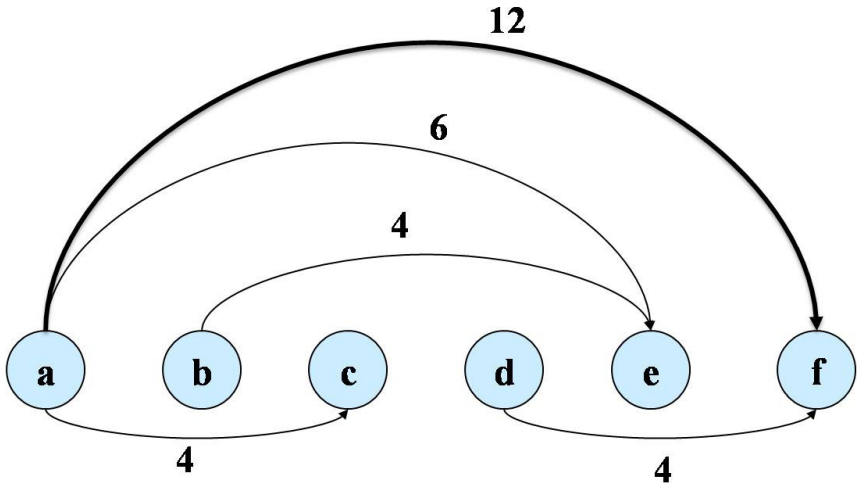

Figure 3.3: RF Step2: Remove the initial least cost path from the network and find the subsequent least cost path from the 'trimmed' network

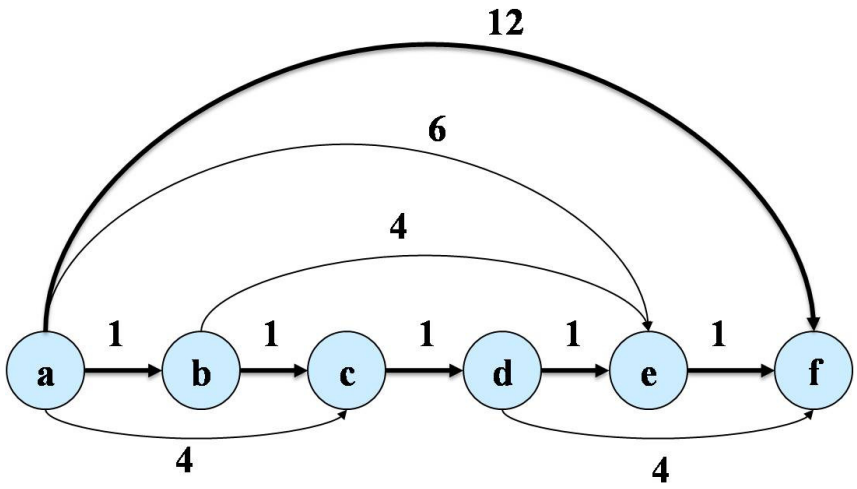

Figure 3.4: A pair of link-disjoint path by the RF method

Consider the first extreme value when $\alpha = 0$. This is a hypothetical case where the secondary backup path is not allocated any bandwidth to protect the primary path. In this scenario, the RF method is ideal as the primary path is the shortest path.

3.3.2 Bhandari Algorithm & Link-Disjoint Bhandari (LB)

The Bhandari algorithm (Bhandari 1994) identifies a pair of node-disjoint paths between a specified source and destination node in a given network. The later LB algorithm (Guo et al., 2003) is developed based on the Bhandari algorithm and shares several similarities. In addition, the Bhandari algorithm performs a unique procedure called 'node-splitting' to ensure the pair of path found is node-disjoint.

The LB solution on the other hand consists of a pair of link-disjoint paths where the sum of their costs is the minimum among all other possible pairs. The LB algorithm is summarized into four steps. The first step is to determine the least-cost path, denote as $\mathbf{P_1}$, from the source to destination. $\mathbf{P_1}$ is found by applying a shortest path algorithm. Figure 3.5 indicates path $\mathbf{P_1}$ in the example. The second step is to replace $\mathbf{P_1}$ with $\mathbf{-P_1}$. $\mathbf{-P_1}$ is obtained by reversing the direction of $\mathbf{P_1}$ links, and changing the associate cost of the $\mathbf{P_1}$ links with equivalent negative values. Figure 3.6 illustrates the modified network with $\mathbf{-P_1}$. The third step is to find the least-cost path in the modified network, denoted by $\mathbf{P_a}$. The Bellman Ford algorithm is used to find $\mathbf{P_a}$ in this stage as the network has links with negative values.

The Bellman-Ford algorithm (Bellman 1958) is another shortest/least-cost path-finding algorithm. The Bellman-Ford algorithm executes a repetitive relaxation process on all edges in the graph for $|V|$-1 cycles to propagate the minimum distance across the whole graph and at the end of the relaxation cycle, the shortest path is obtained. Unlike the Dijkstra's algorithm mentioned earlier, the Bellman-Ford algorithm is capable of identifying the shortest path problem even when there are negative weights present in the graph. The tradeoff to this is the increased complexity of the Bellman Ford at $O(|V||E|)$ (Subramani and Kovalchick 2005).

After $\mathbf{P_a}$ is obtained using the Bellman Ford algorithm, the $\mathbf{-P_1}$ link shared by $\mathbf{P_a}$ is 'virtually' removed. Figure 3.7 shows path $\mathbf{P_a}$ as well as indicates the link shared by $\mathbf{-P_1}$ and $\mathbf{P_a}$. The forth step involves restoring the rest of the reversed links together with their initial non-negative cost and grouping the remaining links into two paths, $\mathbf{P_1'}$ and $\mathbf{P_2'}$. Finally the link that was 'virtually' removed in the third step is restored to its initial state. Figure 3.8 shows such pair of paths (path: a-e-f and path: a-b-c-d-f) obtained by the LB algorithm, where the sum of costs is 14 (7+7), which is the minimum among all.

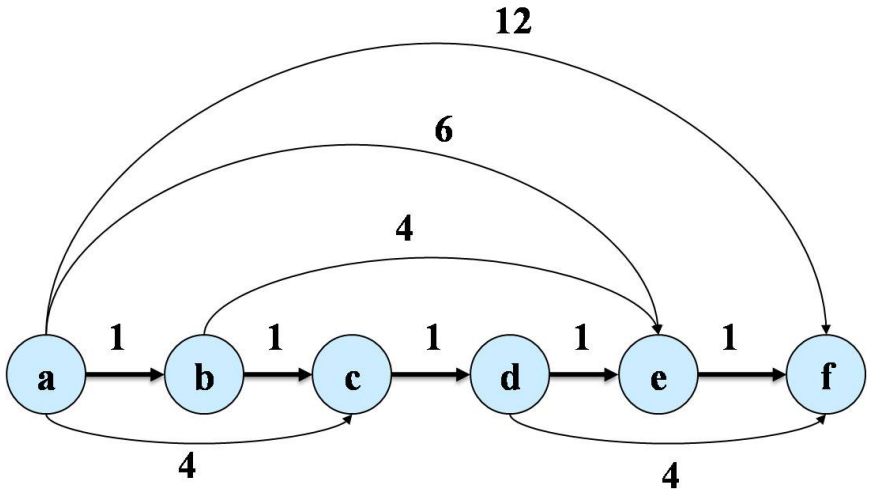

Figure 3.5: LB Step1: Determine the least cost path, P_1, between node 'a' and node 'f'

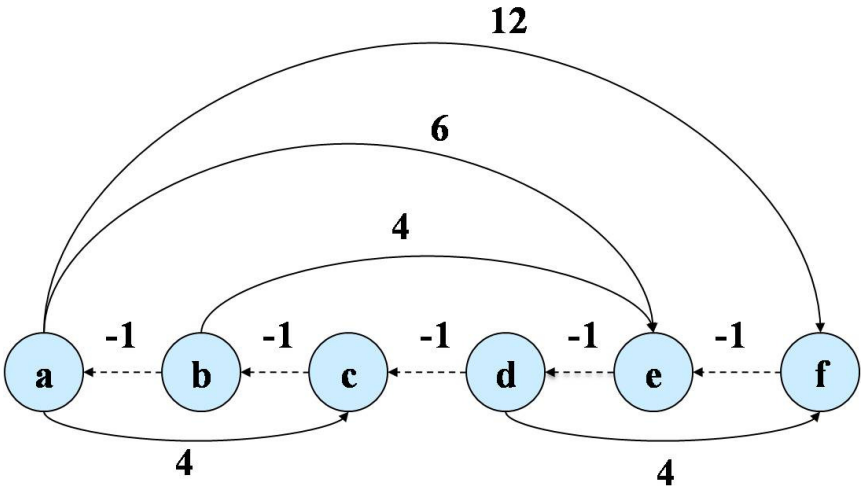

Figure 3.6: LB Step2: Replace P_1 with $-P_1$

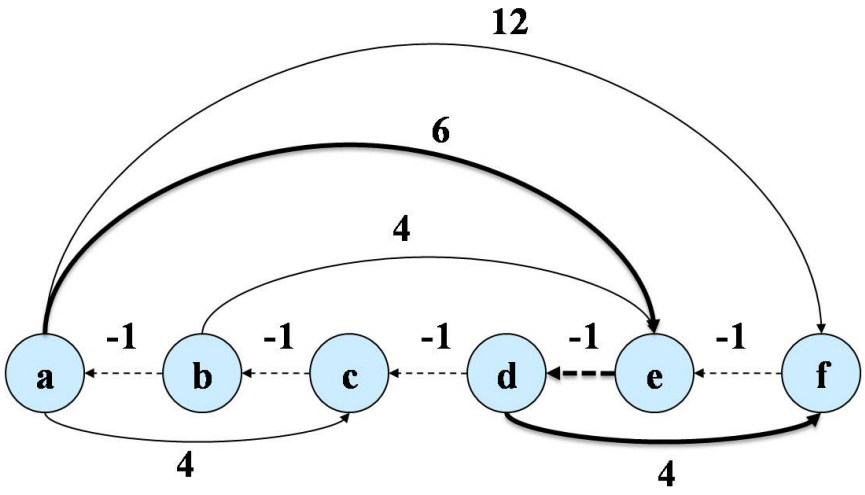

Figure 3.7: LB Step3: Find the subsequent least cost path, P_a, in the modified network and remove the $-P_1$ link that appear in P_a

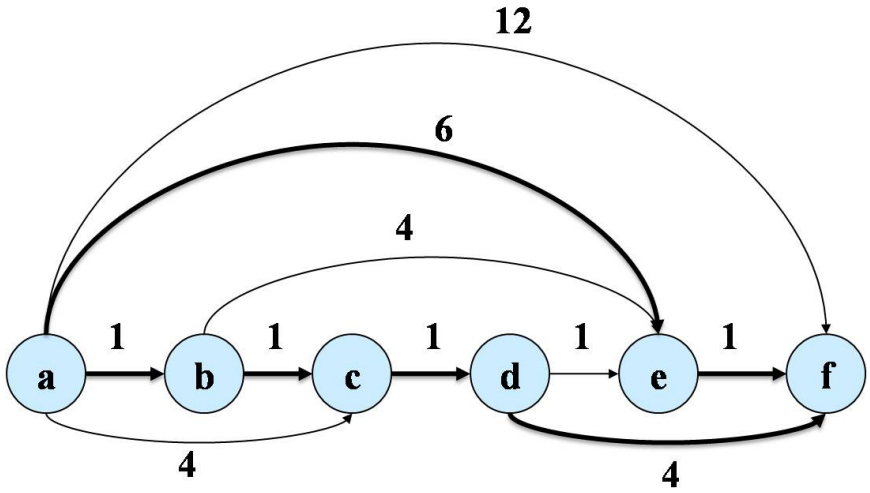

Figure 3.8: A pair of link-disjoint path by the LB algorithm

Consider another extreme value when $\alpha = 1$. This is the case where the secondary link-disjoint path protects the primary path with the full amount of bandwidth (1+1 protection). The LB algorithm is preferred as it ensures the optimality of its solution for such 1+1 protection scheme (Guo et al., 2003).

3.3.3 Suurballe Algorithm

The Suurballe algorithm (Suurballe and Tarjan 1984) is also designed to obtain the optimal pair of link disjoint path for the 1+1 protection similar to the solution produced by the LB algorithm. However, it is mentioned that the LB algorithm utilizes the Bellman Ford algorithm due to the presence of negative links after the graph is modified. The Suurballe algorithm on the other hand performs a graph transformation process which alters the link-cost to non-negative values, assuming the graph contains no negative cycles. This then allows the Suurballe algorithm to utilize the more efficient Dijkstra's algorithm as part of its process. The Suurballe algorithm has two major steps. Step 1 applies the graph transformation properties on the original graph. Step 2 finds the shortest path in the transformed graph. The shortest path in the transformed graph together with the shortest path in the original graph will form a link-disjoint path-pair in the original graph, which is also the optimal solution for the 1+1 protection scheme. The following describes the Suurballe algorithm procedures in detail.

From the network example of Figure 3.1, the shortest path in this graph is identified as a-b-c-d-e-f , with a total path cost of 5. This is also the

minimum spanning tree of the graph, and will be used to re-compute the link-cost in the Suurballe transformed graph.

In Figure 3.9, the Suurballe transformed graph has the link-cost of every edge (v, w), connecting a vertex v to a vertex w (where $v, w \in \boldsymbol{V}$) in the original graph recomputed as follows.

$$c'(v, w) = c(v, w) - d(s, w) + d(s, v) \qquad (2)$$

where $c(v, w)$ is the current link cost in the original graph, $d(s, w)$ the shortest distance from the source node, s, to node w, $d(s, v)$ the shortest distance from node s to node v, and $c'(v, w)$ is the updated link-cost in the Suurballe transformed graph. Note that the directions of all the links on the shortest path in the original graph are reversed in the Suurballe transformed graph with cost equal to zero.

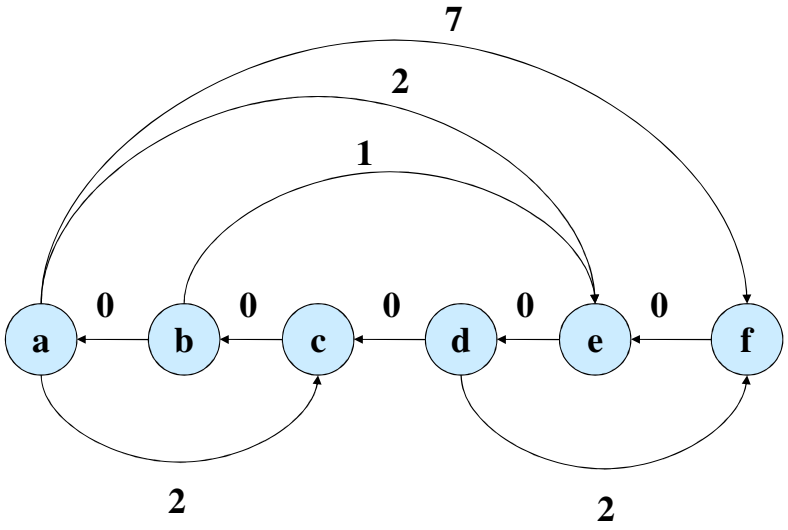

Figure 3.9: Suurballe transformed network

Subsequently, the shortest path found in the Suurballe transformed graph (eg. a-e-d-f in Figure 3.10) is coupled with the shortest path in the original graph (e.g., a-b-c-d-e-f in Figure 3.2); and by removing the common links in the reverse direction (e.g., e-d in Figure 3.10 and d-e in Figure 3.2) they can form a pair of link-disjoint paths which is the optimal solution in 1+1 protection shown in Figure 3.8.

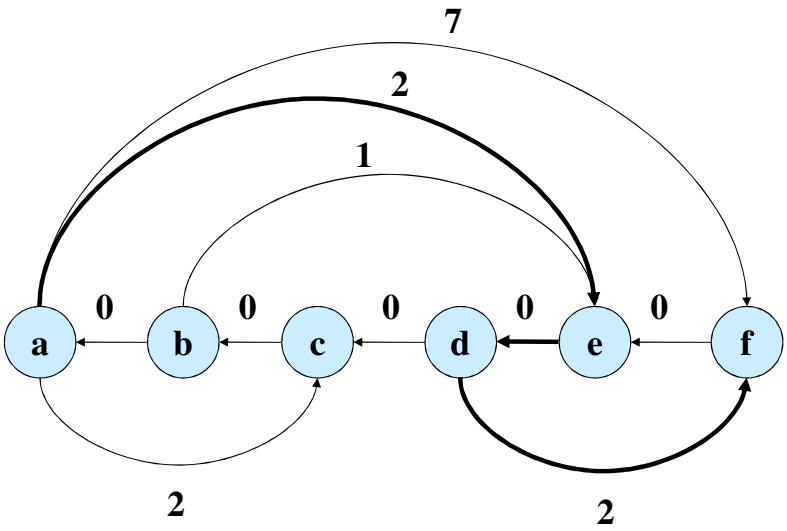

Figure 3.10: Shortest path in the Suurballe transformed network

3.3.4 Multi-Link/Node Protection Approach

Algorithms to identify *k*-disjoint paths for multi-cost networks (each network links could have different costs/weights) are introduced in Rak (2010) and Leepila et al. (2011) respectively. The application of assigning *k*-disjoint paths for a connection is to enhance the survivability for multi-link/node failures. The multi-cost network approach by Rak (2010) and Leepila et al.

(2011) could also offer partial protection through multipath provisioning (Roy and Mukherjee 2008; Das et al., 2009; Huang et al., 2010a; Huang et al., 2010b). In multipath provisioning a connection between the source and destination node would be established through a set of link-disjoint paths where the total bandwidth offer by the respective link-disjoint paths satisfy the connection bandwidth requirement. Partial protection is observed by the amount of guarantee bandwidth still available should path failure occur. It is mentioned that multipath provisioning is more resource efficient and a less expensive approach when compared to full protection.

There are however several disadvantages apparent in multipath provisioning. Among the major issue is the need to frequently synchronize data streams at the destination node. This is a serious challenge especially for real-time applications. Additionally, to constantly handle all incoming traffic streams, high speed buffer would be required at the end node. Although synchronization is also needed by the α+1 approach, it is mostly required when the primary path fails and the connection has to be switch over to the secondary backup path. As such, the α+1 protection is more practical in terms of implementation and suited to handle real-time applications compared to the multi-cost network approach.

3.4 Discussions on Link-Disjoint Paths Cost Function

Now, let P_{RF} and S_{RF} be the costs of primary and secondary paths, respectively, of the RF method, and P_{LB} and S_{LB} those of the LB algorithm.

Accordingly, the P_{SB} and S_{SB} are the primary and secondary path-cost by the Suurballe (SB) algorithm. It is obvious from the above discussions that the results produced by the LB and Suurballe algorithm are the same. Therefore, $P_{LB} = P_{SB}$ and $S_{LB} = S_{SB}$. As such, the following discussions apply to the solutions produced by both methods. Taking into account the characteristics of the RF and LB solutions, we have

$$P_{RF} \le P_{LB} \text{ , and} \tag{3}$$

$$P_{RF} + S_{RF} \ge P_{LB} + S_{LB} \tag{4}$$

In fact, eq. (1) has a linear characteristic. Assume that a network has distinct primary path and secondary path solutions by the RF and LB algorithms, respectively. Figure 3.11 illustrates a typical graph that shows the relationship between the total cost and α, based on the solutions obtained by RF and LB methods, as α varies from 0 to 1.

Note that it is obvious when $P_{RF} = P_{LB}$, the two lines coincide with each other and this results in the same solution for both the RF and LB algorithms. When the equality does not hold for eq. (3) and eq. (4), however, we can define the intersection point of these two lines as (y, C^*), as shown in Figure 3.11, where $C^* = P_{RF} + S_{RF} \cdot y = P_{LB} + S_{LB} \cdot y$. It is easy to solve that

$$y = \frac{P_{LB} - P_{RF}}{S_{RF} - S_{LB}}, \text{ and} \tag{5}$$

$$C^* = \frac{P_{LB} S_{RF} - P_{RF} S_{LB}}{S_{RF} - S_{LB}} \tag{6}$$

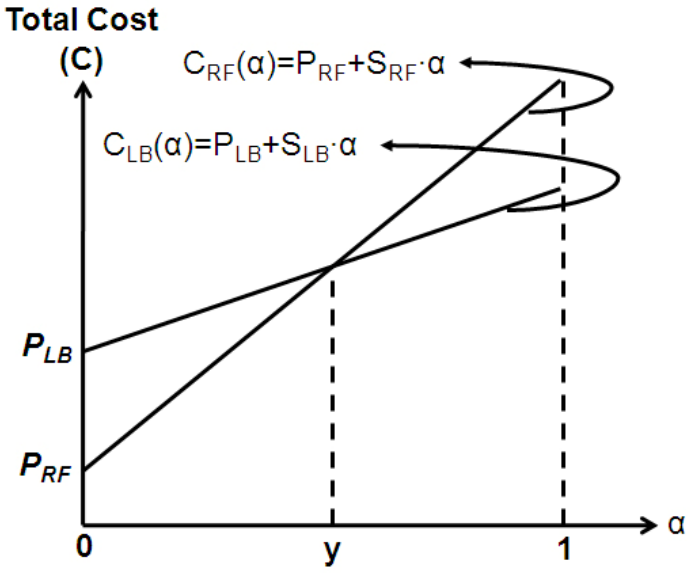

Figure 3.11: Cost functions of distinct RF and LB solutions

3.5 The Mid-Optimal (MO) Solution

Suppose now, the α+1 protection is to be implemented into the example network of Figure 3.1. Let α = 0.4. Based on the RF solution from Figure 3.4, the total cost to implement the α+1, as defined in eq. (1), is $C_{RF}(0.4)=5+(0.4)\times12=9.8$. For the LB solution given in Figure 3.8, the total cost to implement the α+1 is $C_{LB}(0.4)=7+(0.4)\times7=9.8$.

In actual fact, however, the optimal solution is neither the solution by RF nor LB when α = 0.4. Figure 3.12 depicts such an optimal solution (primary path: a-b-e-f, primary path cost = 1+4+1 = 6, secondary path: a-c-d-f, secondary path cost = 4+1+4 = 9) which has a lower cost than RF and LB

solutions, where the total cost, *C'*, to adopt this latest solution is given by *C'*(0.4)=6+(0.4)×9=9.6.

