2006

# Design of survivable WDM network based on pre-configured protection cycle

Byungkyu Kang
*Edith Cowan University*

# DESIGN OF SURVIVABLE WDM NETWORK BASED ON PRE-CONFIGURED PROTECTION CYCLE

By
Byungkyu Kang

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
AT
EDITH COWAN UNIVERSITY
WESTERN AUSTRALIA
AUGUST 2006

# USE OF THESIS

The Use of Thesis statement is not included in this version of the thesis.

# Abstract

Wavelength Division Multiplexing (WDM) is an important technique which allows the transport of large quantities of data over optical networks. All optical WDM-based networks have been used to improve overall communication capacity and provide an excellent choice for the design of backbone networks. However, due to the high traffic load that each link can carry in a WDM network, survivability against failures becomes very important. Survivability in this context is the ability of the network to maintain continuity of service against failures, since a failure can lead to huge data losses. In recent years, many survivability mechanisms have been studied and their performance assessed through capacity efficiency, restoration time and restorability. Survivability mechanisms for ring and mesh topologies have received particular attention. Ring mechanism has the merit that it can offer fast restoration time ($50 \sim 60$ ms), but it requires redundancy in the order of $100\ \% \sim 200\%$, (the ratio of spare capacity to working capacity). On the other hand, mesh protection mechanisms require less redundancy ($50\ \% \sim 70\ \%$) than ring mechanisms, but have higher restoration time. The concept of pre-configured protection cycle ($p$-cycle) has been developed as a hybrid of ring and mesh protection mechanisms, it benefits from the fast restoration time of ring protection and the capacity efficiency of mesh topologies. The $p$-cycle method is based on closed cyclic routes to reduce restoration time. However, unlike the ring mechanism which only protects the working channels on the ring, the $p$-cycle method offers useful back up paths for protecting the straddling spans as well as the on-cycle spans. Interestingly, the $p$-cycle method can offer two restoration paths for straddling spans. As a result, $p$-cycle is reliable and effective in capacity utilization in optical mesh networks.

The purpose of this thesis is to investigate and design survivability algorithms based on the concept of $p$-cycles in optical WDM mesh networks, with the aim of balancing optimality of solution and computational complexity. We shall propose two new heuristic approaches to compute efficient $p$-cycles that can lead to superior performance in terms of capacity efficiency. First we determine a set of $p$-cycles through identification of fundamental cycles in the network, and then an Integer Linear Program (ILP) formulation is applied to protect the whole given traffic demand. Secondly, we propose a heuristic approach without using the ILP model based on a matching criterion between the distribution of the working channels and the distribution of the protected channels on the selected set of $p$-cycles. Significant results has been achieved in terms of capacity efficiency and restoration time through the set of $p$-cycle algorithms developed in this thesis.

DECLARATION

I certify that this thesis does not, to the best of my knowledge and belief:

(i) incorporate without acknowledgment any material previously submitted for a degree of diploma in any institution of higher education.

(ii) contain any material previously publixhed or written by another person except where due reference is made in the text; or

(iii) contain any defamatory material.

I also grant permission for the Library at Edith Cowan University to make duplicate copies of my thesis as required.

Signature: ______________________

Byungkyu Kang

Date: ______________________

# Acknowledgements

I owe my greatest thanks to my principal supervisor Prof. Daryoush Habibi who has been providing me with his graceful guidance and great insights. I am greatly indebted to Prof. Habibi for his infinite patience, encouragements and insightful comments. This thesis could not be carried out without his support. I cherish the time we work together. I am also thankful to Associate Prof. Mike MyungOk Lee for his vision for a better world and his encouragements for pursuing a master degree.

I would like to appreciate my family for supporting me. Dad, seung-gil Kang, I am grateful to support my financial problem throughout my master's education and confidence to move forward. Mom, MiSoon Noh, there is no words to describe how much I love you and appreciate the guidance you have given me over the years. I also thank my younger sister, GiNa Kang.

I need to thank my fiance, YuMi Kim, for her unbelievable support and understanding through my entire study. Without all of your support and love, I am not able to finish my degree on time. I also thank all my friends and ildanpishin club in korea.

Finally, I am very lucky to work with committee members on my research. I would like to thank Nghia, Kungmeng, Viet for providing technical advice, and cooperation on simulation. Also, I would like to thank seongwon, sengmin, heusari, mah, justin, and Chung-Kiak for their kind help and encouragement throughout my study.

# Table of Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

With the rapid evolution of network capability, a huge demand for network service has arisen. This demand is increasing to serve new applications such as video, movies online and multimedia communications. Conventional copper wire-based networks are not enough to support this exponentially growth in bandwidth demand as they are limited by electronic speeds to a few Gbps [1]. Thus, such networks are not able to guarantee serving Quality of Service (QoS) for future network demand. However, optical networks are promising to satisfy network demand for future backbone networks. Current optical networks technically offer bandwidth demand on the order of 50Tbps within extremely low bit error rates [2]. In addition, the potential capabilities of an optical network can include [2];

- a huge bandwidth,
- low signal attenuation of as low as 0.2dB/km,
- low signal distortion and low power requirement,
- low material usage,
- small space requirement,
- high speed transmission.

With these advantages, optical networks are becoming an excellent choice as transmission systems for future backbone networks. Optical networks are able to provide high speed transport services to predominant client layers, such as Internet Protocol (IP), Asynchronous Transfer Mode(ATM) and Synchronous Optical NETwork (SONET) / Synchronous Digital Hierarchy (SDH). These services may cooperate with user applications. Several other layer combinations are possible such as IP over SONET over optical and ATM over optical [3]. Fig. 1 presents a layered view of a network consisting of a second generation optical layer that supports a variety of client layers above an optical network.

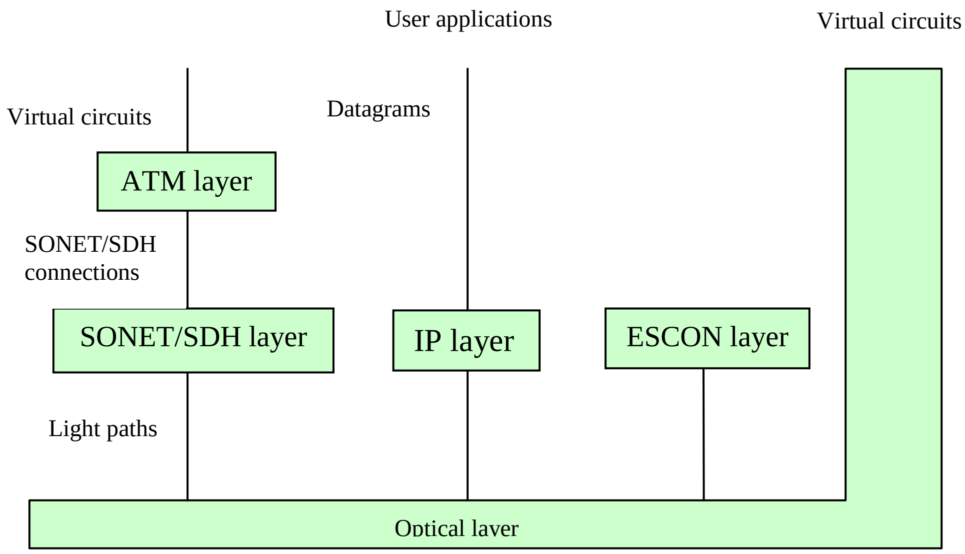



Figure 1.1: Service layer of optical network

In optical networks, Wavelength Division Multiplexing (WDM) has been developed as a technology that can satisfy the required bandwidth demand for the future [1]. Multiple wavelength channels defined as units of capacity, are used to transmit over one single span in WDM networks. This makes it possible for WDM optical networks to solve the bandwidth crunch problem for future backbone networks. However, a channel failure such as fiber cuts or component error can lead to losses of huge amount of date in telecommunication networks. The interruption of service for a even short time may have disastrous consequences unless the channel failure is immediately recovered. For this reason, survivability against network failures is a particularly important issue. Network failures basically occur at either nodes or links of the network. Link failure is caused by cable cuts while node failure refers to the failure of components at the network nodes. The survivability of a network refers to a network's capability to provide continuous service and maintain quality of service in the presence of such failures. In this context, survivability can be classified into three categories: 1) prevention, 2) network design, 3) traffic management and restoration [4]. Prevention focuses on improving equipment and system reliability. Network design aims to achieve efficiency in terms of sufficient spare capacity and faster restoration time against failures. Traffic management and restoration are concerned with the network load in the event of failure and connection re-establishment around the failure. This thesis focuses on network design and survivability.

As mentioned above, WDM transfers optical signal using different wavelengths over a single optical fiber so that the technology is more efficient in terms of utilizing capacity. However, WDM poses *routing and wavelength assignment* (RWA) problems when several optical signals share the same fiber in the networks. A lightpath, which is an optical channel, must use the same wavelength to transfer optical signals from a source node to a destination node. This consideration is known as the *wavelength continuity constraint*. This can lead to high blocking probability and inefficient utilization of wavelength channels. To overcome this problem, wavelength converters are utilized and many researchers have investigated their use [5, 6, 7]. However, we do not consider wavelength converters since RWA problems are out of scope of this thesis. Thus, we assume that full wavelength is available at all nodes in the network.

## 1.1 Research Motivations

Many researchers have investigated the issue of network survivability. Survivability should guarantee maximum restorability to provide QoS against failures. There exists both pre-planned protection and dynamic restoration mechanisms in survivability. Dynamic protection methods are not able to guarantee 100% protection, but offers faster restoration time [8]. Pre-planned protections, however, provide 100% protection by reserving alternative paths in advance [8]; hence this mechanism is a more interesting development in survivable networks. In pre-planned protection, two distributed network survivability schemes have received particular attention [9, 10, 8] over the last few years. The first approach is the ring protection mechanism. This scheme can offer fast restoration time, but it requires redundancy in the order of 100% $\sim$ 200% (the ratio of the amount of spare capacity required to the total working capacity used in the network). The second approach is the mesh protection mechanism. This mechanism can utilize efficient spare capacity and the redundancy is 50% $\sim$ 70%, but this scheme needs more restoration time than ring protection mechanisms and is more complicated to implement [11].