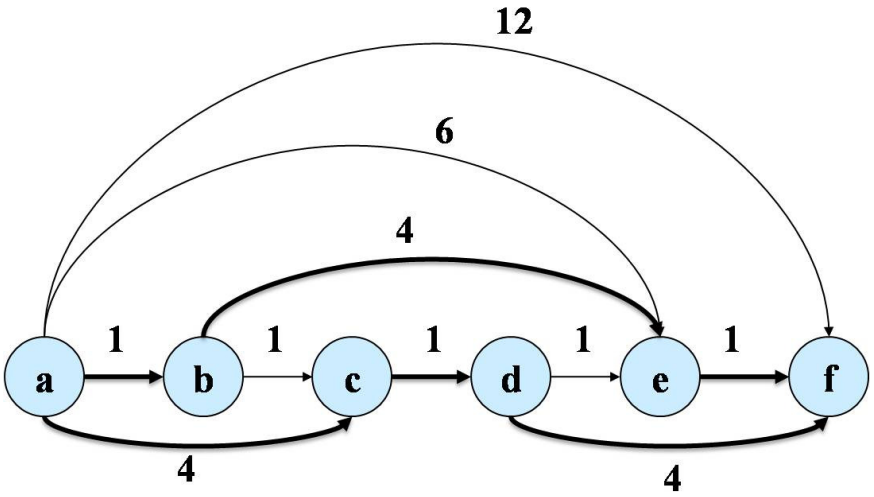

Figure 3.12: Optimal solution when α = 0.4

In some instances, if an optimal solution exists at some value α where it is neither by the RF nor the LB, we name this solution as the mid-optimal (MO) solution (Gan and Liew 2009; Gan and Liew 2010a; Gan and Liew 2010b). Let the MO solution be found at $\alpha = q$, where $0 < q < 1$, and let P_{MO} and S_{MO} be the costs of primary and secondary paths, respectively, given by such an MO solution. We have

$$P_{MO} + qS_{MO} < P_{RF} + qS_{RF}, \quad \text{and} \tag{7}$$

$$P_{MO} + qS_{MO} < P_{LB} + qS_{LB}, \tag{8}$$

Figure 3.13 illustrates the occurrence of an MO solution when $x<\alpha<z$. For the next section, we will look into the conditions for an MO solution if it does exist.

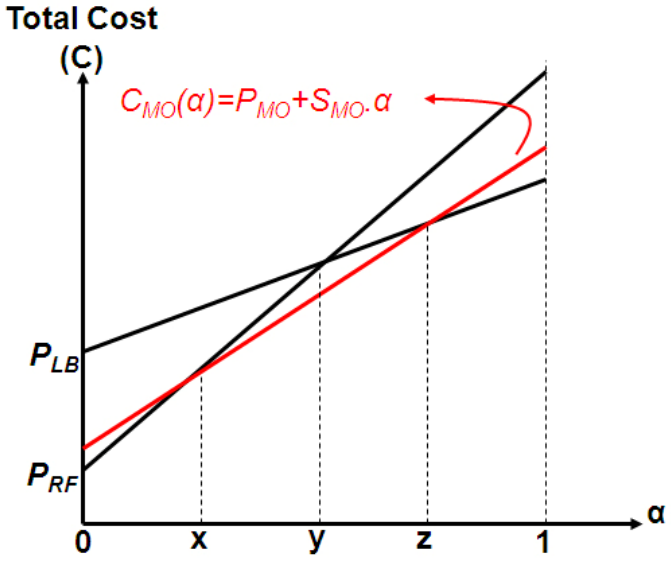

Figure 3.13: Typical total cost function of distinct RF method and LB path protection solution with the occurrence of a mid-optimal solution

3.5.1 Properties of the Mid-Optimal (MO) Solution

It is shown in the previous section that the implementation of the α+1 protection result in the occurrence of the mid-optimal (MO) solution in which existing link-disjoint paths-finding algorithms such as the remove-find (RF) method and the link-disjoint Bhandari (LB) algorithm are unable to identify. As such there is a need to construct an optimum paths-finding algorithm specifically for the application of the α+1 protection with the capability to obtain the MO solution.

Several properties that dictate the occurrence of the MO solutions are observed (Gan and Liew 2009; Gan and Liew 2010a). For the proposed

optimum algorithm to function effectively and efficiently, these properties have to be taken into consideration. Intuitively, from Figure 3.13, the following properties of the MO solution can be distinguished: 1) At α=0, the MO solution cannot have a lower cost than the RF solution because the RF solution consist of the shortest path as the primary path. 2) At α=1, the LB solution has the lowest cost (Guo et al., 2003) and therefore the MO solution have a lower cost than the LB. 3) The MO solution is optimal only when $0 < \alpha < 1$. 4) Additionally, suppose the RF and LB result in the same solution, then the MO solution cannot exist.

The following derives the properties observed in the MO solution.

Lemma: Consider a pair of source and destination nodes in a network graph. If the RF and LB algorithms result in the same pair of link disjoint paths, there is no MO solution.

Proof:

We prove the lemma by contradiction.

If RF and LB have the same solution, then we have, for all α,

$$P_{RF} + \alpha S_{RF} = P_{LB} + \alpha S_{LB} = P + \alpha S \qquad (9)$$

Suppose an optimal solution, that is not similar with the RF and LB solution, exists at $\alpha = q$ where $0 < q < 1$. Let the cost of the primary and secondary paths be denoted by P' and S', respectively. Thus,

$$(P' + qS') - (P + qS) < 0 \qquad (10)$$

Let

$$u = P' - P \tag{11}$$

$$v = S' - S \tag{12}$$

We define the cost difference function as,

$$Cost\ Diff = (P' + \alpha S') - (P + \alpha S) = u + \alpha v \tag{13}$$

When $\alpha = 1$, since the LB solution is always the optimal, we have

$$u + v > 0 \Rightarrow v > -u \tag{14}$$

When $\alpha = q$, from (10) we have

$$u + qv < 0 \Rightarrow v < \frac{-u}{q} \tag{15}$$

Combining the facts from (14) and (15) yields,

$$-u < v < \frac{-u}{q} \tag{16}$$

When $\alpha = 0$, on the other hand, since the RF solution is always the optimal, we have

$$u > 0 \tag{17}$$

And thus for $0 < q < 1$, the following inequality is always true,

$$\frac{-u}{q} < -u \tag{18}$$

Since (16) contradicts against (18), there cannot exist an MO solution if RF and LB algorithms produce the same solution.

Q.E.D

By referring to Figure 3.13, the mid-optimal solution that appears in the α+1 protection scheme is found to exist only when specific conditions are satisfied.

Theorem 1: Given a pair of link-disjoint paths with primary and secondary costs *P'* and *S'*, respectively, for a pair of source and destination nodes in a network graph, it is an MO solution if and only if the following conditions are all satisfied:

$$\text{(i)}\ P_{RF} < P' < P_{LB}\ , \tag{19}$$

$$\text{(ii)}\ P_{RF} + S_{RF} > P' + S' > P_{LB} + S_{LB} \tag{20}$$

$$\text{(iii)}\ \frac{P' - P_{RF}}{S_{RF} - S'} < \frac{P_{LB} - P'}{S' - S_{LB}} \tag{21}$$

Proof:

For the "if" part, (20) – (19) yields

$$S_{RF} > S' > S_{LB} \tag{22}$$

From (20) and (21), define a value *q'* where

$$0 < \frac{P' - P_{RF}}{S_{RF} - S'} < q' < \frac{P_{LB} - P'}{S' - S_{LB}} < 1 \tag{23}$$

we have

$$\frac{P' - P_{RF}}{S_{RF} - S'} < q' \Rightarrow P' + q'S' < P_{RF} + q'S_{RF}, \quad \text{and,} \tag{24}$$

$$q' < \frac{P_{LB} - P'}{S' - S_{LB}} \Rightarrow P' + q'S' < P_{LB} + q'S_{LB} \tag{25}$$

Inequalities (24) and (25) show that the given solution is an MO solution at $\alpha = q'$.

For the "only if" part, we assume *P'* and *S'* constitute an MO solution, thus $P' = P_{MO}$, $S' = S_{MO}$, and (7) and (8) are true. Note that an MO solution is neither the same as a RF solution nor a LB solution. Since the primary path of the RF method is the least-cost / shortest path in the graph, hence it is always true that

$$P_{RF} < P' \tag{26}$$

On the other hand, it has been proven in Guo et al., (2003) that LB solution is the optimal for 1+1 full bandwidth protection, so it is always true that

$$P'+S' > P_{LB} + S_{LB} \tag{27}$$

To prove that $P' < P_{LB}$, from (27) we have

$$-qP'-qS' < -qP_{LB} - qS_{LB} \tag{28}$$

On the other hand, from (8) we have

$$P'+qS' < P_{LB} + qS_{LB} \tag{29}$$

Subsequently, from inequalities (28) and (29), we have

$$(1-q)P' < (1-q)P_{LB} \tag{30}$$

Since $q < 1$, we have $P' < P_{LB}$.

To prove that $P' + S' < P_{RF} + S_{RF}$, from (26) we have

$$(1-q)P_{RF} < (1-q)P' \tag{31}$$

On the other hand, from (7) we have

$$P'+qS' < P_{RF} + qS_{RF} \tag{32}$$

Therefore, from inequalities (31) and (32), we have

$$q(P'+S') < q(P_{RF} + S_{RF}) \tag{33}$$

And since $q > 0$, we have $P' + S' < P_{RF} + S_{RF}$.

To prove that (iii) is also true, from (7) and (8) we have

$$\frac{P'-P_{RF}}{S_{RF} - S'} < q \tag{34}$$

$$q < \frac{P_{LB} - P'}{S'-S_{LB}} \tag{35}$$

Thus, (iii) is true as well.

Q. E. D.

It is fairly obvious that there is a possibility of more than one MO solution to exist in a graph. Suppose that a graph contains n MO solutions, where $n \geq 1$. Let the costs of the pairs of primary and secondary paths of these MO solutions be labeled according to the following manner:

$$(P_{MO(1)}, S_{MO(1)}), (P_{MO(2)}, S_{MO(2)}), (P_{MO(3)}, S_{MO(3)}), ..., (P_{MO(n)}, S_{MO(n)})$$

$$\text{where } P_{RF} < P_{MO(1)} < P_{MO(2)} < P_{MO(3)} < ... < P_{MO(n)} < P_{LB}$$

The derived MO properties in (19), (20) and (21) still apply for all these MO solutions. Moreover, there are additional characteristics that are associated between these MO solutions as shown in the following corollary.

Corollary 1: Consider a network graph which has n MO solutions between a source node and a destination node. Let the costs of the primary and secondary paths of these MO solutions be denoted by $P_{MO(i)}$ and $S_{MO(i)}$, respectively, such that $P_{MO(i-1)} < P_{MO(i)}$, where i=1,2….n and $P_{MO(0)}$ and $S_{MO(0)}$ be defined as P_{RF} and S_{RF}, respectively. The following inequalities are true for all MO solutions.

$$\text{(i) } P_{MO(i-1)} + S_{MO(i-1)} > P_{MO(i)} + S_{MO(i)} \tag{36}$$

$$\text{(ii) } \frac{P_{MO(i)} - P_{MO(i-1)}}{S_{MO(i-1)} - S_{MO(i)}} < \frac{P_{LB} - P_{MO(i)}}{S_{MO(i)} - S_{LB}} \tag{37}$$

Proof:

We prove (i) by contradiction.

Let ($P_{MO(i)}$, $S_{MO(i)}$) be the cost of the optimal solution at $\alpha = y_1$, where $0<y_1<1$. Therefore the subsequent inequalities are true

$$P_{MO(i-1)} + y_1 S_{MO(i-1)} > P_{MO(i)} + y_1 S_{MO(i)} \quad (38)$$

$$P_{LB} + y_1 S_{LB} > P_{MO(i)} + y_1 S_{MO(i)} \quad (39)$$

We know the inequality below to be always true by definition

$$P_{MO(i-1)} < P_{MO(i)} \quad (40)$$

Now suppose

$$P_{MO(i-1)} + S_{MO(i-1)} \leq P_{MO(i)} + S_{MO(i)} \quad (41)$$

From (40) and (41), the subsequent inequality is true for all $0<\alpha<1$.

$$P_{MO(i-1)} + \alpha S_{MO(i-1)} < P_{MO(i)} + \alpha S_{MO(i)} \quad (42)$$

However, at $\alpha = y_1$, (42) contradicts (38). So (41) cannot be true. Therefore we can conclude that:

$$P_{MO(i-1)} + S_{MO(i-1)} > P_{MO(i)} + S_{MO(i)}$$

To prove that (ii) is also true, from (38) and (39) we have

$$\frac{P_{MO(i)} - P_{MO(i-1)}}{S_{MO(i-1)} - S_{MO(i)}} < y_1 \quad (43)$$

$$\frac{P_{LB} - P_{MO(i)}}{S_{MO(i)} - S_{LB}} > y_1 \quad (44)$$

Thus (ii) is true as well

Q. E. D.

With the above corollary, an effective algorithm for finding link-disjoint optimal paths can be constructed. Suppose that P_{LB} is the L^{th} shortest path in the graph, the primary path of any existing MO solution must be the

K^{th} shortest path of the graph where $1<K<L$. This forms the boundary limit that will be taken to account in the construction the proposed algorithm to reduce the running time as well as increase the efficiency of identifying the optimal solution.

CHAPTER 4

α+1 OPTIMUM LINK-DISJOINT PATHS-FINDING ALGORITHM

For this chapter, a review on relevant existing optimum paths-finding algorithms applicable to the α+1 protection are discussed. Subsequently, we present two optimum link-disjoint paths-finding algorithm for the α+1 path protection scheme, labeled as the α-optimum (AO) and the reversed α-optimum (RAO) paths-finding algorithm. This is followed by simulation results on the performance of the two proposed AO and RAO link-disjoint paths-finding algorithms.

4.1 Review on Relevant Existing α+1 Optimum Paths-Finding Algorithms

The paths-finding algorithm proposed under maximum-connection-availability design (MCAD) (Torbatore et al., 2006) for available-based routing, applies both the LB and the RF algorithm on each connection request and selects the solution which returns the minimum cost. Under the α+1 protection, the algorithm in Torbatore et al., (2006) will not be able to account together the MO solution which, as mentioned, is an optimal solution that is neither the solution by LB nor RF and exist when $0<\alpha<1$.

Several algorithms (Laborczi et al., 2001; Ho and Mouftah 2002; Fang et al., 2005; Yang et al., 2005a) utilizes the *K*-shortest path algorithm such as Yen's (Yen 1971), to identify paths in increasing cost order. For each iteration cycle, after the k^{th} shortest path is obtained, the shortest link-disjoint path against the current k^{th} shortest path is identified. This pair of link-disjoint path would then be evaluated to ascertain whether it is an optimal solution with respect to a specific α value. However, without the integration of a stop rule (such as the boundary limit for the MO solution derived in the previous chapter), an exhaustive search would be performed which substantially increases the running time. It was even acknowledged by the authors in Laborczi et al., (2001) that the efficiency of their method remains a concern.

Another method proposed in Laborczi et al., (2001) finds the optimal pair of link-disjoint paths by means of integer linear programming (ILP). The high complexity by the ILP model (especially for large networks) is subsequently reduced through the relaxation of its constraints referred to as Linear Programming Relaxation (LPR) and Single Flow Relaxation (SFR) to become a linear program (LP) and minimal network cost flow (MNCF) problem respectively. Nevertheless, these methods do not guarantee that a solution could be found even though it exists. The cause as pointed out in the paper is due to the "split-flow" problem. The occurrence of "split-flow" yields results which are not useful. Although there are additional approaches being presented to remedy the issue, the corresponding solution obtained may not necessarily be optimal. Furthermore, these additional approaches in dealing with the "split-flow" problem add to the complexity.

The algorithm denoted as DP2LC in Gomes et al., (2008) also incorporates the *K*-shortest path algorithm to enumerate pairs of possible link-disjoint paths in the network with a determined lower and upper bound which represents the algorithm's stop rule. Basically, there are two main iterative cycles in the algorithm labeled as 'Procedure $\mathscr{A}$ and 'Procedure $\mathscr{B}$, each comprising a subroutine. In general, the subroutine will iterate through the graph, using the *K*-shortest path algorithm, according to a specified set of cost metric permutation to find the best candidate pair of path. Nevertheless, the complexity of the algorithm is undefined which could result in an unbounded running time in the worst case.

For the next section, we present two proposed paths-finding algorithms labeled as the α-optimum (AO) and the reversed α-optimum (RAO) paths-finding algorithm These two algorithms are specially constructed to effectively identify the optimal link-disjoint pair of path solutions for the α+1 protection scheme.

4.2 α-Optimum (AO) Link-Disjoint Paths-Finding Algorithm

Generally, the AO algorithm (Gan and Liew 2009; Gan and Liew 2010a; Gan and Liew 2010b) utilizes the *K*-shortest path algorithm proposed by Yen (Yen 1971). The Yen's algorithm is selected because it has the lowest worst-case complexity among *K*-shortest path ranking algorithm (Yen 1971; Martins and Pascoal 2003). The *K*-shortest path algorithm is utilized to

identify paths between the source and destination node in increasing cost order sequentially. In each progressing cycle of the algorithm, after the k^{th} shortest path is found, the links on this path are removed, and the link-disjoint shortest path against the k^{th} shortest path is obtained. These two paths becomes the candidate for an MO solution. The candidate solution is then evaluated upon the MO properties derived in Chapter 3.5.1 to determine the acceptance or rejection as an MO solution. Importantly, according to the properties derived, the algorithm can be safely terminated upon reaching the LB solution and this represents the algorithm's stop rule. The derivation of the MO properties in Chapter 3.5.1 also increases the algorithm's efficiency as rules based on the derived MO properties are incorporated to exclude non-optimal solutions during execution.

The possibility of a 'trap topology' (Guo et al., 2003; Ou et al., 2003) in the graph is another concern. In a graph with 'trap topology', there is no link-disjoint path with respect to the shortest path, and thus the RF method would return no solution for path protection. This issue is also addressed in our algorithm. When a 'trap topology' occurs, the cost of the secondary path, S_{RF}, is assumed as very large, or $S_{RF} \sim \infty$. When α=0, the RF solution is still optimal as S_{RF} will be discounted from the total cost calculation. The properties of the MO solution derived in Section 3 still holds under this assumption.

The following describes the α-optimum (AO) link-disjoint paths-finding algorithm. Path notations in bold ($\mathbf{P_1}$, $\mathbf{S_1}$, $\mathbf{P_{RF}}$, $\mathbf{S_{RF}}$, $\mathbf{P_{LB}}$, $\mathbf{S_{LB}}$, $\mathbf{P_{MO(n)}}$,

$\mathbf{S_{MO(n)}}$, $\mathbf{P_k}$ and $\mathbf{S_k}$) denote the set of links that form their respective path. Regular path notations (P_1, S_1, P_{RF}, S_{RF}, P_{LB}, S_{LB}, $P_{MO(n)}$, $S_{MO(n)}$, P_k, and S_k) denote their path cost.

Algorithm α-optimum (AO)

{Define

s: source node

d: destination node

RF: Remove Find method (also known as two-step method)

LB: Link-disjoint Bhandari algorithm

MO: Mid-Optimal}

{Initialization (Note that k is the next shortest path to be found, where 1st shortest path can be easily found by the Dijkstra's algorithm)}

$k = 2$

$n = 1$

$\mathbf{P_{RF}} = \mathbf{S_{RF}} = \mathbf{P_{LB}} = \mathbf{S_{LB}} = \mathbf{P_{MO}} = \mathbf{S_{MO}} = \mathbf{P_k} = \mathbf{S_k} \Phi$

{ Algorithm starts}

STEP 1. Identify the LB solution between s and d, where $\mathbf{P_{LB}}$ and $\mathbf{S_{LB}}$ are the primary and secondary path of the LB solution respectively.

If no paths exist {No connection exists between s and d}

Go to STEP 11

else

$P_{LB} \leftarrow$ cost ($\mathbf{P_{LB}}$) {LB solution found}

$S_{LB} \leftarrow$ cost ($\mathbf{S_{LB}}$)

end if

STEP 2. Determine the shortest path, $\mathbf{P_1}$, between s and d

if $\mathbf{P_1} = \mathbf{P_{LB}}$ {LB solution is the only optimal solution}

Go to STEP 11

end if

STEP 3. Remove all related links of path $\mathbf{P_1}$ from the graph and find the shortest path in the 'trimmed' graph, $\mathbf{S_1}$, between s and d.

---

if $\mathbf{S_1}$ is found

$\mathbf{P_{RF}} \leftarrow \mathbf{P_1}$ {RF solution found}

$\mathbf{S_{RF}} \leftarrow \mathbf{S_1}$

$P_{RF} \leftarrow$ cost ($\mathbf{P_1}$)

$S_{RF} \leftarrow$ cost ($\mathbf{S_1}$)

Restore graph to original state

else {Graph has trap topology}

$\mathbf{P_{RF}} \leftarrow \mathbf{P_1}$ {RF Solution in a trap topology graph}

$\mathbf{S_{RF}} \leftarrow \Phi$

$P_{RF} \leftarrow$ cost ($\mathbf{P_1}$)

$S_{RF} \leftarrow \infty$

Restore graph to original state

end if

STEP 4. Determine the k^{th} shortest path, $\mathbf{P_k}$, between s and d

$P_k \leftarrow$ cost ($\mathbf{P_k}$)

if $P_k \geq P_{LB}$ {Stop parameter}

Go to STEP 11

end if

STEP 5. Remove all related links of path $\mathbf{P_k}$ from the graph

STEP 6. Find the shortest path in the modified graph, $\mathbf{S_k}$, from s and d.

if no path exists

Go to STEP 10

end if

STEP 7. $S_k \leftarrow$ cost ($\mathbf{S_k}$)

if $S_k < P_k$

Go to STEP 10

else

if n > 1

Go to STEP 9

else

Go to STEP 8

end if

end if

STEP 8. **If** $P_{RF}+S_{RF}<P_k+S_k<P_{LB}+S_{LB}$

if $\frac{P_k - P_{RF}}{S_{RF} - S_k} < \frac{P_{LB} - P_k}{S_k - S_{LB}}$

$\mathbf{P_{MO(n)}} \leftarrow \mathbf{P_k}$

$P_{MO(n)} \leftarrow$ cost($\mathbf{P_k}$)

{First MO solution found}

$\mathbf{S_{MO(n)}} \leftarrow \mathbf{S_k}$

$S_{MO(n)} \leftarrow$ cost($\mathbf{S_k}$)

n ← n+1

Go to STEP 10

end if

else

Go to STEP 10

end if

STEP 9. **If** $P_{MO(n-1)}+S_{MO(n-1)}<P_k+S_k<P_{LB}+S_{LB}$

if $\frac{P_k - P_{MO\,(n-1)}}{S_{MO\,(n-1)} - S_k} < \frac{P_{LB} - P_k}{S_k - S_{LB}}$

$\mathbf{P_{MO(n)}} \leftarrow \mathbf{P_k}$

$P_{MO(n)} \leftarrow$ cos($\mathbf{P_k}$)

{Subsequent MO solutions found}

$\mathbf{S_{MO(n)}} \leftarrow \mathbf{S_k}$

$S_{MO(n)} \leftarrow$ cost($\mathbf{S_k}$)

$n \leftarrow n+1$

Go to STEP 10

end if

else

Go to STEP 10

end if

STEP 10. Restore graph to original state

$k \leftarrow k+1$

Go to STEP 4

STEP 11. End algorithm

Figure 4.1: α-Optimum (AO) paths-finding algorithm

It should be noted that the proposed AO algorithm has a complexity of $O(K_o n_o(m_o + n_o \log n_o))$ where K_o represents the number of paths between the source and destination nodes whose cost are larger than P_{RF} but smaller than P_{LB}, n_o indicates the number of nodes in the graph, and m_o is the number of edges (links). The core complexity of this algorithm is fundamentally constructed from the Yen's algorithm (Martins and Pascoal 2003). The algorithm is applied at the call admission level, which in general allows more time in setting up a call.