To tackle the restoration time issues of mesh protection mechanism and the capacity issues of ring protection, pre-configured protection cycle ($p$-cycle) has been proposed by Grover [12]. The basic ideal of $p$-cycle protection has been developed to utilize the advantages of both ring and mesh protection mechanisms. It benefits from the fast restoration time of ring mechanisms and the capacity efficiency of mesh mechanism. $p$-Cycle protection makes it possible to achieve low spare capacity by determining an appropriate set of $p$-cycles [11]. However, determination of the optimal set of $p$-cycles for protection is an NP-hard problem. Existing approaches for solving the $p$-cycle problem are through the use of an Integer Linear Programming (ILP) model or using heuristic methods. The objective of the ILP model is to achieve the optimal solution in terms of minimizing the spare capacity while maintaining 100% protection. However, the ILP model becomes intractable with large scale networks where the number of possible variables is very high. This gives the motivation to investigate heuristic approaches for *p*-cycle network design. Heuristic approaches can achieve near optimal solutions within acceptable restoration time. Most of the research efforts in p-cycle network design have been focused on heuristic approaches to make the ILP model solvable. Simple and efficient heuristic methods are greatly desirable.

## 1.2 Thesis Objective

The main objective of this thesis is to investigate **the problem of designing survivable WDM Networks based on *p*-Cycles**. In general, computational complexity and optimality of solutions are two performance metrics which pose conflicting requirements in optimization problems. Furthermore, the balance of these two metrics is important and has been the subject of significant research. Hence, this thesis investigates the balance these two performance metrics whilst achieving 100% protection against network failures.

### 1.2.1 Optimality of solutions

One of the most important issues in the network optimization problem is capacity utilization. It is generally evaluated by measuring the redundancy. Low redundancy is more efficient than high redundancy due to the fact that high redundancy requires large spare capacity to protect against failures. The objective of this thesis work is to find capacity optimized *p*-cycles in the network that minimize the use of spare capacity and then achieve optimality of solutions using a set of *p*-cycles found. That is, we investigate a new framework for computing an efficient set of *p*-cycles which can lead to superior performance in terms of capacity utilization in network protection.

### 1.2.2 Computational complexity

Computational complexity is an important factor when evaluating an algorithm. It is defined as the time it takes for an algorithm to find a solution. As discussed, the ILP model suffers from high computational complexity in dense networks even though it computes the optimal solution. If an algorithm can achieve a near optimal solution within an acceptable computational complexity, it will likely be a more desirable solution to the problem. In addition, computational complexity is an important issue to save money in reality. Because, if the network failures are not recovered from as quickly as possible, voice-call revenue losses can accrue at $100,000 a minute [13]. Therefore, one of the objectives in this thesis is to investigate the possibility of using a heuristic method in order to achieve the best performance in terms of computational complexity.

### 1.2.3 Balancing optimality of solution and computational complexity

Computational complexity and optimality of solutions are usually traded off against each other. Even though fast computation time can be achieved, low redundancy can not always be guaranteed. This is because fast computation time sometimes requires trading off network redundancy. In this thesis, computation time and the capacity utilization can be managed by our proposed approaches to achieve the desired requirements in terms of 100% restorability. Consequently, we wish to ensure 100% protection while minimizing the total spare capacity and reducing computation time.

## 1.3 Outline of The Thesis

This thesis is organized as follows.

Chapter 2 describes various schemes for network survivability and contains an introduction to the basic concepts of pre-configured cycle protection ($p$-cycle). In addition, it reviews the previous work done in terms of ILP models and heuristic approaches for $p$-cycle network design.

Chapter 3 introduces the preliminary graph theory and the notations used in this thesis. It also presents the real network topologies employed to evaluate our proposed approaches and details the assumption which the work of this thesis is based.

Chapter 4 presents a heuristic approach to network protection based on ILP model and gives some definition for the algorithms presented. Simulation results are then reported and these are compared with both exiting heuristic approaches and optimal solution.

Chapter 5 introduces a purely heuristic approach without an ILP model and examines the approach under a given traffic demand in a set of test networks. An analysis of the performance is presented in this chapter.

Chapter 6 offers some conclusions and discusses future research for survivability in mesh networks. Finally, it provides a summary of the research contributions made.

## 1.4 Publications

### Referred journal publication

- H. N. Nguyen, D. Habibi, Q. V. Phung, K. Lo, S. Lachowicz, **B. K. Kang**, "finding the optimum Solution in p-cycle Protection using Fundamental Cycles", Submitted to IEEE Communication Letter, January, 2007

### Referred conference publications

- **Byungkyu Kang**, D. Habibi, K. Lo, Q.V.Phung, H.N. Nguyen, " Design of Survivable Optical WDM Networks", submitted to IEEE Consumer Communications and Networking Conference, (CCNC 2007), USA

- **Byungkyu Kang**, D. Habibi, K. Lo, Q.V.Phung, H.N. Nguyen, A. Rassau, "An approach to Generate an Efficient set of Candidate p-Cycle in Optical WDM Mesh Networks", Asia pacific Conference on Communication (APCC 2006), September, 2006, Korea

- **Byungkyu Kang**, D. Habibi, Q. V. Phung, K. Lo, H. N. Nguyen, Myke M. Lee, ,"A Heuristic Approach for Efficient p-Cycle Network Design", International conference on Communication System and Networks (CSN 2006), August, 2006, Spain

- H.N. Nguyen, , D. Habibi, S. Lachowicz, Q.V. Phung, K. Lo, **B. Kang**, "Finding the Optimum Solution in p-Cycle Protection using Fundamental Cycles", IEEE Globecom conference, December, 2006, USA

- K. Lo, D. Habibi, Q.V. Phung, H.N. Nguyen and **B.K. Kang**, "A Hybrid p-Cycle Search Algorithm for Protection in WDM mesh Networks", The 14th IEEE International Conference on Networks (ICON 2006), September 2006, Malaysia

- K. Lo, D. Habibi, Q.V. Phung, H.N. Nguyen and **B.K. Kang**, "Dynamic p-cycles Selection in Optical WDM Mesh Networks", The 13rd International Conference on Networks (ICON 2005), December 2005, Malaysia

# Chapter 2

# Background and Literature Review

In this chapter, we describe optical networks and the main features of optical WDM networks. We present briefly Routing and Wavelength Assignment (RWA) and Wavelength Conversion (WC) problems in WDM networks and existing solutions to tackle these problems are described. In addition, this chapter reviews survivability mechanisms that have been developed and existing algorithms to solve challenges for $p$-cycle network design in terms of optimization of solution and computational complexity.

## 2.1 Optical WDM Network

Optical networks are a communication systems which are based on single or multiple wavelengths of light for transmission. It is a promising technology for accommodating growing bandwidth demands. Each node in the network is equipped with a set of transmitters and receivers to provide data transmission with each end-node connected via a lightpath. Optical networks can be classified into three categories depending on transmission method: Time Division Multiplexing (TDM), Code Division Multiplexing (CDM) and Wavelength Division Multiplexing (WDM). Each of these handles multiple demands using either time slots, wave shape or wavelength [14, 15, 16], respectively. WDM is more popular than TDM and CDM. This is because TDM and CDM require more complex hardware and synchronization whereas WDM is able to use components that are already available.

Optical networks employing WDM carry out data transmission along multiple wavelengths on each fiber. Fig 2.1 presents a diagram of a WDM transmission system. In general, WDM networks support 16, 32, 64, 128 or 256 wavelengths over a fiber, i.e., each wavelength is a single transmission channel at the required bit rate. Thus, optical WDM networks are able to carry a huge amount of traffic demand.

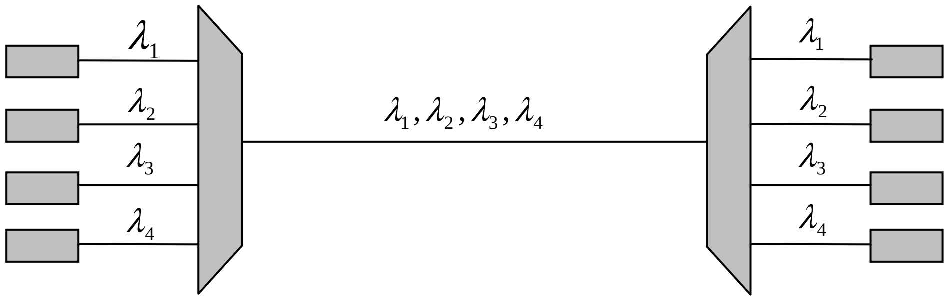



Figure 2.1: *Configuration of a basic WDM transmission system*

However, blocking is a big challenge in the design of WDM networks as mentioned before. Two existing approaches to improve blocking performance are routing wavelength assignment (RWA) and wavelength conversion (WC) placement. These two approaches have been investigated to minimize blocking probability. We present only brief introduction of the RWA and WC approaches since these are outside of scope of this thesis.

### 2.1.1 Routing and wavelength allocation

RWA is the basic control problem for transmission of data in a WDM network. If there exists enough wavelength channels in all fibers, lightpaths are set up to connect each end-node without a RWA problem. But, unfortunately, in practice fibers have a limited number of wavelengths so that RWA becomes a challenging network problem for data routing in WDM network.