Apart from the MO properties derived in Chapter 3.5.1, an additional characteristic can also be observed in the MO solutions. That is, the primary and secondary paths of the MO solution, denoted as $\mathbf{P_{MO}}$ and $\mathbf{S_{MO}}$ respectively, cannot be link-disjoint with the least-cost path, $\mathbf{P_1}$. We prove this characteristic in the next corollary.

Corollary 2: Given a pair of source and destination nodes in a network graph, if an MO solution exists and the solution constitutes a link-disjoint pair of paths $\mathbf{P_{MO}}$ and $\mathbf{S_{MO}}$, both paths are not link-disjoint with the least-cost path, $\mathbf{P_1}$.

Proof:

We prove Corollary 2 by contradiction.

Let the costs of paths $\mathbf{P_{MO}}$, $\mathbf{S_{MO}}$, $\mathbf{P_1}$, $\mathbf{P_{LB}}$ and $\mathbf{S_{LB}}$ be denoted accordingly as P_{MO}, S_{MO}, P_1, P_{LB} and S_{LB}. Recall that $\mathbf{P_{LB}}$ and $\mathbf{S_{LB}}$ are the respective primary

and secondary paths of the LB solution, and they form a pair of link-disjoint paths which has the least cost sum when α=1.

Suppose $\mathbf{P_{MO}}$ is link-disjoint with $\mathbf{P_1}$

Since this pair of paths are said to be link-disjoint, therefore $\mathbf{P_1}$ and $\mathbf{P_{MO}}$ can be a possible solution for the α+1 protection.

Inequality (19) from Theorem 1 has shown that $P_{RF}<P_{MO}<P_{LB}$ where P_{RF} is the primary path cost of the RF solution and hence $P_{RF} = P_1$.

Given that $P_{LB} \leq S_{LB}$, we have $P_1< P_{LB}$ and $P_{MO} < S_{LB}$. Thus

$$P_1 + P_{MO} < P_{LB} + S_{LB} \qquad (45)$$

However, inequality (45) contradicts the fact in Guo et al., (2003) which proved that the LB solution's total cost, $P_{LB} + S_{LB}$, is optimal at 1+1 full bandwidth protection.

Therefore $\mathbf{P_{MO}}$ cannot be link-disjoint with $\mathbf{P_1}$

Now suppose $\mathbf{S_{MO}}$ is link-disjoint with $\mathbf{P_1}$

Since this pair of paths are said to be link-disjoint, therefore $\mathbf{P_1}$ and $\mathbf{S_{MO}}$ can be a possible solution for the α+1 protection.

As $\mathbf{P_1}$ is the least-cost path, $P_1 < P_{MO}$. Thus for all $0 \leq \alpha \leq 1$, it is always true that

$$P_1 + \alpha S_{MO} < P_{MO} + \alpha S_{MO} \qquad (46)$$

On the other hand, by definition the MO solution must be optimal among all other solutions at some $\alpha = q$, where $0<q<1$, so that the following inequality is true

$$P_{MO} + qS_{MO} < P_1 + qS_{MO} \quad (47)$$

Since (47) contradicts (46), $\mathbf{S_{MO}}$ cannot be link-disjoint with $\mathbf{P_1}$

Q.E.D.

In fact Corollary 2 provides another method to find the MO solutions between a source and destination node in a network graph. That is, once the shortest path is identified, we can find all the possible "link-joint" paths against this shortest path, and test their validity to become an MO solution.

4.3 Reversed α-Optimum (RAO) Link-Disjoint Paths-Finding Algorithm

With the additional properties derived in Corollary 2, a supplementary optimum paths-finding algorithm can be constructed for the α+1 protection. The algorithm is labeled as the reversed α-optimum (RAO) algorithm (Gan and Liew 2011).

Generally the RAO algorithm is an extended form of the LB algorithm. Similar to the LB algorithm, the graph will initially be modified. This is done by finding the shortest path in the graph, $\mathbf{P_1}$, and replacing it with $\mathbf{-P_1}$. $\mathbf{-P_1}$ is obtained by reversing the direction of the links associated to $\mathbf{P_1}$ and converting the cost on the links along path $\mathbf{P_1}$ to its equivalent negative value. The *K*-

shortest path algorithm is then used to identify paths between the source and destination node in increasing cost order sequentially, in the modified graph. We denote the k^{th} shortest path as $\mathbf{P'_k}$. For every $\mathbf{P'_k}$ that is found, the links which are shared with $-\mathbf{P_1}$ are temporary removed while the remaining links of $-\mathbf{P_1}$ are restored to their original direction and values. The links of $\mathbf{P'_k}$ and $\mathbf{P_1}$ which are left will be grouped to form a pair of link-disjoint path, $\mathbf{P_k}$ and $\mathbf{S_k}$ where $P_k \leq S_k$. This pair would be the candidate for an MO solution. The candidate solution is then evaluated upon the MO properties derived in Chapter 3.5.1 to determine the acceptance or rejection as an MO solution. The algorithm stops when the k^{th} shortest path ($\mathbf{P'_k}$) found is link-disjoint with $-\mathbf{P_1}$. This stop rule is based on the derivation of Corollary 2. It can be further validated using derivations for LB algorithm found in Guo et al., (2003) that subsequent solutions beyond this limit are not optimal.

The 'trap topology' is not an issue for the RAO algorithm as the core structure of the algorithm is based on the LB algorithm. Due to its characteristic, the LB algorithm is able to circumvent the 'trap topology' (Guo et al., 2003).

The following describes the reversed α-optimum (RAO) link-disjoint paths-finding algorithm. Similarly, the notations in bold ($\mathbf{P_1}$, $\mathbf{S_1}$, $\mathbf{P_{RF}}$, $\mathbf{S_{RF}}$, $\mathbf{P_{LB}}$, $\mathbf{S_{LB}}$, $\mathbf{P_{MO(n)}}$, $\mathbf{S_{MO(n)}}$, $\mathbf{P'_k}$, $\mathbf{P_k}$ and $\mathbf{S_k}$) denote the set of links that form their respective paths while regular notations (P_1, S_1, P_{RF}, S_{RF}, P_{LB}, S_{LB}, $P_{MO(n)}$, $S_{MO(n)}$, P'_k, P_k, and S_k) denote their respective path costs.

Algorithm Reversed α-Optimum (RAO)

{Define

s: source node

d: destination node

RF: Remove Find method (also known as two-step method)

LB: Link-disjoint Bhandari algorithm

MO: Mid-Optimal}

{Initialization (Note that k is the next shortest path to be found in the modified graph, where the 1st shortest path can be found using the Bellman-Ford's algorithm. This is due to the presence of negative link cost in the modified graph)}

$k = 2$

$n = 1$

$\mathbf{P_{RF}} = \mathbf{S_{RF}} = \mathbf{P_{LB}} = \mathbf{S_{LB}} = \mathbf{P_{MO}} = \mathbf{S_{MO}} = \mathbf{P_k} = \mathbf{S_k} = \mathbf{P'_k} = \Phi$

{ Algorithm starts}

STEP 1. Identify the LB solution between s and d, where $\mathbf{P_{LB}}$ and $\mathbf{S_{LB}}$ are the primary and secondary path of the LB solution respectively.

If no paths exist {No connection exists between s and d}

Go to STEP 13

else

$P_{LB} \leftarrow$ cost ($\mathbf{P_{LB}}$) {LB solution found}

$S_{LB} \leftarrow$ cost ($\mathbf{S_{LB}}$)

end if

STEP 2. Determine the shortest path, $\mathbf{P_1}$, between s and d

if $\mathbf{P_1} = \mathbf{P_{LB}}$ {LB solution is the only optimal solution}

Go to STEP 13

end if

STEP 3. Remove all related links of path $\mathbf{P_1}$ from the graph and find the shortest path in the 'trimmed' graph, $\mathbf{S_1}$, between s and d.

if $\mathbf{S_1}$ is found

$\mathbf{P_{RF}} \leftarrow \mathbf{P_1}$ {RF solution found}

$\mathbf{S_{RF}} \leftarrow \mathbf{S_1}$

$P_{RF} \leftarrow$ cost ($\mathbf{P_1}$)

$S_{RF} \leftarrow$ cost ($\mathbf{S_1}$)

Restore graph to original state

else {Graph has trap topology}

$\mathbf{P_{RF}} \leftarrow \mathbf{P_1}$ {RF Solution in a trap topology graph}

$\mathbf{S_{RF}} \leftarrow \mathbf{\Phi}$

$P_{RF} \leftarrow$ cost ($\mathbf{P_1}$)

$S_{RF} \leftarrow \infty$

Restore graph to original state

end if

STEP 4. Replace $\mathbf{P_1}$ with $\mathbf{-P_1}$ in the graph. $\mathbf{-P_1}$ is obtained by reversing the direction of all related links of path $\mathbf{P_1}$ from the graph and replacing the associated cost of the $\mathbf{P_1}$ links with equivalent negative values.

STEP 5. Determine the k^{th} shortest path, $\mathbf{P'_k}$, between s and d in the modified graph constructed in STEP 4.

STEP 6. Remove the links along path $\mathbf{P'_k}$, that is shared with $\mathbf{-P_1}$

if $\mathbf{P'_k} \cap \mathbf{-P_1} = \mathbf{\Phi}$ {Stop parameter}

Go to STEP 13

end if

STEP 7. Restore the remaining links of $\mathbf{-P_1}$ back to its original direction together with their respective initial non-negative cost

STEP 8. Group the remaining links of $\mathbf{P'_k}$ and $\mathbf{P_1}$ into two paths $\mathbf{P_k}$ and $\mathbf{S_k}$ where; cost ($\mathbf{P_k}$) $\leq$ cost ($\mathbf{S_k}$)

STEP 9. $P_k \leftarrow$ cost ($\mathbf{P_k}$)

$S_k \leftarrow$ cost ($\mathbf{S_k}$)

if n > 1

Go to STEP 11

else

Go to STEP 10

end if

STEP 10. **If** $P_{RF}+S_{RF}<P_k+S_k<P_{LB}+S_{LB}$

if $\frac{P_k - P_{RF}}{S_{RF} - S_k} < \frac{P_{LB} - P_k}{S_k - S_{LB}}$

$\mathbf{P_{MO(n)}} \leftarrow \mathbf{P_k}$

$P_{MO(n)} \leftarrow$ cost ($\mathbf{P_k}$)

{First MO solution found}

$\mathbf{S_{MO(n)}} \leftarrow \mathbf{S_k}$

$S_{MO(n)} \leftarrow$ cost ($\mathbf{S_k}$)

n ← n+1

Go to STEP 12

end if

else

Go to STEP 12

end if

STEP 11. **If** $P_{MO(n-1)}+S_{MO(n-1)}<P_k+S_k<P_{LB}+S_{LB}$

if $\frac{P_k - P_{MO(n-1)}}{S_{MO(n-1)} - S_k} < \frac{P_{LB} - P_k}{S_k - S_{LB}}$

$\mathbf{P_{MO(n)}} \leftarrow \mathbf{P_k}$

$P_{MO(n)} \leftarrow$ cost ($\mathbf{P_k}$)

{Subsequent MO solutions found}

$\mathbf{S_{MO(n)}} \leftarrow \mathbf{S_k}$

$S_{MO(n)} \leftarrow$ cost ($\mathbf{S_k}$)

$n \leftarrow n+1$

Go to STEP 12

end if

else

Go to STEP 12

end if

STEP 12. Restore graph to original state

$k \leftarrow k+1$

Go to STEP 4

STEP 13. End algorithm

Figure 4.2: Reversed α-Optimum (RAO) paths-finding algorithm

The complexity of RAO is given as $O(K_o m_o n_o^2)$, where K_o represents the number of paths in the modified graph between the source and destination node which are "link-joint" with path $-\mathbf{P_1}$, n_o indicates the number of nodes in the graph, and m_o is the number of edges (links). The complexity of RAO is larger by a factor of n_o when compared to the AO. This is because there are negative link costs in the modified graph. As a result, the *K*-shortest path algorithm in RAO will utilize the Bellman Ford algorithm (instead of the Dijkstra's algorithm) which leads to a higher overall complexity.

Although having a higher complexity than the AO, the RAO has a more novel stop parameter. The main iteration cycle in RAO stops when the K^{th} shortest path found in the modified graph is link disjoint with $-\mathbf{P_1}$. Additionally the RAO is able to circumvent the 'trap topology' thereby eliminating unnecessary iterations that do not produce the expected candidate solution as may happen in the case of AO. This generally results in the RAO having to run fewer iteration cycles compared to the AO.

Moreover, by replacing the graph modification (in Step 4 of the RAO) with the Suurballe graph transformation, the negative links in the graph can be removed. Hence, the Dijkstra's algorithm is applicable in the transformed graph instead of having to use the Bellman Ford algorithm. This slight adjustment to the RAO would reduce its complexity to be $O(K_o n_o(m_o + n_o \log n_o))$ which is similar to the AO algorithm.

In the next section, simulation results are presented to show the performance of the proposed AO and RAO paths-finding algorithms. Since both the AO and RAO paths-finding algorithms are designed to identify the optimal solutions (including the MO solution) for the α+1 protection, therefore both algorithms are generally expected to produce similar performance outcomes in terms of blocking rate and average cost of calls.

4.4 Simulation Results

4.4.1 Simulation over USAnet

Simulation is carried out on a 28 node 45 bidirectional link USAnet topology as shown in Figure 4.3. The capacity of each link is assumed to be W=16 units of bandwidth. The links are assumed to be weighted links. As mentioned, the link weights represent the cost (per unit bandwidth) incurred for data to traverse the particular link. Although there are numerous possible link-weights permutations, the particular combination explicitly highlight the average cost difference between the solutions provided by the RF, LB and the proposed AO and RAO paths-finding algorithms.

Calls arrive in accordance to a Poisson process with rate λ. Source and destination node is randomly selected based on a uniform distribution. The call duration is exponentially distributed with a mean of $1/\mu$. Therefore the Erlang load offered to the entire network is $\rho=\lambda/\mu$. Each point on the graph shows the average connection cost after 1 million calls.

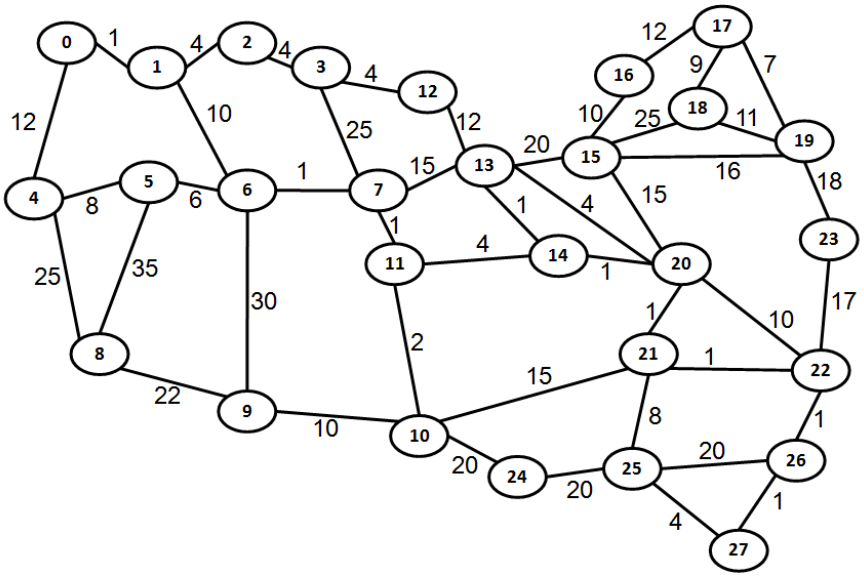

Figure 4.3: USAnet with preset link weights

A successful call request would reserve 1 unit of bandwidth on the primary path while on the secondary path the bandwidth reservation is proportional to α. If there is not enough capacity for either the primary or the secondary path's request, the call is blocked. The algorithms are modeled using C++ programming language and simulated on a system with Intel Core 2 Quad Q9300 processor at 2.5GHz with 8GB of RAM.

Simulation 1 (USAnet): Static Link-Cost

Utilizing the link-cost configuration of Figure 4.3, Figure 4.4 shows the average total costs for calls monitored between node 0 and node 27 with respect to α at the network Erlang load of 40. As this load value is considered low, no call blocking was noticed in the simulation for all α. The results in Figure 4.4 show that our proposed algorithms ensure minimum average total

cost for all values of α. It was also able to identify the MO solution (noticeable between α=0.2 to α=0.9). These were not achieved by the existing RF and LB paths-finding algorithms.

The result for AO/RAO in Figure 4.4 is such that when α is equal or very close to zero, it converges with the solution by RF. This observation verifies our discussion on the characteristic of the optimum α+1 solution in Chapter 1.2.1. Likewise, when α is equal or very close to one, AO/RAO is seen to converge with the LB solution as expected.

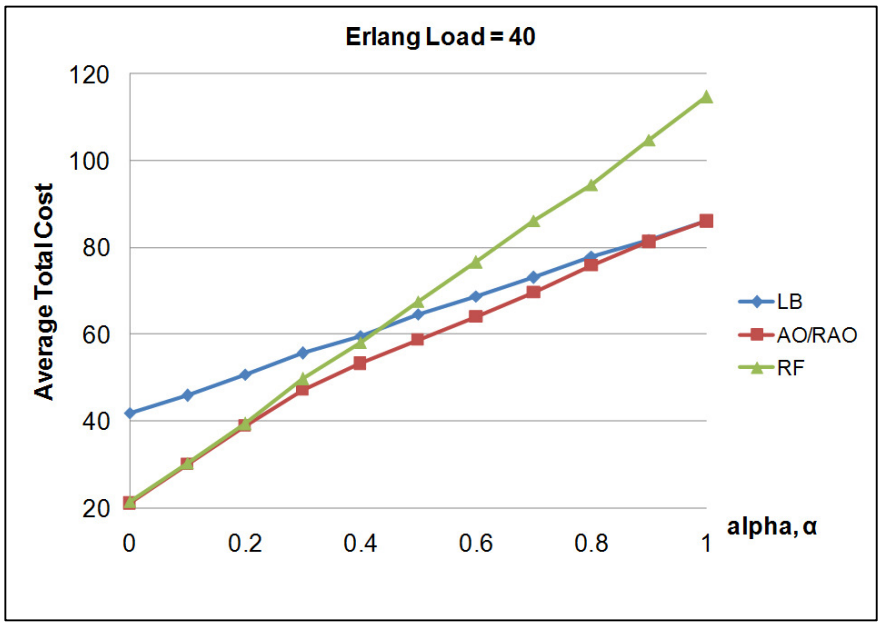

Figure 4.4: Average total cost at Erlang load=40 (static link cost)

Figure 4.5 and Figure 4.6 show the average total costs of AO, RAO, LB, and RF and their respective blocking probabilities at the network Erlang load of 100 while Figure 4.7 and Figure 4.8 are respectively the average total

cost and blocking probability evaluated at the network Erlang load of 150. As the network load transition to a higher setting, there is an overall increase in the average total cost as well as blocking probability. The reason being the network is handling more calls, and it is likely that links along the optimal paths are fully utilized most of the time. Hence, it is understandable that the costs of incoming calls, at times, would be higher because these calls have to be routed through a more expensive but still available set of links. Since at this state the network's resource is scarce, there is a higher occurrence of call blocking.

Figure 4.7 shows that despite the network's resource limitations at high loads, the AO and RAO managed to generally maintain a minimum average total cost compared to LB and RF. Figure 4.8 further demonstrates that in terms of blocking probability, the trade-off by AO and RAO is minimal in contrast with the cost savings attained over RF and LB.