RWA is classified into two types of problems depending on traffic demand: static and dynamic [17]. For static traffic, the connection requests are known in advance and lightpaths for these connections are set up. The objective is to minimize the number of wavelength channels requested for a given network. The RWA problem for static traffic is known as Static Lightpath Establishment (SLE) [18]. On the other hand, for dynamic traffic, a lightpath is established for each connection request when a connection request arrives. The objective is to establish a lightpath and assign wavelengths to maximize connection probability or improve the blocking performance. The RWA problem for dynamic traffic is known as Dynamic Lightpath Establishment (DLE) [18]. Many algorithms to solve these problems are found in the literature [19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29] and summarised in Fig 2.2. The RWA problem can be decoupled into two sub problems: the routing sub-

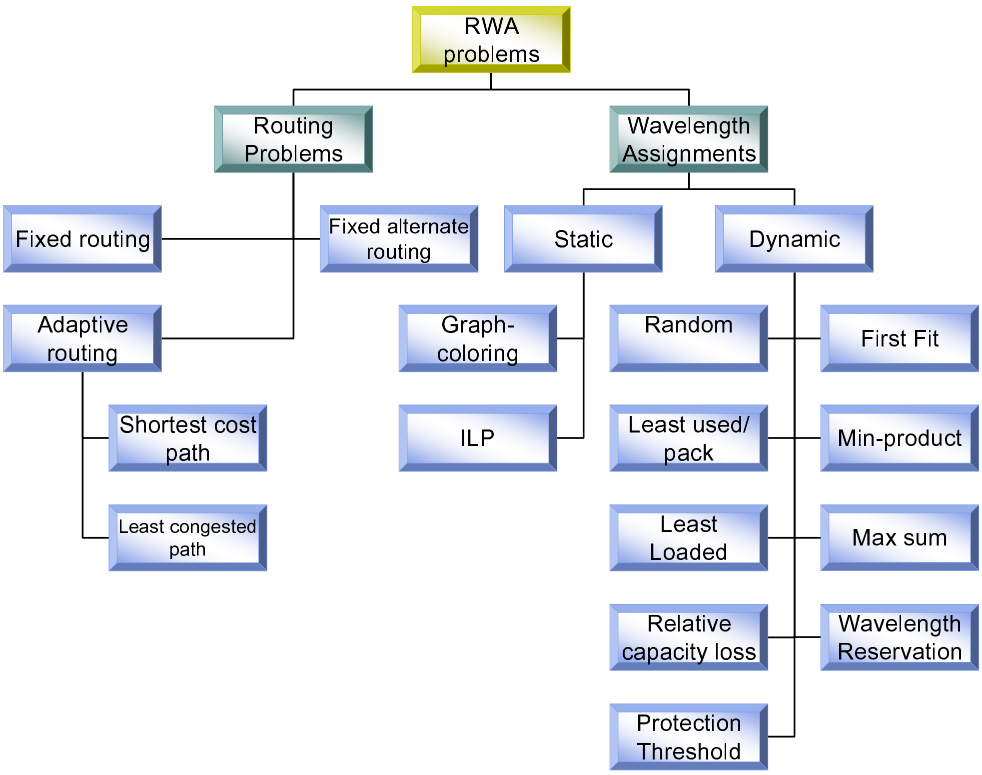



Figure 2.2: *Algorithms for RWA problems*

problem and the wavelength assignment subproblem. For the routing subproblem, there are three well-known approaches in literature: fixed routing, fixed alternate routing and adaptive routing. For the wavelength assignment subproblem, various heuristics and an ILP model have been proposed as shown in Fig 2.2

**Routing problem**

- In fixed routing, the most basic approach to a connection is to select the same fixed path between a source node and the destination node. The fixed shortest path routing approach is one such approach [18]. It pre-determines the routes using standard shortest path algorithms, such as Dijkstra's algorithm or the Bellman Ford algorithm or the modified Dijkstra algorithm. A connection is then set up using the pre-determined route. The fixed shortest path routing is simplest, but it leads to high blocking probability if sufficient wavelength channels are not available. Thus, this approach requires a large number of wavelengths along the path.

- In fixed alternate routing, multi-routes between a source node and the destination node are selected and a source node sets up connection when a connection request arrives. If the connection is not able to find an available wavelength along the shortest path route, a source node attempts to establish a connection along the second shortest path route from an ordered list of a number of fixed routes [18]. Therefore, the fixed alternate routing can reduce the blocking probability compared with fixed shortest path routing. Moreover, this approach is flexible and simple to establish a connection.

- In adaptive routing, the connection is dynamically selected dependant on the current network state such as traffic patterns and channels available. Two well-known approaches are adaptive shortest cost path routing and least loaded path routing. For adaptive shortest cost path routing, each unused link has a cost of 1 unit, a route selects a shortest cost path upon arrival of a connection request. For least loaded path routing, a connection is determined by the number of wavelengths available on a link. The link which has enough available wavelengths is first considered to establish a connection. Both of those approaches always provide shortest paths. If there exist multiple paths with the same distance, one of them is dynamically selected.

**Wavelength assignment problem**

For dynamic wavelength assignment, heuristic approaches are required to assign wavelengths to route a connection. In Fig. 2.2, the first seven methods are to reduce the blocking probability for the new connection. *Wavelength Reservation* and *Protecting Threshold* improve blocking performance for a connection that traverses more than one link [18]. Static wavelength assignment can be formulated as an ILP model to minimize the blocking probability. The ILP formulation is expressed as follows.