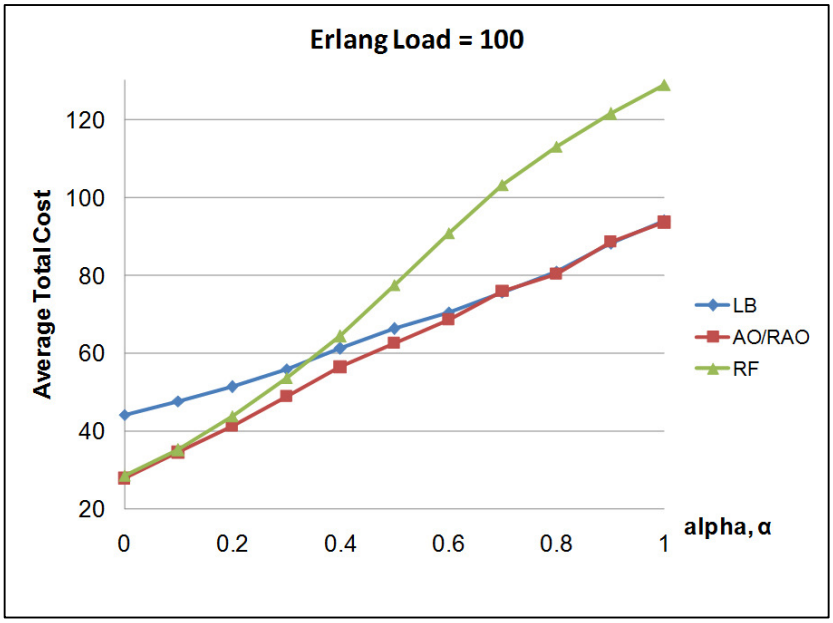

Figure 4.5: Average total cost at Erlang load=100 (static link cost)

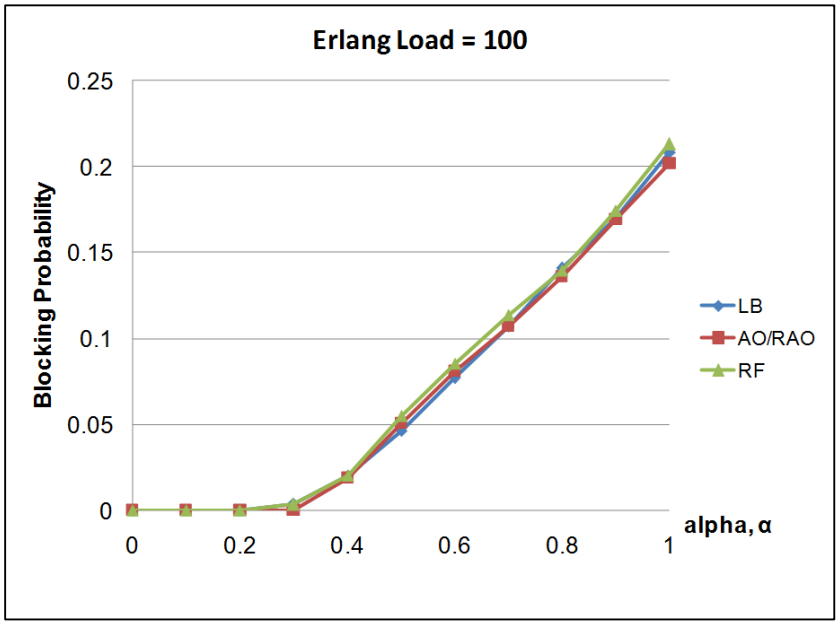

Figure 4.6: Blocking probability at Erlang Load=100 (static link cost)

It is observed in Figure 4.6 and Figure 4.7 that the RF is inclined towards a higher average total cost compared to the AO, RAO and LB. This is

because unlike the solutions by AO, RAO and LB, the solutions by RF have a large cost difference between its primary path and secondary path. This is due to characteristic of the RF solution in which its secondary path has to be completely link-disjoint from the current shortest path. Consequently as the value of α increases, the cost of the secondary path in the RF solution would cause the disparity in average total cost by the RF to be apparent.

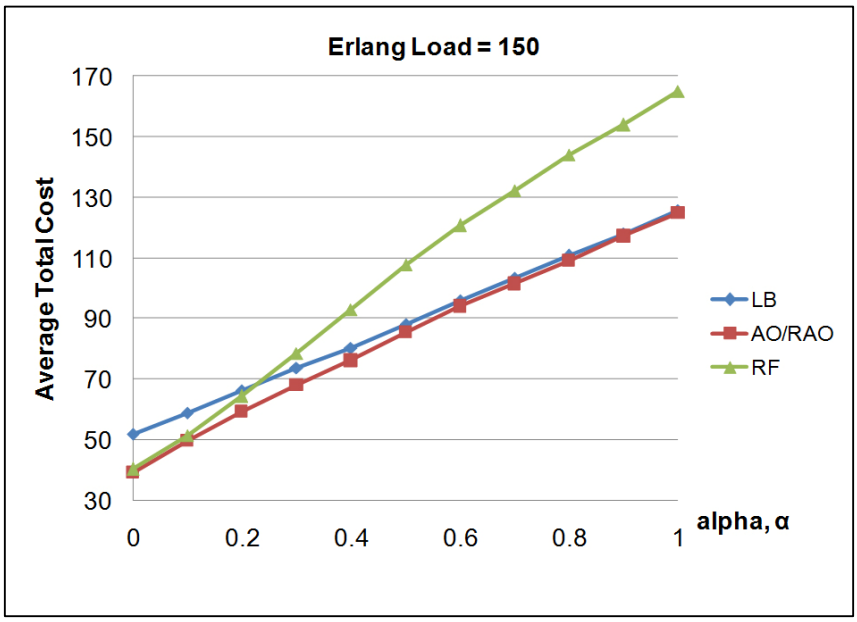

Figure 4.7: Average total cost at Erlang load=150 (static link cost)

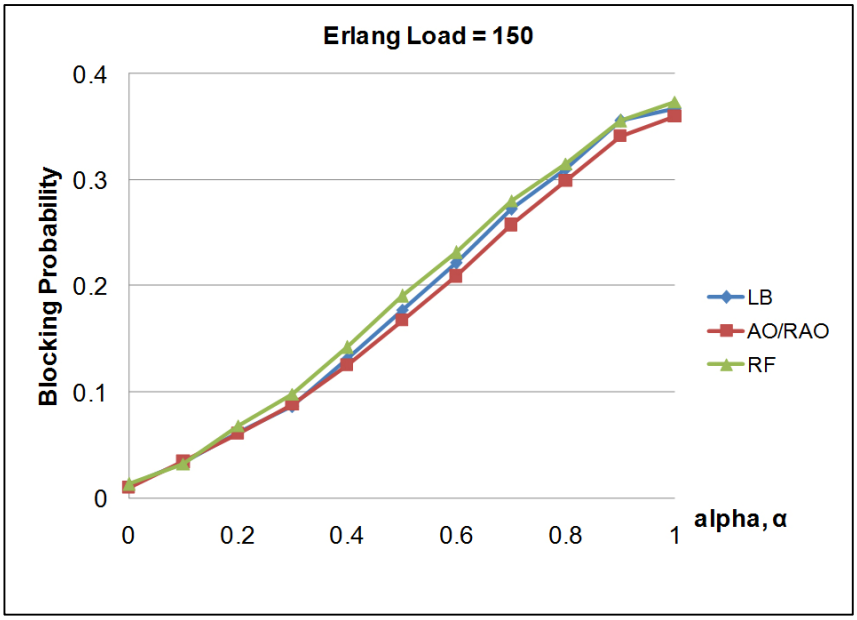

Figure 4.8: Blocking probability at Erlang load=150 (static link cost)

Simulation 2 (USAnet): Dynamic Link-Cost

In the second phase of the simulation, the network is assigned with dynamic link cost. The implementation of dynamic link cost provides a more realistic network setting to evaluate the performance of the paths-finding algorithms by enabling the load to be more evenly distributed over all the links (Xiao and Ni 1999; Guo el al., 2004; Guo et al., 2005). The cost of each link is influenced by their respective residual bandwidth. Let W denote the link capacity, c_i the constant link cost (as shown in Figure 4.3) and R_i the residual bandwidth of link I, respectively, where $0 \leq R_i \leq W$. For an incoming call request, prior to computing the pair of primary and link-disjoint secondary paths, the link costs in the network are adjusted according to eq. (48) as shown,

$$c_i' = c_i + \beta \left(\frac{1}{R_i + \varepsilon} - \frac{1}{W} \right) \qquad (48)$$

where c_i' is the updated cost of link I, β a positive constant, and ε a positive constant with a very small value to avoid undefined link cost . In the simulation, we set $\beta = W = 16$, and ε=0.0001. It can be seen from eq. (48) that when the residual of a link reduces, the link cost increases. Therefore, such congested links will be avoided by the paths-finding algorithm in favour for other links with higher residual bandwidth.

Using the network with initial link cost as in Figure 4.3, Figure 4.9, Figure 4.10 and Figure 4.12 show the average total cost of AO, RAO, LB and RF at the network Erlang load of 40, 100 and 150 respectively with dynamic link costs corresponding to eq. (48). Figure 4.11 and Figure 4.13 depict the respective blocking probabilities at the network Erlang Load of 100 and 150. No call blocking was noticed at the Erlang Load of 40 for all values of α. The results in Figure 4.9 show that at a low load setting, the AO and RAO algorithms function as expected by ensuring minimum average total cost for all values of α. From Figure 4.10 and Figure 4.12, the change in the average total costs can be observed as the load increases. At a high load setting the average total costs are almost the same with the AO and RAO having a slightly lower average cost.

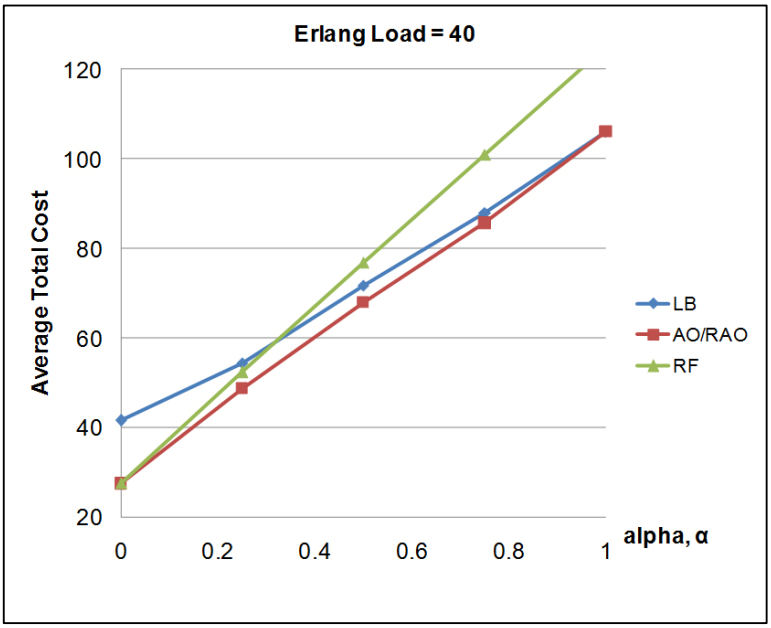

Figure 4.9: Average total cost at Erlang load = 40 (dynamic link cost)

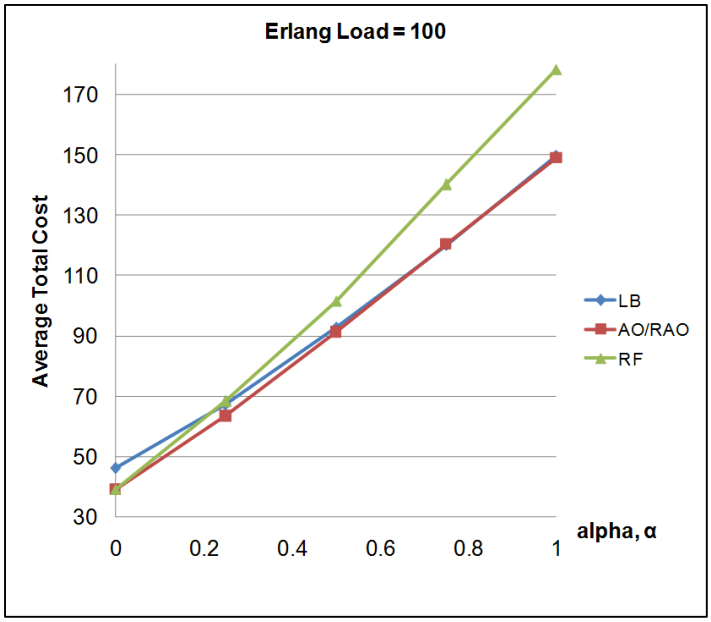

Figure 4.10: Average total cost at Erlang load=100 (dynamic link cost)

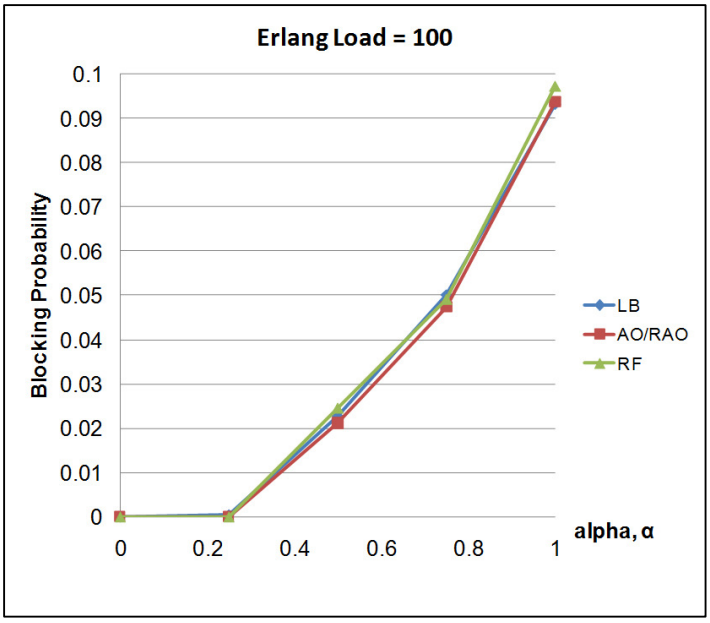

Figure 4.11: Blocking probability at Erlang load=100 (dynamic link cost)

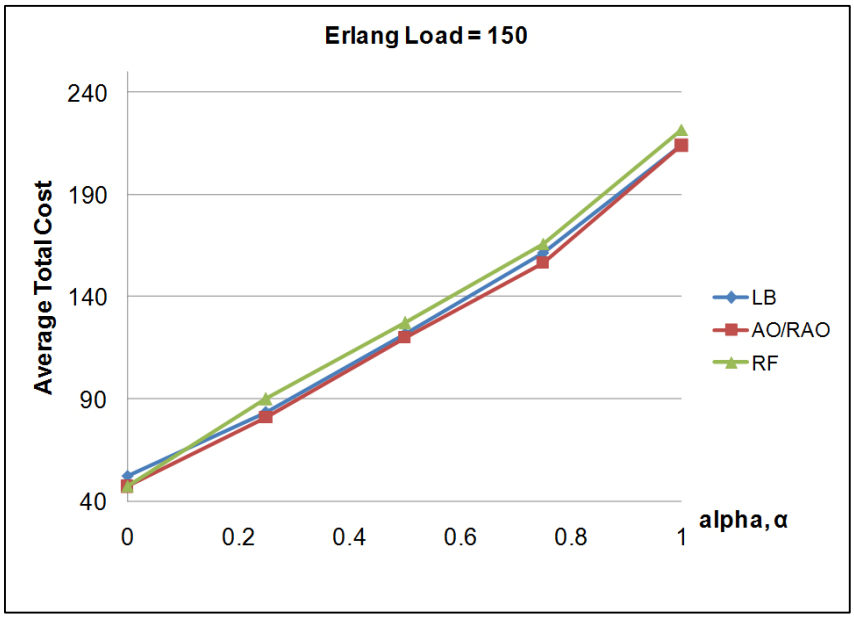

Figure 4.12: Average total cost at Erlang load=150 (dynamic link cost)

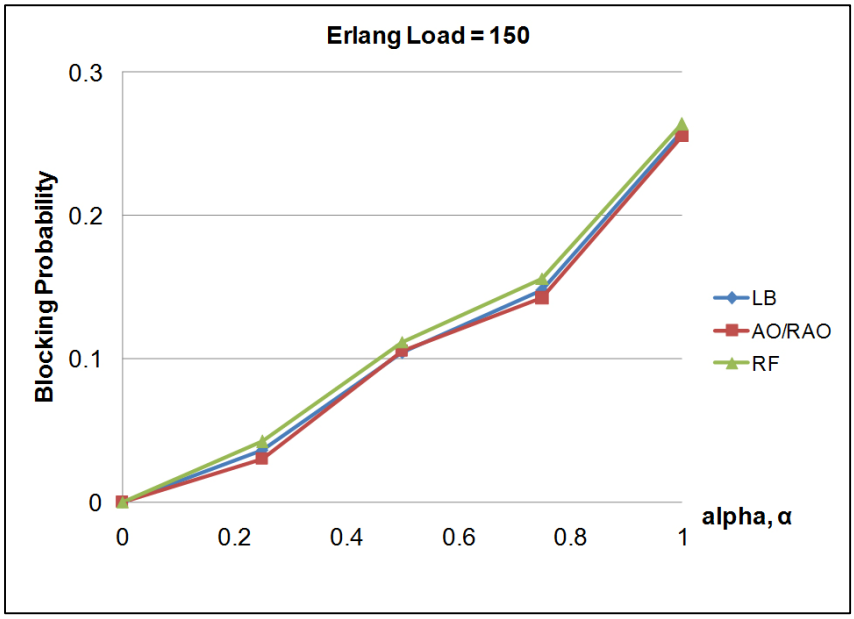

Figure 4.13: Blocking probability at Erlang load=150 (dynamic link cost)

It should be noted that in general the average total costs by the network with dynamic link cost are higher than the network with static link cost when compared against their respective Erlang loads. This is attributed to the consideration of dynamic link cost. The utilization of links increases the link costs. This leads to a higher average total cost. However, with dynamic link cost, the load is more evenly distributed over the network. By comparing the results of blocking probabilities, it is obvious that the network with dynamic-link cost achieves better performance by having an overall lower blocking probability.

4.4.2 Simulation over NSFnet

In this section, we present several supplementary simulation results (Gan and Liew 2010a; Gan and Liew 2010b) conducted at the initial stages of

this study, over the 14 node 21 bidirectional link NSFnet with the preset link weights as shown in Figure 4.14. The results are to show the robustness and consistency of our proposed algorithm under different network connectivity and network sizes.

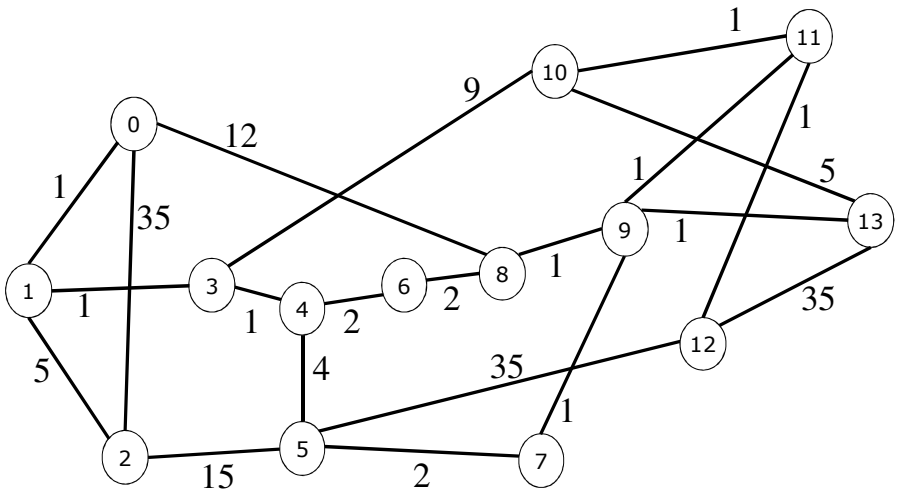

Figure 4.14: NSFnet with preset link weights

We assume W=16 unit bandwidth per link. Calls arrive in accordance to a Poisson process with rate λ and call duration is exponentially distributed with a mean of $1/\mu$. Therefore the Erlang load offered to the entire network is $\rho=\lambda/\mu$. A successful call reserves 1 unit of bandwidth along the primary path and α unit of bandwidth on the secondary path. Calls are monitored between node 1 and node 13. Each matching parameter of the results shows the computed average after 1 million calls.

The results in Figure 4.15 and Figure 4.16 show, at low Erlang load settings, the proposed AO and RAO algorithm ensures minimum average total

cost for all values of α when modeled over the NSFnet. The MO solution (noticeable between α=0.2 to α=0.8) are successfully identified by the proposed algorithm as well. There were no detected occurrences of call blocking.

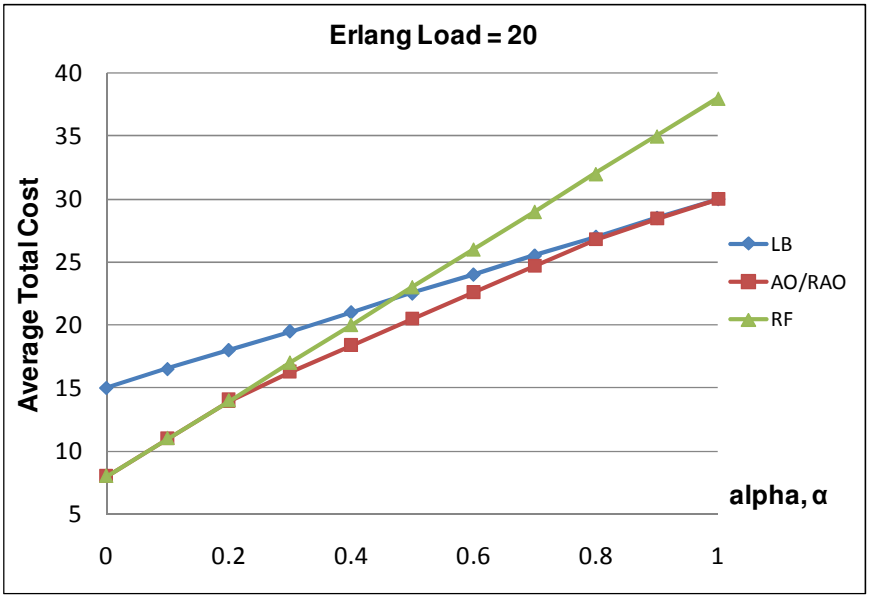

Figure 4.15 Average total cost at Erlang load = 20 (NSFnet)

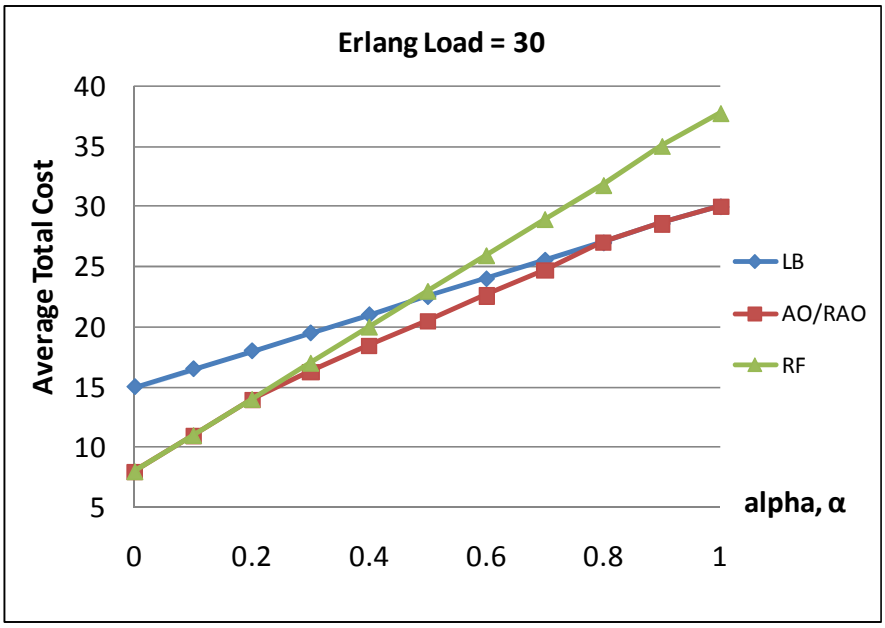

Figure 4.16 Average total cost at Erlang load = 30 (NSFnet)

Figure 4.17 and Figure 4.18, shows the average total cost of AO / RAO, LB, and RF and their respective blocking probability at high loads. Similarly, in a high load setting, the network would be busy servicing calls. And links along the optimal paths are at capacity. Hence, the cost of incoming calls would be higher because these calls are routed over the more expensive but still available set of links. At this congested state, the solutions by the paths-finding algorithms are also virtually the same limited by the network size and the number of available links. The high load setting notices the occurrence of call blocking as well.

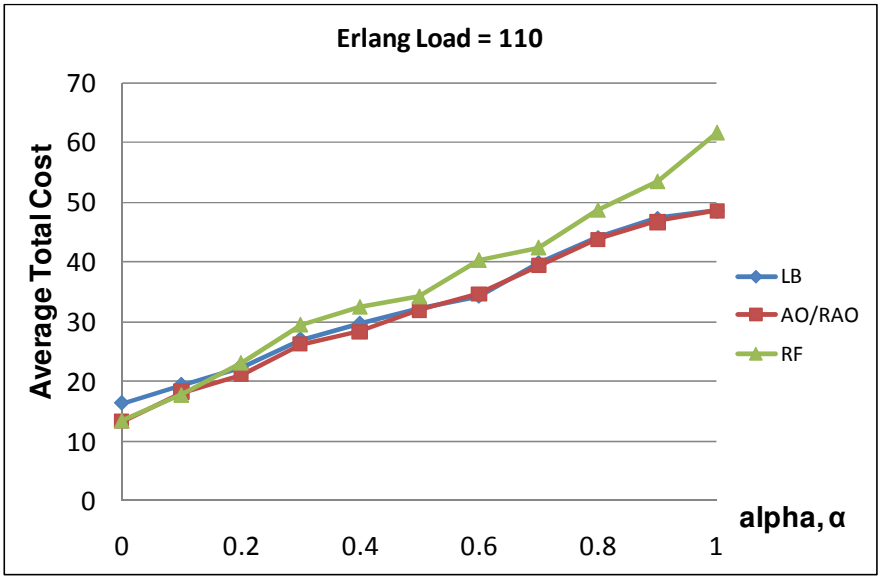

Figure 4.17 Average total cost at Erlang load = 110 (NSFnet)

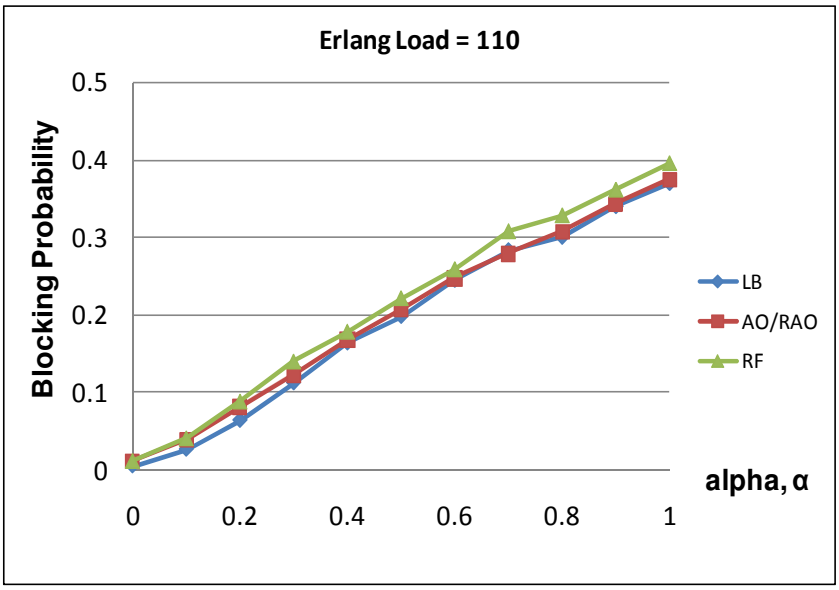

Figure 4.18 Blocking probability at Erlang load = 110 (NSFnet)

The results presented in this section verifies the consistency of our proposed algorithm under different network sizes and configurations to still effectively identify the optimal solution for the α+1 path protection scheme.

CHAPTER 5

THE TOGGLING DUAL COST (TDC) ALGORITHM

By looking at the optimum algorithms for the α+1 protection (Laborczi et al., 2001; Ho and Mouftah 2002; Fang et al., 2005; Yang et al., 2005a; Gan and Liew 2010a; Gan and Liew 2011), generally the *K*-shortest path algorithm is applied to perform an iterative search for the optimal solution. The *K*-shortest path algorithms commonly have high computation complexity, with the Yen's being O($KV(E+ V \log V)$) (Martins and Pascoal 2003), where *K* is the number of paths between the source and destination nodes, while *V* and *E* represent the numbers of nodes and edges in the graph, respectively. Thus, the *K*-shortest path algorithm increases significantly the overall complexity of the proposed paths-finding algorithms. Furthermore, in order to identify the optimal solution, the algorithm may potentially need to perform an exhaustive and time consuming search by evaluating all the likely candidate link-disjoint pair of paths.

To address this issue, an exact optimum link-disjoint paths-finding algorithm for α+1 protection without the application of the *K*-shortest path algorithm is proposed. The algorithm is labeled as the Toggling Dual Cost (TDC) algorithm (Liew and Gan 2012). The exactness of the TDC algorithm comes from the fact that the optimal solution is obtained through a more

efficient approach as opposed to the near brute-force method by the other algorithms mentioned. TDC yields a polynomial time complexity of $O(VE)$, to identify the optimal pair of link-disjoint paths for the α+1 protection.

5.1 Toggling Dual Cost – Example

In the following, the operation of our proposed TDC algorithm is illustrated with an example to effectively identify the optimum solution. With reference to the network given in Figure 5.1, suppose that a reliable communication is needed to be set up between the source node 'A' and destination node 'H', where the value on each link represents the cost (per unit bandwidth) incurred for data to traverse the particular link.

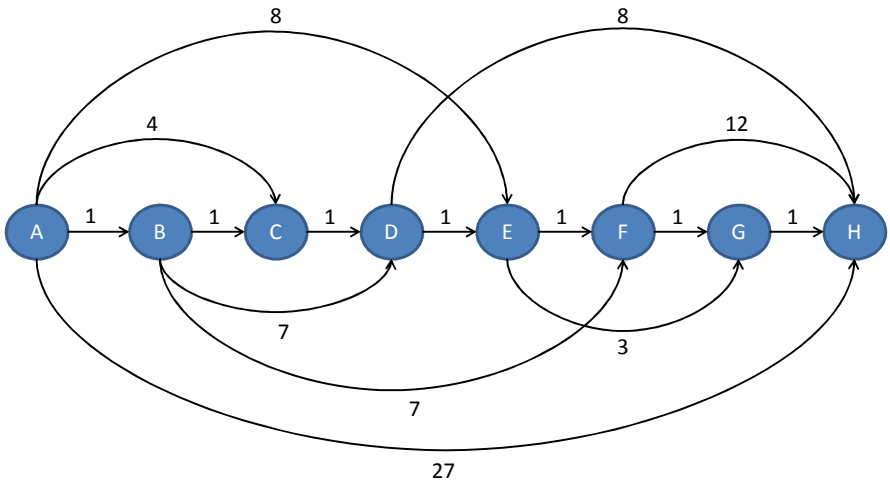

Figure 5.1: Network example

In general, α can be any value (pre-determined by users or network administrators) in between 0 and 1. However, different α value may result in a

different optimal solution (i.e., pairs of link-disjoint paths) that minimizes the cost given in eq. (1). Assume that $\alpha = 0.25$. Figure 5.2 shows the optimal link-disjoint solution (primary path: A-> C-> D-> E-> F-> G-> H, primary path cost = 4+1+1+1+1+1=9, secondary path: A-B-D-H, secondary path cost = 1+7+8 = 16) where the total cost, C, for this solution is $C(0.4)=9+(0.25)\times16=13$, following eq. (1). The challenge is how to identify such a solution at low time complexity.

The TDC algorithm comprises of two main steps. The first step requires the graph to be transformed according to the Suurballe graph transformation (Suurballe and Tarjan 1984). The second step runs a modified version of the Bellman-Ford algorithm on the transformed graph.

The details of the TDC algorithm are demonstrated as follows to find the exact solution given in Figure 5.2 (primary path: A-> C-> D-> E-> F-> G-> H, secondary path: A-> B-> D-> H).

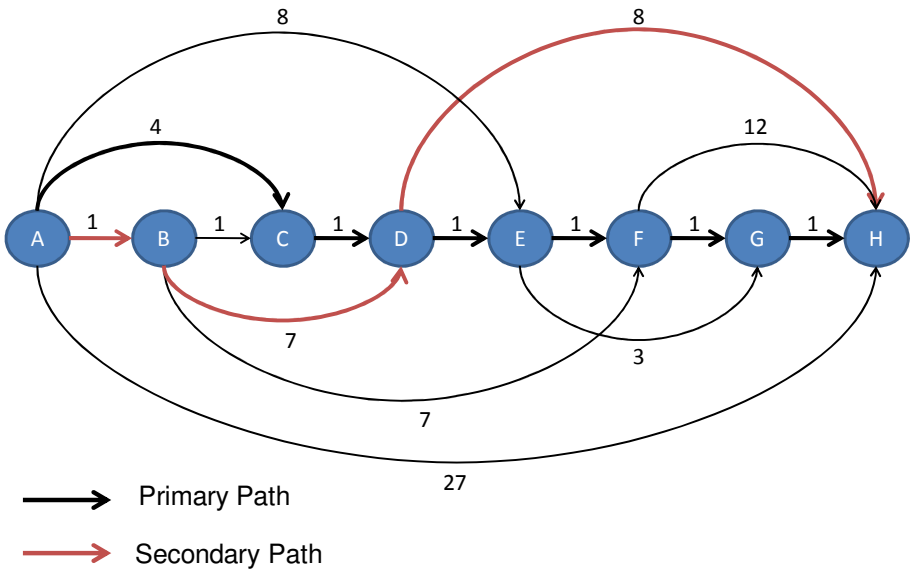

Figure 5.2: Optimal α +1 solution at α = 0.25

5.1.1 Step 1: Suurballe Graph Transformation

From Figure 5.1, the shortest path in this example is identified as A-> B-> C-> D-> E-> F-> G-> H, with a total path cost of 7. Applying the Suurballe graph transformation described in Chapter 3.3.3, Figure 5.3 shows the graph after transformation with the link-cost of every edge recomputed according to eq.(2).

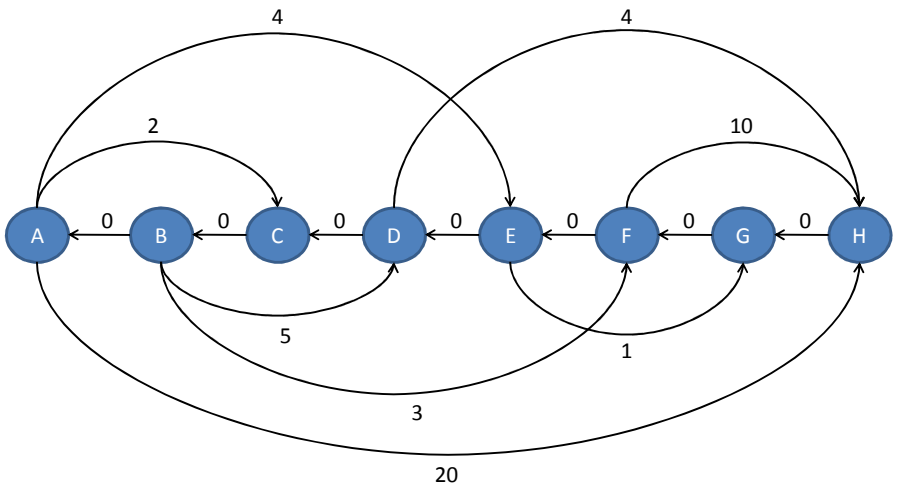

Figure 5.3: Suurballe transformed network

Given a specified source and destination node, it has been illustrated in Suurballe and Tarjan (1984) that any path found in the Suurballe transformed graph could, together with the shortest path in the original graph, be translated as a link-disjoint path-pair in the original graph. For instance, the shortest path found in the Suurballe transformed graph (e.g., A-> C-> B-> D-> H in Figure 5.3) could be coupled with the shortest path in the original graph (e.g., A-> B-> C-> D-> E-> F-> G-> H in Figure 5.1); and by removing the common links in the reverse direction (e.g., C-B in Figure 5.3 and B-C in Figure 5.1) they can form a pair of link-disjoint paths which is the optimal solution in 1+1 protection (Suurballe and Tarjan 1984).

By the same token, a specific path in the Suurballe transformed graph, with respect to a defined α value, could be identified by another algorithm; whereby the path could subsequently be translated to the optimal pair of link-

disjoint paths for the α+1 protection in the original graph. The challenge is the derivation of such an exact algorithm for the above objective.

Observation

In order to explain the TDC approach, the optimal solution of Figure 5.2 with respect to α = 0.25, can in fact be reflected by the Suurballe transformed graph as a single path shown in Figure 5.4 (path: A-> C-> B-> D-> H). Cost computation of this path is however slightly different. The path is initially divided into segments. Each segment spans from one shortest path node to another shortest path node. The path in Figure 5.4 has four segments (segment 1: A-C, segment 2: C-B, segment 3: B-D, segment 4: D-H).

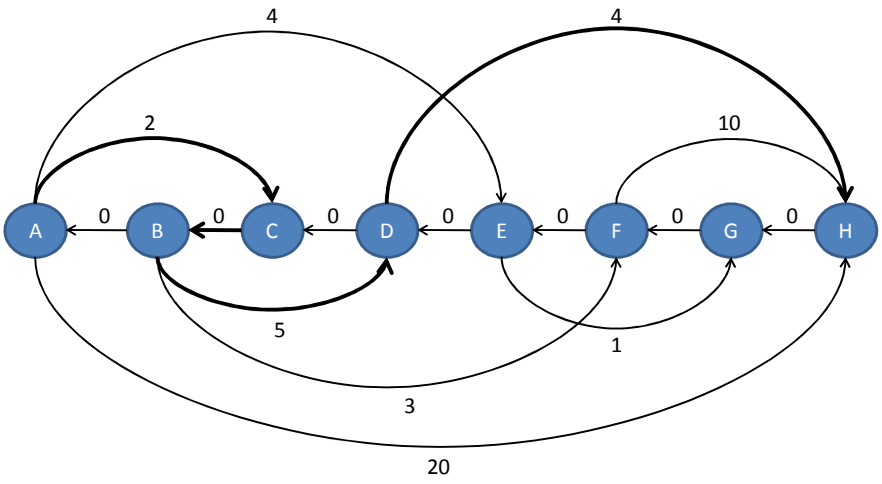

Figure 5.4: Optimal α +1 solution at α = 0.25 in Suurballe transformed network

It is observed that the non-zero-cost path segments in the Suurballe transformed graph is directly associated to either the primary or secondary path in the original graph. In other words, the cost from any one of this segment affects either the primary or secondary path cost. For the 1+1

protection, this observation is not an issue as both the primary and secondary paths will contribute equally to the total cost computation. But in the α+1 protection, the primary-secondary path cost are asymmetrically weighted. Therefore the segments belonging to the secondary path must be multiplied with α. The key is determining whether a particular segment belongs to the primary or secondary path.

Therefore, a segment can be of either one of the two groups. For segments in the first group, belonging to that of the primary path, the cost of each segments are summed together. In the second group, that of the secondary path, the cost-segments are directly sum together and multiplied by α. Cost computation for the path in the Suurballe transformed network comprise the summation of both the first and second group's cost. The segments which traverse the reverse shortest path link (zero-cost segments) do not belong to any group but it triggers a toggle on the subsequent segment's grouping.

Referring to the path from the graph shown in Figure 5.4 (with the reverse shortest path: H-> G-> F-> E-> D-> C-> B-> A), the path has four segments (segment 1: A-C, segment 2: C-B, segment 3: B-D, segment 4: D-H). Segment 1 could begin from either the first group or the second group. Both possibilities would produce a candidate for the optimal solution. Since this is the case, both of these options must be evaluated (the proposed TDC algorithm utilizes a dual label system to account for all these possibilities. Further details on the dual label system will be discussed later).

However to demonstrate rather, how the path cost is computed in the Suurballe transform network, the following example assumes that we have prior knowledge on the grouping of segment 1 for the path in Figure 5.4. Therefore the example begins with segment 1 being on the first group. Note that the other possibility (where segment 1 begins on the second group) would also be considered by the TDC dual label system. Segment 2 traverses the reverse shortest path link and therefore triggers a toggle for the subsequent segment. As a result, segment 3 goes to the second group.

The group placement of segment 4 requires additional consideration. This is because segment 3 and segment 4 are successive non-zero segments. In this particular occurrence, the grouping of segment 4 to either the first or second group is valid (similar to the situation described in the paragraph above, the grouping of segment 4 to either the first or second group would each result in yet another candidate optimum solution respectively. The dual label system in the proposed TDC algorithm also accounts for this occurrence). For this example, segment 4 is placed in the second group. Thus at the end, the first group consist of segment 1 while the second group has segment 3 and segment 4.

The cost sum of the first group, denoted as Primary Path Cost Sum (PPCS), which has segment 1 (cost-segment =2) is PPCS= Σ cost-segments = 2. The cost sum of the second group, denoted as Secondary Path Cost Sum (SPCS), with segment 3 (cost-segment = 5) and segment 4 (cost-segment = 4)

is SPCS=α * Σ cost-segments= 0.25 * (5+4) = 2.25. So total costs for the path in the Suurballe transformed graph is PPCS + SPCS = 2 + 2.25 = 4.25.

5.1.2 Step 2: Modified Bellman-Ford

In order to identify the path in the Suurballe transformed graph (for producing the optimal link-disjoint path-pair in the original graph) as discussed in the Observation section, the TDC algorithm applies a modified Bellman-Ford algorithm on the Suurballe transformed graph. Figure 5.5 depicts the network after the Suurballe graph transformation together with a table listing the path routes (Current Identified Paths column) for better visualization on the algorithm's operations. Each listed path route has two labels. The labels are denoted as path cost 'Path Cost Estimate 1' and 'Path Cost Estimate 2' respectively. The two labels function to indicate the costs from the source node to the present node, with the segment under consideration belonging to the primary path (Path Cost Estimate 1) or the secondary path (Path Cost Estimate 2).

The modified Bellman Ford algorithm requires the use of two labels because as mentioned in the Observation section, there are certain segments in the Suurballe transformed graph (such as the first segment and successive non-zero segments of a particular path) could be part of the primary path or the secondary path in the original graph. The appropriate grouping of segments for the path identified in the Suurballe transformed graph would produce the optimal link-disjoint path solution in the original graph. Therefore the two

labels by the modified Bellman Ford algorithm are to account for all possibilities of the path segment grouping in the Suurballe transformed graph in order to ensure the eventual path obtained by the modified Bellman Ford algorithm produces the optimal link-disjoint path in the original graph.

a) Initialization

At initialization, the source node (node 'A') at the preliminary is only aware of the route to itself and thus its path cost is set to zero. No other paths from the source node to the other nodes in the graph have been identified yet.

Current Identified Paths	Path Cost Estimate 1	Path Cost Estimate 2
A -> A	0	0

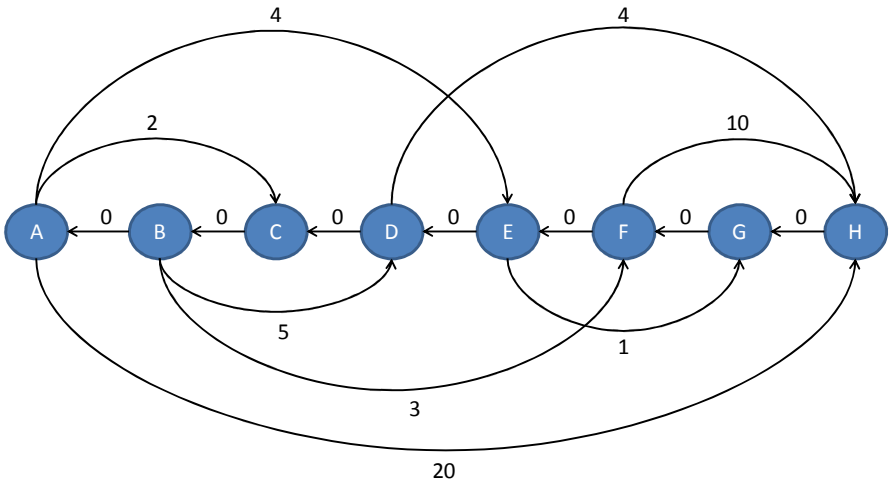

Figure 5.5: Modified Bellman Ford – Initialization

The path route list (Current Identified Paths column) is updated during the relaxation of segments. The relaxation process is adapted from the Bellman-Ford relaxation. When a segment is being relaxed, path routes in the

path list in which the end node matches the start node of the relaxed segment will be selected. Both the path route and path cost will be updated with regards to the relaxed segment. To update the Path Cost Estimate 1 column field, the current value of the column field is summed with the cost-segment. Similarly, the Path Cost Estimate 2 column field is updated by summing the current value of the column field, with the cost-segment multiplied by α.

b) Relaxation Cycle 1

In the first round of the relaxation cycle, the links connecting node 'A' to adjacent nodes (link A-C, link A-E and link A-H) are relaxed. Illustrated in Figure 5.6, the path route list and their respective path cost are updated accordingly. Updated fields are in red. In a particular relaxation cycle, the order in which the links are relaxed is randomly pre-determined. For this example, the relaxation sequence starts with the links from node 'A' to its adjacent nodes. This is followed by the respective links from node 'B', node 'C, node 'D', node 'E', node 'F' and node 'G'. No relaxation of links from node 'H' is performed because it is the destination node. Subsequent discussions only highlight the crucial steps involved in the execution of the modified Bellman-Ford relaxation cycle.