Notation:

| | |
|---|---|
| $s$ | source node |
| $d$ | destination node |
| $w$ | wavelength on link |
| $\Gamma_{s,d,w}$ | the traffic demand from source node to destination node on any wavelength |
| $T_{i,j}^{s,d,w}$ | the traffic demand from source node (s) to destination node (d) on link i,j using wavelength |
| $\vartheta_{s,d}$ | the number of connections requested between source node and destination node |

Objective:

$$\min T_{max} \tag{2.1.1}$$

Subject to:.

$$T_{max} \geq \sum_{s,d,w} T_{i,j}^{s,d,w} \qquad \forall i,j \tag{2.1.2}$$

$$\sum_{i} T_{i,j}^{s,d,w} - \sum_{k} T_{j,k}^{s,d,w} = \begin{cases} -\Gamma_{s,d,w} & \text{if } s \text{ is equal to } j \\ \Gamma_{s,d,w} & \text{if } d \text{ is equal to } j \\ 0 & otherwise \end{cases} \tag{2.1.3}$$

$$\sum_{w} \Gamma_{s,d,w} = \vartheta_{s,d} \tag{2.1.4}$$

$$T_{i,j}^{s,d,w} = 0, 1 \tag{2.1.5}$$

$$\sum_{s,d} T_{i,j}^{s,d,w} \leq 1 \tag{2.1.6}$$

### 2.1.2 Wavelength conversion

It is impossible to establish lightpaths between nodes if the available wavelengths on links are different even though there exists a free wavelength. This is because the *wavelength continuity constraint* should be considered to establish any lightpath in the WDM network. This may cause a high blocking probability. However, if the network supports a capability for wavelength conversion (WC) at every node, this problem disappears.

WC is a transformation device to convert the optical signal from the incoming wavelength to the desired wavelength. Fig. 2.3 shows an example of wavelength conversion. In generally, WC is classified into two types: opto-electro-opto (OEO) conversion and all optical conversion [3].

- **optical crossconnects with opto-electro-opto conversion**, which converts signals from the optical domain to the electronic domain and then convert them back to the optical domain.

- **all optical crossconnections**, in which signals remain in the optical domain.

An ideal all optical wavelength converter should contain the following characteristics [6, 30]:

- Simple implementation,

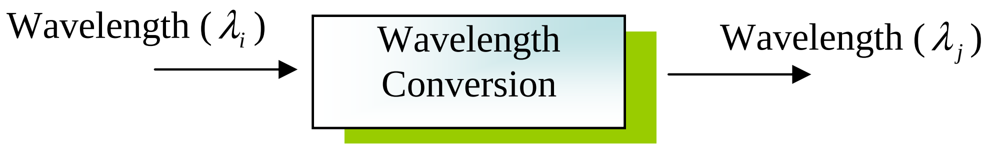



Figure 2.3: *Wavelength conversion*

- insensitivity to input signal polarization,
- possibility for same input output wavelengths,
- conversion of both shorter and longer wavelengths,
- fast setup time of output wavelength,
- transparency to bit rates and signal formats,
- moderate input power levels,
- low-chirp output signal with high extinction ratio and large signal to noise ratio.

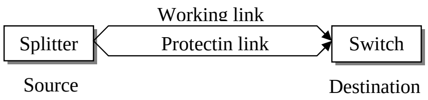
WCs have different wavelength conversion capabilities. There exists full wavelength conversion and limited wavelength conversion [31]. Full wavelength conversion is able to convert any wavelength to any other wavelength without any restriction since WC is available at any node. However the number of nodes in the network may be hundreds or even more. Hence, a huge number of WCs are required. On the other hand, limited wavelength conversion utilizes a small number of wavelength converter to achieve a blocking performance close to full wavelength conversion. However, using a limited wavelength conversion is more complex and it is still unknown how many converters are required to achieve a satisfactory performance [31]. In addition, wavelength conversion increases network cost since this still remains very expensive. Thus, the problem of wavelength converter placement is important to save cost and minimize blocking probability. The optimal placement of converters in WDM mesh networks is an NP-hard problem. Many heuristic algorithms have been proposed to solve the WC placement problem in [32, 33, 34, 35, 36, 37]. An objective of most approaches is to minimize the overall blocking probability using a small number of wavelength converters.

## 2.2 Network Survivability

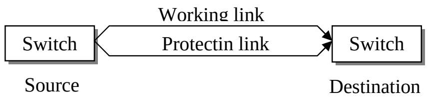
Currently, optical WDM networks are required to satisfy the growing network traffic demands. Therefore, survivable network design against failure in WDM network becomes a critical issue since a channel failure can lead to huge data losses. In this context, each primary path affected by the failure has to be switched to an alternative path. The primary paths are called working paths and alternative paths are called backup paths. Many researchers have studied traditional networks as well as optical networks and several survivability schemes have been proposed so far. The technique for survivability in optical networks can be classified into two categories depending on how the spare capacity is allocated: pre-planned protection and dynamic restoration [38]. We will first discuss the survivability based on these categories. The classification of the survivability techniques is shown in Fig 2.13. Furthermore, we review Shared Backup Path Protection (SBPP) and Pre-configured protection cycle ($p$-cycle), in which the protection paths are known in advance. Specially, $p$-cycle protection is becoming popular since this method can balance between the capacity utilization and computation time as discussed in chapter 1.