Current Identified Paths	Path Cost Estimate 1	Path Cost Estimate 2
A -> C	2	0.5
A -> E	4	1
A -> H	20	5

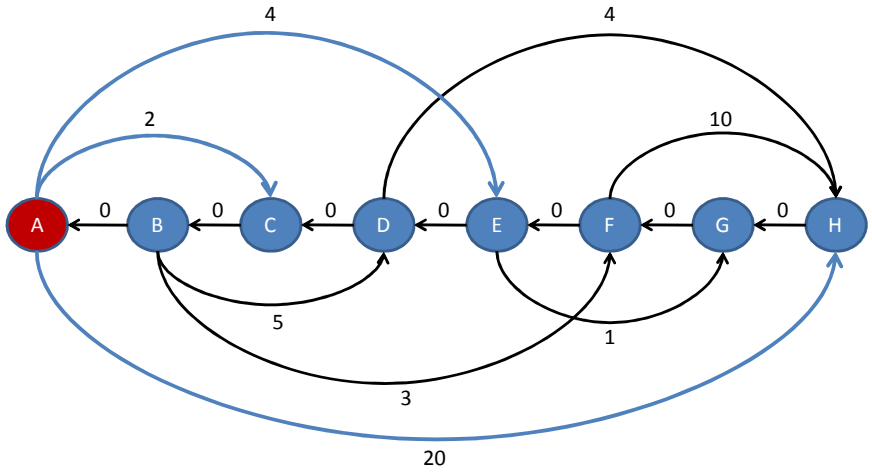

Figure 5.6: Relaxation cycle 1- Link relaxation from node 'A'

No changes occur in the relaxation of links from node 'B'. Relaxing the links from node 'C' updates the path route list as in Figure 5.7. The updated path route is flagged (denoted with '*') because the segment that was relaxed affecting the particular path route traverses a reversed shortest path link. The flagged path route will trigger a toggle which modifies the path cost computation of the next segment.

Current Identified Paths	Path Cost Estimate 1	Path Cost Estimate 2
A -> C -> B *	2	0.5
A -> E	4	1
A -> H	20	5

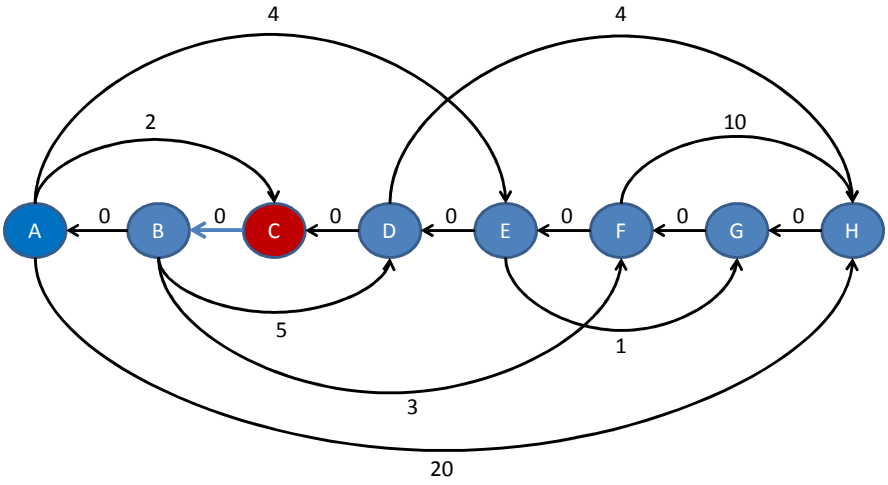

Figure 5.7: Relaxation cycle 1- Link relaxation from node 'C'

The relaxations of links from the remaining nodes with node 'G' as the last node in the relaxation cycle result in the updated table as illustrated in Figure 5.8. The relaxation cycle is to run for *V*-1 rounds (where *V* is the number of nodes in the graph) as stipulated by the Bellman-Ford algorithm.

Current Identified Paths	Path Cost Estimate 1	Path Cost Estimate 2
A -> C -> B *	2	0.5
A -> E -> D *	4	1
A -> E -> G -> F*	2	1.25
A -> H	20	5

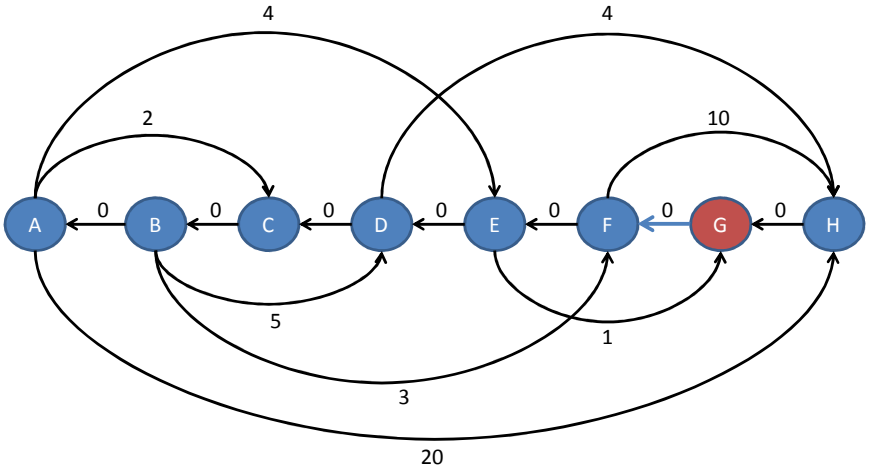

Figure 5.8: Relaxation cycle 1- Link relaxation from node 'G'

c) Relaxation Cycle 2

The second round in the relaxation cycle, there were no changes observed from the relaxation of links from node 'A'. Moving on to the relaxation of links from node 'B', the path route A -> C -> B will be affected when segments B -> D and B -> F are relaxed. Given that this path route (A -> C -> B) is flagged, it thus triggers a toggle before the link relaxations.

To emulate a toggle, the affected path route's corresponding costs namely will exchange respective values with each other as illustrated in Figure 5.9. Subsequently, the exchanged values are used in the relaxation process.

Figure 5.10 shows the ensuing updated table after executing the relaxation of the associated segments.

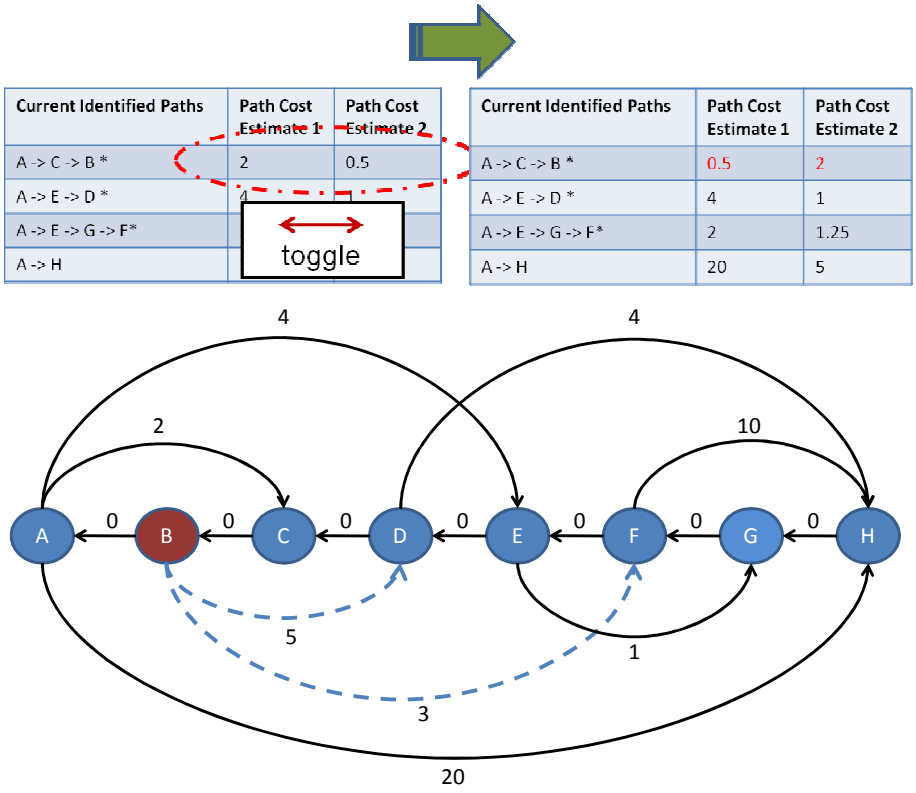

Figure 5.9: Relaxation cycle 2 – Toggling process before relaxation of links from node 'B'

Current Identified Paths	Path Cost Estimate 1	Path Cost Estimate 2
A -> C -> B -> D	5.5	3.25
A -> C -> B -> F	3.5	2.75
A -> E -> D *	4	1
A -> E -> G -> F*	2	1.25
A -> H	20	5

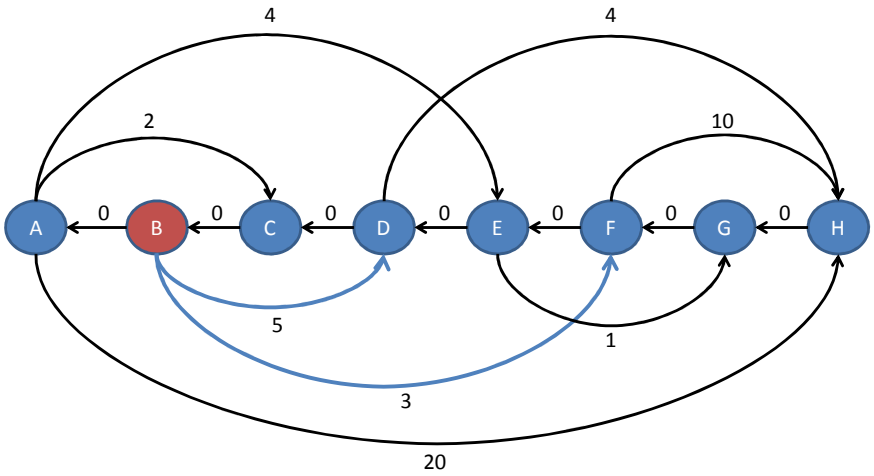

Figure 5.10: Relaxation cycle 2 – Links relaxation from node 'B'

Progressing further with the relaxation cycles until the stop of the modified Bellman-Ford algorithm yields the resulting path route from the source node 'A' to destination node 'H' with the respective path cost as shown in Figure 5.11.

Current Identified Paths	Path Cost Estimate 1	Path Cost Estimate 2
A -> C -> B -> D -> H	7.25	4.25

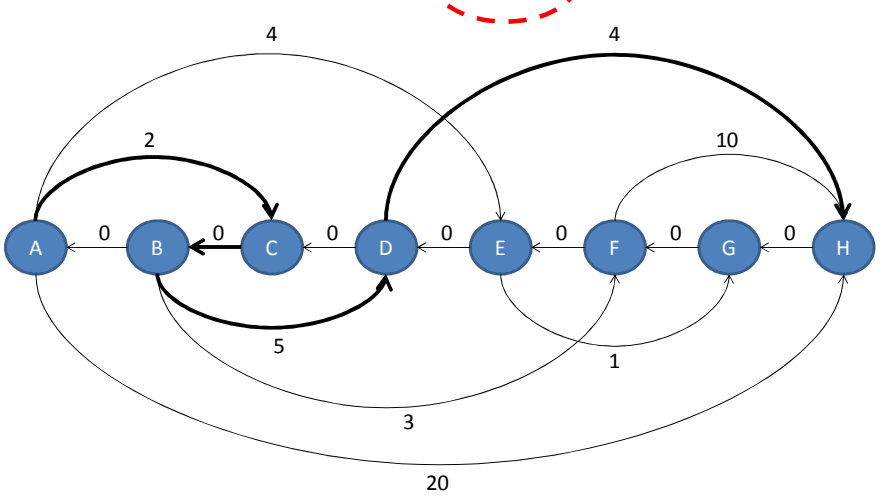

Figure 5.11: Final graph at the stop of the modified Bellman-Ford algorithm

The final table result in Figure 5.11 indicates the optimal path (identified by the TDC at α=0.25) in the transformed graph between source node 'A' to destination node 'H' (path: A-> C-> B-> D-> H) has a path cost of 4.25. The result is the same as the expected optimal path cost calculated earlier.

Lastly, the path found in the Suurballe transformed graph forms an optimal pair of link-disjoint path in the original graph for the α+1 protection as in Figure 5.2. A detail description of the TDC algorithm is presented in the next section.

5.2 Toggling Dual Cost – Algorithm Description

The TDC algorithm has been demonstrated to successfully determine the optimal path for the α+1 protection from the network example discussed in the previous section. The specified α value that was used in the example is in fact determined from another MO property.

It is observed that the MO solution happens to be optimal in comparison to the RF and LB solution when

$$\frac{P_{MO} - P_{RF}}{S_{RF} - S_{MO}} < \alpha < \frac{P_{LB} - P_{MO}}{S_{MO} - S_{LB}} \qquad (49)$$

This property is proved in the following corollary

Corollary 3: If there exists a MO solution with primary and secondary costs P' and S', respectively, for a given network graph, then $\alpha = y = (P_{LB} - P_{RF})/(S_{RF} - S_{LB})$ is one of the points where the MO solution has a lower total cost than that of RF and LB solution, C^*.

Proof:

Since the given primary and secondary paths are a MO solution, then inequality (21) holds true. We have

$$(P' - P_{RF})(S' - S_{LB}) < (P_{LB} - P')(S_{RF} - S') \qquad (50)$$

$$\Rightarrow P'S' - P_{RF}S' - P'S_{LB} + P_{RF}S_{LB} < P_{LB}S_{RF} - P'S_{RF} - P_{LB}S' + P'S'$$

$$\Rightarrow P'S_{RF} - P'S_{LB} + P_{LB}S' - P_{RF}S' < P_{LB}S_{RF} - P_{RF}S_{LB}$$

$$\Rightarrow P'(S_{RF} - S_{LB}) + S'(P_{LB} - P_{RF}) < P_{LB}S_{RF} - P_{RF}S_{LB}$$

$$\Rightarrow P' + \frac{P_{LB} - P_{RF}}{S_{RF} - S_{LB}} S' < \frac{P_{LB} S_{RF} - P_{RF} S_{LB}}{S_{RF} - S_{LB}} \qquad (51)$$

Combining eq. (51) with eq. (5) and (6), we have

$$P' + yS' < C^* \qquad (52)$$

Q. E. D.

By using the properties derived in, Theorem 1(iii), Corollary 1(ii), and Corollary 3, the various critical α values are identified for the TDC algorithm. If an optimal solution exists at the given α value, the TDC algorithm would identify its pair of link-disjoint paths.

Figure 5.12 and Figure 5.13 presents the TDC algorithm in detail. As mentioned, the TDC algorithm has a complexity of O(*VE*). The complexity arrives from the Bellman Ford algorithm which was modified in the TDC algorithm to suite the search criteria. The modifications however do not affect the overall complexity.

Algorithm TDC (Main) – Modified Bellman Ford

{ Definition

V: The set of nodes in the network

E: The set of directed edges in the network

$Y(i)$: Total number of paths found in cycle i

s: Source node

d: Destination node

$d_{(i,y)}[u]$: Path Cost Estimate 1 – Length of the path y at loop i from s to node u

$d_{\alpha(i,y)}[u]$: Path Cost Estimate 2 – Length of the path y at loop i from s to node u

$d_{op}[u]$: Current length of the optimal path from source to destination from Path Cost Estimate 1

$d_{\alpha op}[u]$: Current length of the optimal path from source to destination from Path Cost Estimate 2

$\pi_{(i,y)}[u]$: Current predecessor node of u

$\pi_{op}[u]$: Optimal path route from source to destination from Path Cost Estimate 1

$\pi_{\alpha op}[u]$: Optimal path route from source to destination from Path Cost Estimate 2

$\mathbf{V}_{P1}$': Set of nodes that are traversed by the reversed shortest path

$\mathbf{P_1}$': The reversed shortest path from the original graph

$f_{(i,y)}[u]$: A flag to trace whether the link towards node u traverse $\mathbf{P_1}$'

$m_{(i,u)}$: Counter to calculate links out of node u which has been relaxed

$w(u, v)$: Weight of edge(u,v) where edge$(u,v) \in \boldsymbol{E}$

$n'_{(i,y)}$: End node in path $\pi_{(i,y)}$

$l[u]$: Number of edges out of node u}

Modify the graph according to Suurballe transformation (Suurballe and Tarjan 1984)

for $i \leftarrow 1$ to $|V| - 1$ **do**
    for $u \in \boldsymbol{V}$ **do**
        $m_{(i,u)} \leftarrow 0$
    end for
end for

for $i \leftarrow 1$ to $|V| - 1$ **do**
    $y \leftarrow 0$
    $Y(i) \leftarrow 0$
    for each edge $(u, v) \in \boldsymbol{E}$ **do**
        Find $l[u]$, number of edges out of node u
        Call relax(u, v)
    end for
end for

Transform optimal path to link-disjoint path in the original graph

Figure 5.12: Toggling Dual-Cost – Main Program

Procedure Link Relaxation

procedure relax(u, v)

if i=1 AND u=s **then**

$d_{(i,y)}[u] \leftarrow 0$

$d_{\alpha(i,y)}[u] \leftarrow 0$

$\pi_{(i,y)}[s] \leftarrow s$

$d_{op}[d] \leftarrow \infty$

$d_{\alpha op}[d] \leftarrow \infty$

$d_{(i,y)}[v] \leftarrow d_{(i,y)}[u] + w(u, v)$

$d_{\alpha(i,y)}[v] \leftarrow d_{\alpha(i,y)}[u] + \alpha * w(u, v)$

$\pi_{(i,y)}[v] \leftarrow u$

$f_{(i,y)}[v] \leftarrow 0$

y++

$Y(i) \leftarrow y$

if v = d **then**

if $d_{(i,y)}[v] < d_{op}[v]$

$d_{op}[v] \leftarrow d_{(i,y)}[v]$

for $n \in V$ **do**

$\pi_{op}[n] \leftarrow \pi_{(i,y)}[n]$ {transfer of entire path}

end for

end if

if $d_{\alpha(i,y)}[v] < d_{\alpha op}[v]$ **then**

$d_{\alpha op}[v] \leftarrow d_{\alpha(i,y)}[v]$

for $n \in V$ **do**

$\pi_{\alpha op}[n] \leftarrow \pi_{(i,y)}[n]$ {transfer of entire path}

end for

end if

end if

else if i=1 AND u!=s

for $y \leftarrow 0$ to $Y(i)$

if $n'_{(i,y)} = u$ {check the last node in the path}

if $\pi_{(i,y)}[v] \leftarrow u$ is loopless **then**

if $l[u]$-$m_{(i,u)} > 1$ **then**

{check for multiple branches out of node }

$Y(i)$++

if $f_{(i,y)}[u]$=1 and edge(u,v) $\notin$ $\mathbf{P_1}$**'then**

{consideration of toggling}

$d_{(i,Y(i))}[v] \leftarrow d_{\alpha(i,y)}[u] + w(u, v)$

$d_{\alpha(i,Y(i))}[v] \leftarrow d_{(i,y)}[u] + \alpha * w(u, v)$

else if $u \in \mathbf{V}_{P1'}$ and $f_{(i,y)}[u]$=0 and

edge(u,v) $\notin$ $\mathbf{P_1}$**' then**

{***to consider special case ***}

$d_{(i,Y(i))}[v] \leftarrow d_{\alpha(i,y)}[u] + w(u, v)$

$d_{\alpha(i,Y(i))}[v] \leftarrow d_{\alpha(i,y)}[u] + \alpha * w(u, v)$

else

$d_{(i,Y(i))}[v] \leftarrow d_{(i,y)}[u] + w(u, v)$

$d_{\alpha(i,Y(i))}[v] \leftarrow d_{\alpha(i,y)}[u] + \alpha * w(u, v)$

end if

$m_{(i,u)}$++

else

$Y(i)$++

if $f_{(i,y)}[u]$=1 and edge(u,v) $\notin$ $\mathbf{P_1}$**'then**

{consideration of toggling}

$d_{(i,Y(i))}[v] \leftarrow d_{\alpha(i,y)}[u] + w(u, v)$

$d_{\alpha(i,Y(i))}[v] \leftarrow d_{(i,y)}[u] + \alpha * w(u, v)$

else if $u \in \mathbf{V}_{P1'}$ and $f_{(i,y)}[u]$=0 and

edge(u,v) $\notin$ $\mathbf{P_1}$**' then**

{***to consider special case ***}

$d_{(i,Y(i))}[v] \leftarrow d_{\alpha(i,y)}[u] + w(u, v)$

$d_{\alpha(i,Y(i))}[v] \leftarrow d_{\alpha(i,y)}[u] + \alpha * w(u, v)$

else

$d_{(i,Y(i))}[v] \leftarrow d_{(i,y)}[u] + w(u, v)$

$d_{\alpha(i,Y(i))}[v] \leftarrow d_{\alpha(i,y)}[u] + \alpha * w(u, v)$

end if

end if

for $n \in V$ **do**

$\pi_{(i,Y(i))}[n] \leftarrow \pi_{(i,y)}[n]$ {transfer of entire path}

```
                                    end for
                                    π(i,Y(i))[v]←u
                                    if v = d then
                                        if d(i,y)[v] < dop[v]
                                            dop[v]← d(i,y)[v]
                                            for n ∈ V do
                                                πop[n]← π(i,y)[n]
                                                {transfer of entire path}
                                            end for
                                        end if
                                        if dα(i,y)[v] < dαop[v] then
                                            dαop[v]← dα(i,y)[v]
                                            for n ∈ V do
                                                παop[n]← π(i,y)[n]
                                                {transfer of entire path}
                                            end for
                                        end if
                                    end if
                                    if edge(u,v) ∈ P1'
                                        f(i,Y(i))[v] ← 1
                                    end if
                                end if
                            end if
                        end for

else if i >1
        for y ←0 to Y(i-1)
            if n'(i-1,y) = u {check the last node in the path}
                if π(i-1,y)[v]←u is loopless then
                    if l[u]-m(i,u) > 1 then
                        {check for multiple branches out of node }
                        Y(i)++
                        Y(i-1)++
                        if f(i-1,y)[u]=1 and edge(u,v) ∉ P1' then
                                {consideration of toggling}
                            d(i-1,Y(i-1))[v]←dα(i-1,y)[u] + w(u, v)
```

$d_{\alpha(i-1,Y(i-1))}[v] \leftarrow d_{(i-1,y)}[u] + \alpha^* w(u, v)$

$d_{(i,Y(i))}[v] \leftarrow d_{\alpha(i-1,y)}[u] + w(u, v)$

$d_{\alpha(i,Y(i))}[v] \leftarrow d_{(i-1,y)}[u] + \alpha^* w(u, v)$

else if $u \in V_{P1'}$ and $f_{(i,y)}[u]=0$ and

edge(u,v) $\notin$ **P_1' then**

{***to consider special case ***}

$d_{(i-1,Y(i-1))}[v] \leftarrow d_{\alpha(i-1,y)}[u] + w(u, v)$

$d_{\alpha(i-1,Y(i-1))}[v] \leftarrow d_{\alpha(i-1,y)}[u] + \alpha^* w(u, v)$

$d_{(i,Y(i))}[v] \leftarrow d_{\alpha(i-1,y)}[u] + w(u, v)$

$d_{\alpha(i,Y(i))}[v] \leftarrow d_{\alpha(i-1,y)}[u] + \alpha^* w(u, v)$

else

$d_{(i-1,Y(i-1))}[v] \leftarrow d_{(i-1,y)}[u] + w(u, v)$

$d_{\alpha(i-1,Y(i-1))}[v] \leftarrow d_{\alpha(i-1,y)}[u] + \alpha^* w(u, v)$

$d_{(i,Y(i))}[v] \leftarrow d_{(i-1,y)}[u] + w(u, v)$

$d_{\alpha(i,Y(i))}[v] \leftarrow d_{\alpha(i-1,y)}[u] + \alpha^* w(u, v)$

end if

$m_{(i,u)}$++

for $n \in V$ **do**

$\pi_{(i-1,Y(i-1))}[n] \leftarrow \pi_{(i-1,y)}[n]$

{transfer of entire path}

end for

$\pi_{(i-1,Y(i-1))}[v] \leftarrow u$

if v = d **then**

if $d_{(i-1,y)}[v] < d_{op}[v]$

$d_{op}[v] \leftarrow d_{(i-1,y)}[v]$

for $n \in V$ **do**

$\pi_{op}[n] \leftarrow \pi_{(i-1,y)}[n]$

{transfer of entire path}

end for

end if

if $d_{\alpha(i-1,y)}[v] < d_{\alpha op}[v]$ **then**

$d_{\alpha op}[v] \leftarrow d_{\alpha(i-1,y)}[v]$

for $n \in V$ **do**

$\pi_{\alpha op}[n] \leftarrow \pi_{(i-1,y)}[n]$

{transfer of entire path}

end for

end if

end if

if edge(u,v) ∈ $\mathbf{P_1}$'

$f_{(i-1,Y(i-1))}[v] \leftarrow 1$

end if

else

Y(i)++

if $f_{(i,y)}[u]=1$ and edge(u,v) ∉ $\mathbf{P_1}$'**then**

{consideration of toggling}

$d_{(i,Y(i))}[v] \leftarrow d_{\alpha(i,y)}[u] + w(u, v)$

$d_{\alpha(i,Y(i))}[v] \leftarrow d_{(i,y)}[u] + \alpha * w(u, v)$

else if $u \in \mathbf{V}_{P1'}$ and $f_{(i,y)}[u]=0$ and

edge(u,v) ∉ $\mathbf{P_1}$' **then**

{***to consider special case ***}

$d_{(i,Y(i))}[v] \leftarrow d_{\alpha(i,y)}[u] + w(u, v)$

$d_{\alpha(i,Y(i))}[v] \leftarrow d_{\alpha(i,y)}[u] + \alpha * w(u, v)$

else

$d_{(i,Y(i))}[v] \leftarrow d_{(i,y)}[u] + w(u, v)$

$d_{\alpha(i,Y(i))}[v] \leftarrow d_{\alpha(i,y)}[u] + \alpha * w(u, v)$

end if

end if

for $n \in V$ **do**

$\pi_{(i,Y(i))}[n] \leftarrow \pi_{(i-1,y)}[n]$ {transfer of entire path}

end for

$\pi_{(i,Y(i))}[v] \leftarrow u$

if $v = d$ **then**

if $d_{(i,y)}[v] < d_{op}[v]$

$d_{op}[v] \leftarrow d_{(i,y)}[v]$

for $n \in V$ **do**

$\pi_{op}[n] \leftarrow \pi_{(i-1,y)}[n]$

{transfer of entire path}

end for

end if

if $d_{\alpha(i,y)}[v] < d_{\alpha op}[v]$ **then**

```
                                        d_αop[v]← d_α(i,y)[v]
                                        for n ∈ V do
                                            π_αop[n]← π_(i-1,y)[n]
                                            {transfer of entire path}
                                        end for
                                    end if
                                end if
                                if edge(u,v) ∈ P1'
                                    f_(i,Y(i))[v] ← 1
                                end if
                            end if
                        end if
                    end for
end if
```

Figure 5.13: Toggling Dual-Cost – Link Relaxation Procedure

5.3 Correctness of the TDC Algorithm

The proof takes into account the characteristics of the Suurballe's link-weight modification to verify that the solution by the TDC algorithm reflects a valid link-disjoint path-pair solution for the α+1 protection in the original graph. It is derived in Suurballe and Tarjan (1984) that the paths order in the Suurballe reweighted (only the link-costs are altered without changes to the link's direction) graph are the same as that in the original graph. This implies that the Suurballe reweighted graph preserves all properties of the original graph except changes to the link costs.

Moreover, given a source and destination node where the source node is also the root node of the shortest-path tree in the graph, the cost of path $\mathbf{P_{SB}'}$

in the Suurballe reweighted graph, denoted as P_{SB}', is related to P, which is the cost of path **P** in the original graph with the following expression (Suurballe and Tarjan 1984);

$$P = P_{SB}' + P_1 \tag{53}$$

where P_1 is the cost of the shortest path, denoted as $\mathbf{P_1}$, in the original graph.

Theorem 2 derives the relation between the optimal path, $\mathbf{P_0'}$, in the Suurballe transformed (modifications to the link-cost and link-direction) graph by the TDC algorithm with the optimal link-disjoint paths-pair in the original graph.

Theorem 2: The cost of path $\mathbf{P_0'}$, found in the Suurballe transformed graph by the TDC algorithm, denoted as P_0', is related to P and S, which are the costs of the link-disjoint pair of paths **P** and **S** in the original graph with the following property;

$$P_0' + (1+\alpha)P_1 = P + \alpha S \tag{54}$$

Proof:

It can be easily verify that

$$P_0' = P_{SB}' + \alpha S_{SB}' \tag{55}$$

where P_{SB}' and S_{SB}' are the cost of path $\mathbf{P_{SB}'}$ and $\mathbf{S_{SB}'}$ respectively. Paths $\mathbf{P_{SB}'}$ and $\mathbf{S_{SB}'}$ are the link-disjoint path in the Suurballe reweighted graph translated from $\mathbf{P_0'}$ as described in Suurballe and Tarjan (1984)

We expand (55) to become

$$P_0'+(1+\alpha)P_1 = (P_{SB}'+P_1)+\alpha(S_{SB}'+P_1) \qquad (56)$$

Applying (53) into (56) yields $P_0'+(1+\alpha)P_1 = P+\alpha S$

Q. E. D.

5.4 Complexity and Performance Comparison

Table 5.1 list the complexity of the TDC with other existing partial bandwidth protection, optimum path-finding algorithms. In terms of complexity, it is observed that the TDC exhibits a high efficiency over the other algorithms.

Table 5.1: Algorithm Complexity Comparison

Optimal Algorithm (abbreviated notation)	**Complexity**
TDC	O (VE)
SALC (Fang et al., 2005)	O ($KV(E + V \log V)$)
AO (Gan and Liew 2010a)	O ($KV(E + V \log V)$)
RAO (Gan and Liew 2011)	O ($KV(E + V \log V)$)
LPR/SFR (Laborczi et al., 2001)	O ($V^2E \log V$)
α^+-MIN-SUM (Yang et al., 2005a)	O ($E^3 + L$) where ; L: sum of all edge length
SOPS (Ho and Mouftah 2002)	O ($E^2 \log V$)
DP2LC (Gomes et al., 2008)	Undefined

The result in Figure 5.14 compares the performance of TDC against two well-known link-disjoint paths-finding algorithms which are the Remove-Find (RF) method (or Two-Step method) and the Suurballe (SB) algorithm. These algorithms are able to identify the optimal solution in their respective (extreme) settings of α. That is when α=0, the solution by RF is optimal (Gan and Liew 2010a). When α=1, it is actually a 1+1 protection, and hence the solution by Suurballe is known to be optimal (Laborczi et al., 2001).

Using the network example and link-cost parameter of Figure 5.1 with node 'A' as the source node and node 'H' the destination node, Figure 5.14 shows the total cost incurred (interpolated) by the respective link-disjoint paths-finding algorithm with respect to α.

There were 3 MO solutions detected by the TDC while the algorithm was running at α = 0.1818, α = 0.25 and α=0.4 respectively. These α values are the critical values as determined from the properties derived in Theorem 1(iii), Corollary 1(ii), and Corollary 3.

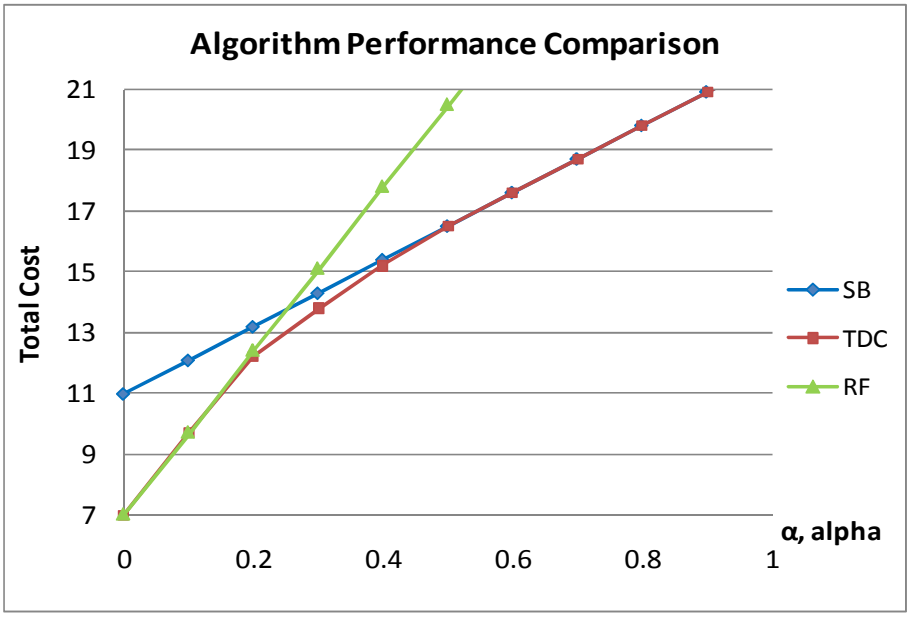

Figure 5.14: Paths-Finding Algorithms Performance Comparison

By observing the performance of the paths-finding algorithms in Figure 5.14, the TDC algorithm ensures the link-disjoint solution found is minimal for all values of α, especially between α=0.1 to α=0.5, where the optimal solutions (MO solutions) were neither found by the RF nor the SB.

In Figure 5.15, Figure 5.16 and Figure 5.17, the average CPU time per call for the three proposed α+1 optimum algorithm namely the AO, RAO and TDC are presented with α = 0.18, α = 0.25 and α=0.4 accordingly. The AO, RAO and TDC algorithms are modeled using C++ programming language. The CPU times were obtained from a system running on Intel Core 2 Quad Q9300 processor at 2.5GHz with 8GB of RAM. The results in Figure 5.15, Figure 5.16 and Figure 5.17 show that the TDC with its low complexity has the lowest running time as expected. The RAO has the second lowest run time followed by the AO. As mentioned, the RAO incorporates additional derived

optimal solution properties in its search parameters which allows it to reduce the process in its algorithm resulting in a lower running time as compared to the AO.

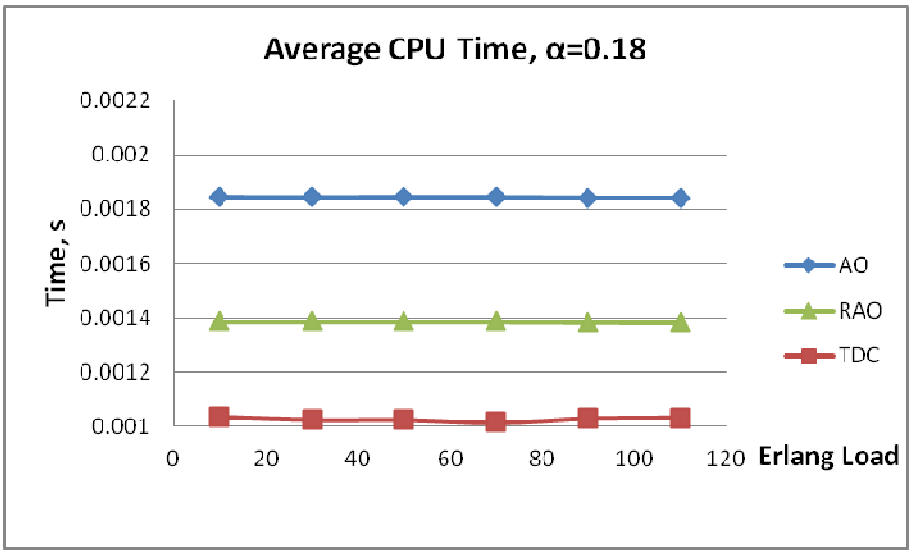

Figure 5.15: Average CPU time per call, α=0.18

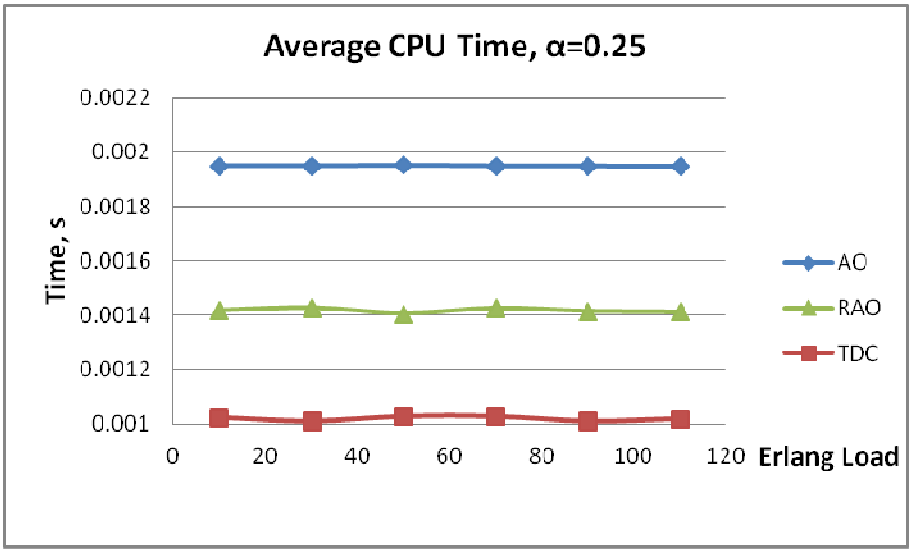

Figure 5.16: Average CPU time per call, α=0.25

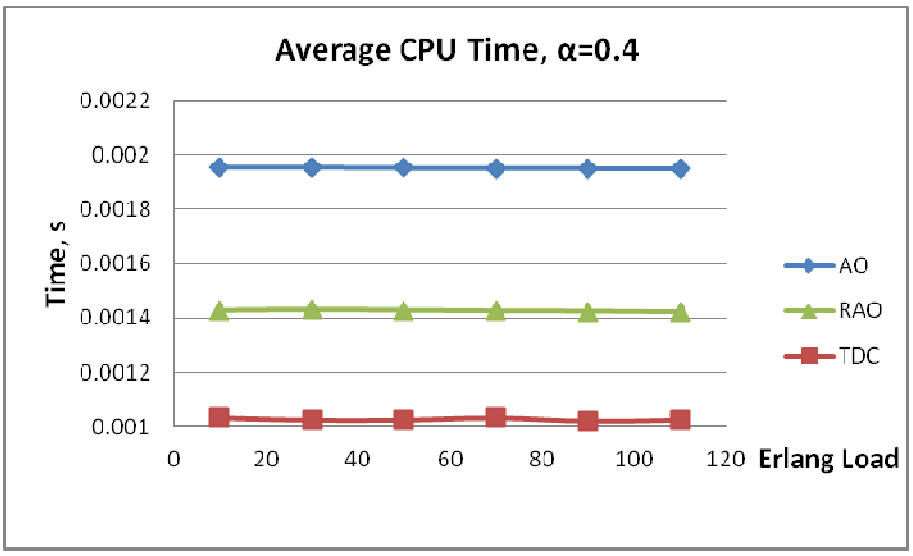

Figure 5.17: Average CPU time per call, α=0.4

CHAPTER 6

CONCLUSIONS

The outcome of this thesis was to develop novel algorithms for the application of α+1 path protection whereby the solution consists of the optimal pair of link-disjoint paths. The problem formulation is expressed as the minimization of the total cost for the primary-secondary path-costs where the secondary path cost is weighted by α, and $0<\alpha<1$. The need for a novel algorithm specifically for the α+1 path protection arises when the occurrence of the mid-optimal (MO) solution was observed and current link-disjoint paths-finding algorithms are not designed to efficiently identify the optimal solutions.

The properties of the MO solution derived in Chapter 3 provides the elements to construct a more efficient and effective optimum paths-finding algorithm for the α+1 protection. The α-optimum (AO) and reversed α-optimum (RAO) algorithms proposed in Chapter 4 utilizes the common *K*-shortest path algorithm incorporated with the MO properties to effectively obtain the optimal solution. With the use of the MO properties, a number of candidate solution check rules are introduced into the proposed algorithms, in addition to a stop rule to ensure efficient running time during execution.

In Chapter 5, the toggling dual cost (TDC) algorithm applies the Suurballe graph transformation and a modified version of the Bellman-Ford algorithm to find the optimal path for the α+1 protection in the transformed graph. The path found in the transformed graph can be later translated as the optimal link-disjoint path-pair in the original graph. Due to the approach of this algorithm which does not perform the *K*-shortest path iteration, it is shown to exhibit a lower time complexity compared to the other algorithms. A performance comparison through simulation is presented to demonstrate the effectiveness of the proposed algorithm in obtaining the optimal solutions.

The results from this research can be further extended to consider a distributed approach in implementing the α+1 protection. The application of α+1 protection in wireless multihop networks is a potential research direction worth investigating. The proposed optimum algorithms could also be further developed to be integrated into existing Internet protocols such as the Resource Reservation Protocol (RSVP).

REFERENCES

Aalamifar, F., Hassanein, H.S., and Takahara, G., 2012. Viability of Powerline Communication for the Smart Grid. *26th Biennial Symposium on Communications (QBSC)*, 28-29 May 2012, Kingston, ON, pp. 19-23.

Assi, C., Ye, Y., Shami, A., Dixit, S., and Ali, M., 2003. A Hybrid Distributed Fault-Management Protocol for Combating Single-Fiber Failures in Mesh-Based DWDM Optical Networks. *IEEE Global Telecommunications Conference (GLOBECOM) 2002*, 17-21 November 2002, Taipei, Taiwan, vol. 3, pp. 2676-2680.

Barbehenn, M., 1998. A Note on the Complexity of Dijkstra's Algorithm for Graphs with Weighted Vertices. *IEEE Transactions on Computers*, 47(2), pp. 263.

Bellman, R., 1958. On a Routing Problem. *Quarterly of Applied Mathematics*, 16(1), pp. 87-90

Benhamiche, A., Mahjoub, A.R. and Perrot, N., 2010. Design of Optical WDM Networks. *14th International Telecommunications Network Strategy and Planning Symposium (NETWORKS)*, 27-30 September 2010, Warsaw, pp. 1-7.

Bhandari, R., 1999. *Survivable Networks: Algorithms for Diverse Routing*. Massachusetts: Kluwer Academic Publishers.

Bhandari, R., 1994. Optical Diverse Routing in Telecommunication Fiber Networks. *IEEE INFOCOM 1994*, 12-16 June 1994, Toronto, Canada, pp. 1498-1508.

Charbonneau, N. and Vokkarane, V.M., 2011. A Survey of Advance Reservation Routing and Wavelength Assignment in Wavelength-Routed WDM Networks. *IEEE Communications Surveys and Tutorials*, (99), pp. 1-28.

Cisco, 2000, *Introducing DWDM* [Online]. Available at: http://www.cisco.com/univercd/cc/td/doc/product/mels/dwdm/dwdm_fns.htm [Accessed: 28 August 2012]

Das, A., Martel, C., and Mukherjee, B., 2009. A Partial-Protection Approach Using Multipath Provisioning. *IEEE International Conference on Communications*, 14-18 June 2009, Dresden, pp.1-5.

Department of Defense, 2005 *The Implementation of Network-Centric Warfare*. Washington, D.C.: U.S. Government Printing Office.

Dijkstra, E.W., 1959. A Note on Two Problems in Connexion with Graphs. *Numerische Mathematik*, 1(1), pp. 269-271.

Ellison, R.J., Fisher, D.A., Linger, R.C., Lipson, H.F., Longstaff, T., and Mead, N.R., 1999. *Survivable Network Systems: An Emerging Discipline.* Technical Report: CMU/SEI-97-TR-013 ESC-TR-97-013, Carnegie Mellon Software Engineering Institute, Carnegie Mellon University. Pittsburgh, PA.

Fang, J., Sivakumar, M., Somani, A.K., and Sivalingam, M., 2005. On Partial Protection in Groomed Optical WDM Mesh Networks. *Dependable Systems and Networks 2005*, 28 June – 1 July 2005, Yokohama, Japan, pp. 228-237.