### 2.2.1 Pre-planned protection

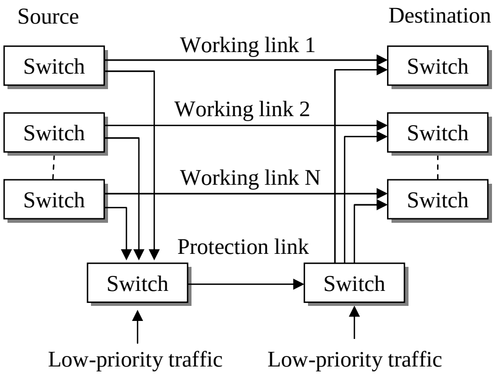
Pre-planned protection reserves backup paths against failure at the same time as the working paths are allocated. All working channels are dynamically switched to reserved paths in the case of failure. Automatic Protection Switch (APS), Self Healing Ring (SHR) network mechanism and mesh network mechanism belong to pre-planned protection since these schemes allocate backup paths in advance. This protection technique is able to offer fast




(a) 1 + 1 APS




(b) 1 : 1 APS




(c) 1 : N APS

Figure 2.4: Automatic protection switching

computation time since backup paths are pre-computed and the nodes on the working path only need to be changed to the nodes on the backup path. However, capacity utilization is a major challenge in these protection scheme, except for the mesh network mechanisms. Unlike the other two schemes, the mesh mechanism offers efficient capacity utilization, but has higher computation time.

**Automatic Protection Switching (APS)**

Automatic Protection Switching is one of the attractive schemes to provide satisfactory quality of service in the networks. The APS can be divided into three different architectures depending on the assignment of protection resources: 1+1 APS, 1:1 APS and 1:N APS. These three models select backup path after link failures. 1+1 APS is a sort of dedicated protection, 1:1 APS and 1:N APS are shared protection.

**a) 1 + 1 APS**

1 + 1 APS transmits the signal of information on a working link and a backup reserved link at the same time. The destination node monitors the two received signals and selects the better one. Upon a single link failure, the receiver at the destination dynamically switches solely to the backup channel to maintain the service. The advantage of this technique is that it is able to offer fast restoration time because of the fastest possible switching speed, which requires 60ms or less [39, 8]. However, 1+1 APS requires large spare capacity and at least 100% redundancy. This is because this technique does not allow sharing of spare capacity for other uses. Fig 2.4(a) illustrates the 1 + 1 APS configuration.

**b) 1 : 1 APS**

1 : 1 configuration offers a reserved channel for each working channel. Unlike 1+1 APS, 1:1 APS only transmits the optical signal on the working channel and the reserved channel does not carry this signal. In addition, the reserved channel can be used to transmit low priority traffic. The low priority traffic is redirected and the working channel switches to the reserved channel when a single link failure occurs. For this reason, this method is better than 1+1 APS in terms of network capacity utilization. On the other hand, its restoration time is higher than 1+1 APS since the working channel must switch to the reserved channel after a failed link is detected. Fig 2.4 illustrates the 1 : 1 APS configuration.

**c) 1 : N APS**

Fig 2.4(c) shows a shared link protection scheme, the 1 : N configuration. In this configuration, N working channels share a single reserved channel to recover against single link failures of any of the N working links. The shared link protection only covers the failure of any one of the N working links and thus the traffic has to be switched back to the primary working link after the failure is repaired. The reserved link is able to only protect any one failure of the working channels. Consequently, 1 : N APS configuration offers efficient capacity utilization but it has high blocking probability when a protected link fails. An extension of 1:N APS, the k:N configuration, in which k reserved channels are available, is more popular.

**Self Healing Ring (SHR)**

In optical networks, a ring topology, known as *Self Healing Ring* (SHR), is the simplest protection network and has been suggested in [39, 40]. SHR can easily provide 100% restorability and offer faster computation time because of simple control management and simple routing policy. Hence, it is a promising architecture in reality. On the other hand, the main problem with ring networks are that these require at least 100% redundancy, and sometimes over 200% $\sim$ 300% [41]. The basic idea of SHR is that half of total capacity is reserved for working capacity while the other half is used as spare capacity. Upon a failure of a single link, the working channel is rerouted along a backup channel in the opposite direction. The SHR is classified into two categories depending on the direction of traffic: unidirectional SHR (USHR) and bidirectional SHR (BSHR) [38]. The configuration of these SHR is shown in Fig 2.5.

**a) Unidirectional SHR (USHR)**

USHR contains two optical ring fibers, where one transmits the primary signal as the working channel and another transmits the backup signal as the reserved channel. When a network failure occurs, the working channel is dynamically switched to the protection ring for the backup channel. The backup signal is rerouted in the opposite direction to the primary signal. Also, the primary signal from source to destination node is transmitted on one working fiber ring and the return signal travels on the opposite side of that ring in the same direction [42]. An example of USHR is shown in Fig 2.5(a).