Fiedler, J., Riede, C., and Magedanz, T., 2009. Converged Media and Communication Services. *First Conference on IT Revolutions*, 17-19 December 2008, Venice, pp. 1-7.

Fu-Li, S., Yong-Lin, L., and Yi-Fan, Z., 2011. A Military Communication Supernetwork Structure Model for Netcentric Environment. *International Conference on Computational and Information Sciences (ICCIS)*, 17-19 December 2010, Chengdu, pp. 33-36.

Galati, G., 2000. *Trading Volumes, Volatility and Spreads in Foreign Exchange Market: Evidence from Emerging Market Countries*. Technical Report: Working Paper No 93, Bank for International Settlements, Basel, Switzerland.

Gamage, M., Hayasaka, M., and Miki, T., 2007. QoS Guaranteed and Reliable Network Architecture for Real-Time Applications over the Connection Oriented Internet. *International Telecommunications Network Strategy and Planning Symposium*, 6-9 November 2006, New Delhi, pp. 1-6.

Gan, M.L., and Liew, S.Y., 2009. Uninterrupted Network Communication with Efficient Resource Utilization. *Symposium on Progress of Information & Communication Technology 2009*, 7-8 December 2009, Kuala Lumpur, Malaysia, pp. 90-95. Available from: http://spict.utar.edu.my/SPICT-09CD/contents/pdf/SPICT09_A-4_5.pdf [Accessed: 29 August 2012]

Gan, M.L., and Liew, S.Y., 2010a. An Optimum Paths-Finding Algorithm for α+1 Paths Protection. *Ultra Modern Telecommunications and Control Systems and Workshops,* 18-20 October 2010, Moscow, Russia, pp. 531-537.

Gan, M.L., and Liew, S.Y., 2010b. Performance Study of Paths-Finding Algorithm in α+1 Path Protection. *2nd Symposium on Progress of Information & Communication Technology 2010*, 12-13 December 2010, Kuala Lumpur, Malaysia, pp. 61-66. Available from: http://spict.utar.edu.my/SPICT-10CD/papers/spict10_30.pdf [Accessed: 29 August 2012]

Gan, M.L., and Liew, S.Y., 2013. Effective Algorithms for Finding Optimum Pairs of Link-Disjoint Paths in α+1 Path Protection. *Telecommunication Systems Journal*, 52(2), pp. 783-797.

Gerstel, O. and Ramaswami, R., 2000. Optical Layer Survivability: A Service Perspective. *IEEE Communications Magazine*, 38(3), pp. 104-113.

Gerstel, O., and Sasaki, G., 2010. A General Framework for Service Availability for Bandwidth Efficient Connection-Oriented Networks. *IEEE/ACM Transactions on Networking*, 18(3), pp. 985-995.

Ghassemi, H., and Wunnava, S., 2002. Development of an Operational Medical Network (MEDNET) Model. *IEEE Southeastcon '95. 'Visualize the Future'*, 26-29 March 1995, Raleigh, NC, pp. 162-164.

Giorgetti, A., Andriolli, N., Valcarenghi, L., and Castoldi, P., 2005. Impact of Idle Protection Capacity Reuse on Multi-Class Optical Networks. *International Workshop on Design of Reliable Communication Networks (DRCN) 2005*, 16-19 October 2005, Ischia, Italy, pp. 149-154.

Gomes, T, Craveirinha, J., and Jorge, L., 2008. An Effective Algorithm for Obtaining the Minimal Cost Pair of Disjoint Paths with Dual Arc. *Computers & Operations Research*, 36(5), pp. 1670-1682.

Graves, A.F., Wallace, B., Periyalwar, S., and Riccardi, C., 2005. Clinical Grade – A Foundation for Healthcare Communications Networks. *5th International Workshop on Design of Reliable Communications Networks*, 16-19 October 2005, Ischia, Italy, pp. 395-402.

Grover, W.D., and Clouqueur, M., 2005. Span-Restorable Mesh Networks with Multiple Quality of Protection (QoP) Service Classes. *Photonics Network Communications*, 9(1), pp. 19-34.

Gumaste, A., Mehta, S., Arora, I., Goyal, P., Rana, S., and Ghani, N., 2010. Omnipresent Ethernet – Technology Choices for Future End-to-End Networking. *Journal of Lightwave Technology*, 28(8), pp. 1261-1277.

Guo, L., Wang, X., Cao, J., Li, L., Yang, T., and Yang, F., 2008. Shared Sub-Path Protection Algorithm with Recovery Time Constraint in WDM Networks. *4th International Conference on Networking and Services 2008*, 16-21 March 2008 Gosier, pp. 153-158.

Guo, L., Yu, H., and Li, L., 2005. Dynamic Survivable Algorithm for Meshed WDM Optical Networks. *Journal of Network and Computer Applications*, 30(1), pp. 282-295.

Guo, L., Yu, H., Zhou, T., and Li, L., 2004. Dynamic Shared-Path Protection Algorithm for Dual-Risk Failures in WDM Mesh Networks. *International Conference on Parallel Processing Workshops (ICPP), 2004*, 15-18 August 2004, Montreal, Quebec, pp. 394-398.

Guo, Y., Kuipers, F., and Mieghem, P.V., 2003. Link-disjoint Paths for Reliable QoS Routing. *International Journal of Communication Systems*, 16(9), pp. 779 – 798.

Haider, A., and Harris, R., 2007. Recovery Techniques in Next Generation Networks. *IEEE Communications Surveys & Tutorials*, 9(3), pp. 2-17.

Harris, J.M., Hirst, J.L., and Mossinghoff, M.J., 2008. *Combinatorics and Graph Theory*. New York: Springer.

Hershberger, J., Maxel, M., and Suri, S., 2007a. Finding the *k* Shortest Simple Paths: A New Algorithm and Its Implementation. *ACM Transactions on Algorithms*, 3(4), article 45.

Hershberger, J., Suri, S., and Bhosle, A., 2007b. On the Difficulty of Some Shortest Path Problems. *ACM Transactions on Algorithms*, 3(1), article 5.

Ho, P.H., and Mouftah, H.T., 2002. Issues on Diverse Routing for WDM Mesh Networks with Survivability. *Computer and Communication Networks 2001*, 15-17 October 2001, Scottsdale, pp. 61-66.

Huang, S., Xia, M., Martel, C.U., and Mukherjee, B., 2010a. A Multistate Multipath Provisioning Scheme for Differentiated Failures in Telecom Mesh Network. *Journal of Lightwave Technology*, 28(11), pp. 1585-1596.

Huang, S., Xia, M., Martel, C., and Mukherjee, B., 2010b. A Multistate Multipath Provisioning Scheme for Combating Node Failures in Telecom Mesh Networks. *Military Communications Conference 2009*, 18-21 October 2009, Boston, MA, pp. 1-5.

Hunt, L., 2012. Drivers for Packet-Based Utility Communications Networks-Teleprotection. *International Conference on Advanced Power System Automation and Protection (APAP)*, 16-20 October 2011, Beijing, 3, pp. 2453-2457.

Jelassi, S. et al., 2012. Quality of Experience of VoIP Service: A Survey of Assessment Approaches and Open Issues. *IEEE Communications Surveys & Tutorials*, 14(2), pp. 491-513.

Jungnickel, D., 2008. *Graphs, Networks and Algorithms*. New York: Springer.

Kerivin, H., and Mahjoub, A.R., 2005. Design of Survivable Networks: A Survey. *Networks*, 46(1), pp. 1-21.

Kim, J., Lee, I., and Noh, S., 2010. VoIP QoS (Quality of Service) Design of Measurement Management Process Model. *International Conference on Information Science and Applications (ICISA)*, 21-23 April 2010, Seoul, pp. 1-6.

Kodian, A., and Grover, W.D., 2006. Multiple-Quality of Protection Classes Including Dual-Failure Survivable Services in p-Cycle Networks. *International Conference on Broadband Networks*, 3-7 October 2005, Boston, 1, pp. 231-240.

Koster, A.M.C.A., Zymolka, A., Jager, M., and Hulsermann, R., 2005. Demand-wise Shared Protection for Meshed Optical Network. *Journal of Network and Systems Management*, 13(1), pp. 35-55.

Kuperman, G., Modiano, E., and Narula-Tam, A., 2011. Analysis and Algorithms for Partial Protection in Mesh Networks. *IEEE INFOCOM 2011*, 10-15 April 2011, Shanghai, pp. 516-520.

Laborczi, P., 2002. "*Configuration of Fault Tolerant Infocommunication Networks*," Ph.D. thesis, Department of Telecommunications and Telematics, Budapest University of Technology and Economics, Budapest, Hungary.

Laborczi, P., Tapolcai, J., Ho, P.H., Cinkler, T., Recski, A., and Mouftah, H.T., 2001. Algorithms for Asymmetrically Weighted Pair of Disjoint Paths in Survivable Networks. *DRCN 20*01, 7-10 October 2001, Budapest, pp.220-227. Available from: http://opti.tmit.bme.hu/~tapolcai/papers/tapolcai01drcn.pdf [Accessed: 29 August 2012]

Leepila, R., Oki, E., and Kishi, N., 2011. Scheme to Find *k* Disjoint Paths in Multi-Cost Networks. *IEEE International Conference on Communications (ICC)*, 5-9 June 2011, Kyoto, pp. 1-5.

Li, G., Yates, J., Wang, D., and Kalmanek, C., 2001. Reliable Optical Network Design. *SPIE 2001 (Optical Networking)*, 12 November 2001, Beijing China, 4585(111).

Li, H., Zhang, Y., and Wu, J., 2008. The Future of E-Commerce Logistics. *IEEE International Conference on Service Operations and Logistics, and Informatics*, 12-15 October 2008, Beijing, 1, pp. 1403-1408.

Li, W., 2011. Study on Survivability Mechanism of Military Optical Fiber Communication Transmission Network. *IEEE 3rd International Conference on Communication Software and Networks,* 27-29 May 2011, Xi'an, pp. 144-147.

Liao, X., Jin, H., and Shu, C., 2011. A Voice Gateway System Based on P2P SIP VOIP. *6th International Conference on Pervasive Computing Applications (ICPCA)*, 26-28 October 2011, Port Elizabeth, pp. 315-323.

Liew, S.Y., and Gan, M.L, 2012. An Exact Optimum Paths-Finding Algorithm for α+1 Paths Protection. *2012 International Conference on Information Networking*, 1-3 February 2012, Bali, Indonesia, pp. 210-215.

Liu, Y., Mendiratta, V.B., and Trivedi, K.S., 2004. Survivability Analysis of Telephone Access Network. *15th International Symposium on Software Reliability Engineering*, 2-5 November 2004, Bretagne, France, pp. 367-377.

Long, L., 2010. "*On Traffic Grooming and Survivability in WDM Optical Networks*," Ph.D. thesis, Iowa State University, Ames, Iowa.

Luan, W., Sharp, D., and Lancashire, S., 2010. Smart Grid Communication Network Capacity Planning for Power Utilities. *IEEE PES Transmission and Distribution Conference and Exposition*, 19-22 April 2010, New Orleans, LA, USA, pp. 1-4.

Luo, H., Li, L., and Yu, H., 2009. Routing Connections with Differentiated Reliability Requirements in WDM Mesh Network. *IEEE/ACM Transactions on Networking*, 17(1), pp. 253-266.

Luo, H., Yu, H., Li, L., and Wang, S., 2006. On Protecting Dynamic Multicast Sessions in Survivable Mesh WDM Networks. *IEEE International Conference on Communications 2006*, 11-15 June 2006, Istanbul, 2, pp. 835-840.

Martins, E., and Pascoal, M., 2003. A New Implementation of Yen's Ranking Loopless Paths Algorithm. *4OR: Quarterly Journal of the Belgian, French and Italian Operations Research Societies*, 1(2), pp. 121-133.

McGorman, R.E., 2002. A Methodology for Designing Survivable Telephone Networks. *IEEE International Conference on Communications 1988*, 12-15 June 2012, Philadelphia, PA, 2, pp. 1172-1176.

Mukherjee, B., 2006. *Optical WDM Networks,* New York: Springer.

Mukherjee, B., 1992. WDM-based Local Lightwave Networks. II. Multihop Systems. *IEEE Networks*, 6(2), pp. 20-32.

Needham, M., and Harris, J., 2008. Traffic and Network Modeling for Next Generation Applications. *IEEE International Symposium on Broadband Multimedia Systems and Broadcasting 2008*, 31 March – 2 April 2008, Las Vegas, NV, pp. 1-18.

Ou, C., Zhang, J., Zang, H., Sahasrabuddhe, L.H., and Mukherjee, B., 2003. Near Optimal Approaches for Shared-Path Protection in WDM Mesh Networks. *IEEE International Conference on Communications (ICC)*, 11-15 May 2003, Anchorage, Alaska, pp. 1320-1324.

Peros, H.G., 2006. *Connection-Oriented Networks*, West Sussex, England: John Wiley & Sons Ltd.

Pinciroli, F., Corso, M., Fuggetta, A., Masseroli, M., Bonacina, S., and Marceglia, S., 2011. Telemedicine and E-Health. IEEE Pulse, 2(3), pp. 62-70.

Rak, J., and Molisz, W., 2009. A New Approach to Provide the Differentiated Levels of Network Survivability under a Double Node Failure. *11th International Conference on Transparent Optical Networks (ICTON),* 28 June – 2 July 2009, Azores, pp. 1-4.

Rak, J., 2010. *k*-Penalty: A Novel Approach to Find *k*-Disjoint Paths with Differentiated Path Costs. *IEEE Communications Letters*, 14(4), pp. 354-356.

Rak, J., 2012. Fast Service Recovery Under Shared Protection in WDM Networks. *Journal of Lightwave Technology*, 30(1), pp. 84-95.

Ramamurthy, S. and Mukherjee, B., 1999. Survivable WDM Mesh Networks, Part 1 – Protection. *IEEE INFOCOM 1999*, 21-25 March 1999, New York, pp. 744-751.

Ramamurthy, S., and Mukherjee, B., 2002. Survivable WDM Mesh Networks, Part II –Restoration. *IEEE International Conference on Communications 1999*, 6-10 June 1999 Vancouver, BC, pp. 2023-2030.

Rinaldi, S.M., Peerenboom, J.P., and Kelly, T.K., 2001. Identifying, Understanding and Analyzing Critical Infrastructure Interdependencies. *IEEE Control Systems Magazine*, 2(6), pp. 11-25.

Roy, R., and Mukherjee, B., 2008. Degraded-Service-Aware Multipath Provisioning in Telecom Mesh Networks. *Optical Fiber Communication/National Fiber Optic Engineers Conference 2008*, 24-28 February 2008, San Diego, CA, pp. 1-3.

Saeedinia, R., 2011. Protection Switching in Communication Networks. *16th European Conference on Network and Optical Communications 2011*, 20-22 July 2011, Newcastle-Upon-Tyne, pp. 20-23.

Santos, J., Pedro, J., Monteiro, P., and Pires, J., 2009. Multilayer Optimization of Inverse-Multiplexed 100 Gb/s Ethernet Services over Optical Transport Networks. *Next Generation Internet Networks 2009*, Aveiro, 1-3 July 2009, pp. 1-8.

Schupke, D.A., and Prinz, R.G., 2004. Capacity Efficiency and Restorability of Path Protection and Rerouting in WDM Networks Subject to Dual Failure. *Photonic Network Communications*, 8(2), pp. 191-207

Shi, F., Li, C., Qin, D., Zhu, Y., and Yang, F., 2012. A Complexity Measure for Military Communication Networks. *Military Communications Conference 2011*, 7-10 November 2011, Baltimore, MD, pp. 1708-1713.

Sivakumar M., Sivalingam K.M., and Somani, A.K., 2007. Partial Protection in Optical WDM Networks: Enhanced Support for Dynamic Traffic. *2006 Broadband Communications, Networks and Systems*, 1-5 October 2006, San Jose, CA, pp. 1-8.

Song, L., and Mukherjee, B., 2009. Accumulated-Downtime-Oriented Restoration Strategy with Service Differentiation in Survivable WDM Mesh Networks. *IEEE/OSA Journal of Optical Communications and Networking*, 1(1), pp. 113-124.

Song, L., Zhang, J., and Mukherjee, B., 2007. Dynamic Provisioning with Availability Guarantee for Differentiated Services in Survivable Mesh Networks. *IEEE Journal on Selected Areas in Communications*, 25 (3), pp. 35-43.

Sterbenz, J.P.G., Hutchison, D., Cetinkaya, E.K., Jabbar, A, Rohrer, J.P., Scholler, M., and Smith, P., 2010. Resilience and Survivability in Communication Networks: Strategies, Principles and Survey of Disciplines. *Computer Networks*, 54(8), pp. 1245-1265.

Subramani, K., and Kovalchick, L., 2005. A Greedy Strategy for Detecting Negative Cost Cycles in Networks. *Future Generation Computer Systems*, 21(4), pp. 607-623.

Sukhni, E.M.A., and Mouftah, H.T., 2008. Distributed Lightpath Control and Management Simulator for Survivable Wavelength-Routing Networks. *11th Communications and Networking Simulation Symposium*, 14-17 April 2008, Ottawa, Canada, pp. 109-114.

Suurballe, J.W., and Tarjan, R.E., 1984. A Quick Method for Finding Shortest Pairs of Disjoint Paths. *Networks*, 14(2), pp. 325-336.

Tizghadam, A., and Leon-Garcia, A., 2010. Betweenness Centrality and Resistance Distance in Communication Networks. *IEEE Network*, 24(6), pp. 10-16.

Todd, B., and Doucette, J., 2011. Fast Efficient Design of Shared Backup Path Protected Networks Using a Multi-Flow Optimization Model. *IEEE Transactions of Reliability*, 60(4), pp. 788-800.

Todimala, A., and Ramamurthy, B., 2006. A Heuristic with Bounded Guarantee to Compute Diverse Paths Under Shared Protection in WDM Mesh Networks. *Global Telecommunications Conference, 2005*, 28 November – 2 December 2005, St. Louis, MO, pp. 1915-1919.

Torbatore, M., Maier, C., and Pattavina, A., 2006. Capacity Versus Availability Trade-Offs for Availability Based Routing. *Journal of Optical Networking*, 5(11), pp. 858-869.

Tracca, M., Fumagalli, A., Paradisi, A., Unghvary, F., Gadhiraju, K., Lakshmanan, S., Rossi, S.M., de Campos Sachs, A., and Shah, D.S., 2006. Differentiated Reliability in Optical Networks: Theoretical and Practical Results. *Journal of Lightwave Technology*, 21(11), pp. 2576-2586.

Vadrevu, C.S.K., Wang, R., and Mukherjee, B., 2012. Degraded Services in Mixed-Line-Rate Networks Using Multipath Routing. *IEEE 5th International Conference on Advanced Networks and Telecommunication Systems (ANTS),* 18-21 December 2011. Bangalore, pp. 1-3.

Veeraraghavan, M., and Karol, M., 1999. Internetworking connectionless and connection oriented networks. *IEEE Communications. Magazine*, 37(12) pp. 130–138.

Veeraraghavan, M., Zheng, X., and Huang, Z., 2006. On the Use of Connection-Oriented Networks to Support Grid Computing. *IEEE Communications Magazine*, 44(3), pp. 118-123.

Wang, H., Modiano, E., and Medard, M., 2002. Partial Path Protection for WDM Networks: End-to-End Recovery Using Local Failure Information. *ISCC'02*, 1-4 July, Taormina, Italy, pp. 719.

Wu, B., Ho, P.H., Yeung, K.L., Tapolcai, J., and Mouftah, H.T., 2011. Optical Layer Monitoring Schemes for Fast Link Failure Localization in All-Optical Networks. *IEEE Communications Surveys & Tutorials*, 13(1), pp. 114-125.

Wu, R., Qi, H., and Tang, L., 2010. An Optimal Resilient Routing Strategy with Differentiated Reliability Requirements Based on Hierarchy Model in WDM Mesh Networks. *IEEE International Conference on Network Infrastructure and Digital Content*, 24-26 September 2010, Beijing, pp. 892-896.

Xiao, X., and Ni, L.M., 1999. Internet QoS: A Big Picture. *IEEE Network*, 3(2), pp. 8-18.

Xue, G., Zhang, W., Tang, J., and Thulasiraman, K., 2005. Establishment of Survivable Connections in WDM Networks using Partial Path Protection. *IEEE International Conference on Communications (ICC) 2005*, 16-20 May 2005, Seoul, 3, pp. 1756-1760.

Yang, B., Zheng, S.Q., and Lu, E., 2005a. Finding Two Disjoint Paths in a Network with Normalized α^+-MIN-SUM Objective Function. *16th International Symposium Algorithms and Computation (ISAAC) 2005*, 19-21 December 2005, Sanya, Hainan, China. Berlin: Springer – Lecture Notes in Computer Science, 3827/2005, pp. 954-963.

Yang, B., Zheng, S.Q., and Lu, E., 2005b. Finding Two Disjoint Paths in a Network with Normalized α^--MIN-SUM Objective Function. *17th IASTED International Conference on Parallel and Distributed Computing and Systems*, 14-16 November 2005, Phoenix, AZ. Canada: ACTA Press pp. 342-348.

Yao, W., and Ramamurthy, B., 2005. Survivable Traffic Grooming in WDM Mesh Networks under SRLG Constraints. *International Conference on Communications 2005*, 16-20 May 2005. Seoul, Korea, 3, pp. 1751-1755.

Yao, W., and Ramamurthy, B., 2004. Survivable Traffic Grooming with Path Protection at the Connection Level in WDM Mesh Networks. *BroadNets'2004*, 25-29 October 2004, San Jose, CA, USA, pp. 310-319.

Ye, Y., Assi, C., Dixit, S., and Ali, M.A., 2002. A Simple Dynamic Integrated Provisioning/Protection Scheme in IP over WDM Networks. *IEEE Communications Magazine*, 39(11), pp. 174-182.

Yen, J.Y., 1971. Finding the K Shortest Loopless Paths In A Network. *Management Science*, 17(11), pp. 712-716.

Younis, O., and Fahmy, S., 2003. Constraint-Based Routing in the Internet: Basic Principles and Recent Research. *IEEE Communications Surveys & Tutorials*, 5(1), pp. 2-13.

Zhou, D., and. Subramaniam, S., 2000. Survivability in Optical Networks. *IEEE Network*, 14(6), pp. 16 – 23.