Two type of configuration in USHR are path protection switched USHR (USHR/P) and line protection switched USHR (USHR/L) as illustrated in Fig 2.5(b) and 2.5(c), respectively. The difference between them is the rerouting process. USHR/P transmits working traffic in both the working path and the protection path, and the receiver at the destination chooses the stronger signed out of the two, to maintain the service in the event of failure. Thus, USHR/P is the fastest but only used to transmit low traffic demand. On the other hand, in USHR/L, the node-pairs of a failure are switched to the protection channel to protect the traffic affected by a failed link.

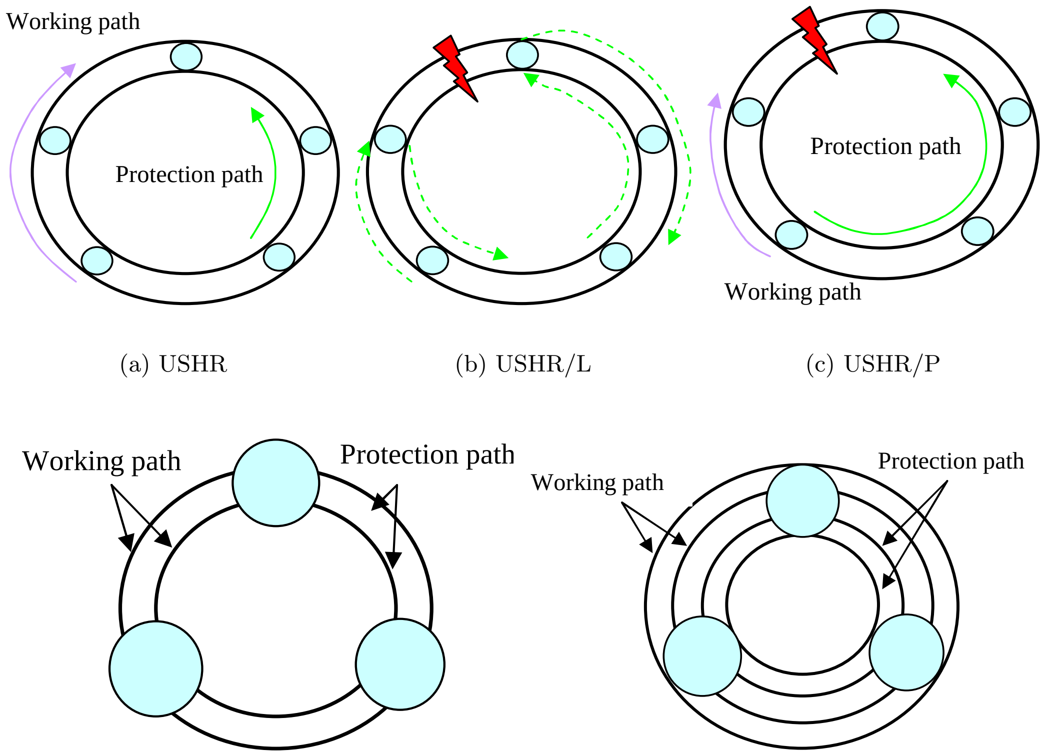
**b) Bidirectional SHR (BSHR)**

The BSHR is divided into two architectures depending on the number of fiber channels in the ring: BSHR/2 and BSHR/4. BSHR/2 is defined as two fiber channels protection. Half of the capacity on each ring is used to serve the working capacity and the other half is for the spare capacity. This means that the wavelength channel of each fiber is divided into two equal parts [42]. Fig 2.5(d) shows two fiber channels protection configuration in BSHR. On the other hand, in four fiber channels protection (BSHR/4), two fiber rings are reserved for working channel and the other two fiber rings are for the protection channels as shown in Fig 2.5(e). When a single link failure occurs, the two end nodes of a failed link are reconnected to the reserved channel.

From the literature review, BLSR is more efficient in terms of network capacity than USHR. However, capacity utilization in SHR is one of the performance metrics, which poses conflicting requirements, as discussed. These schemes require high redundancy but offer fast computation time, which is generally around 50ms. Thus, many researchers have investigated a balance between capacity utilization and computation time [43, 44].

### Mesh network based survivability

Despite the fact that the most popular physical topology is the SONET Self-Healing Ring (SHR) network, mesh networks are becoming important. This is because the mesh topology is preferred for large networks while ring topology is for simple and small-scale networks. In mesh network survivability, it is more complicated to solve the optimization problem




(a) USHR (b) USHR/L (c) USHR/P

(d) BSHR/2 (e) BSHR/4

Figure 2.5: The self-healing ring configuration

since a higher number of routing and wavelength channels must be considered. But, mesh network survivability offers high capacity utilization because spare capacity can be shared to protect against network failures so that this network mechanism is preferred more from an efficiency point of view. Mesh network based survivability is classified into two categories depending on the re-connecting process: path protection and link protection.

**a) Path protection**

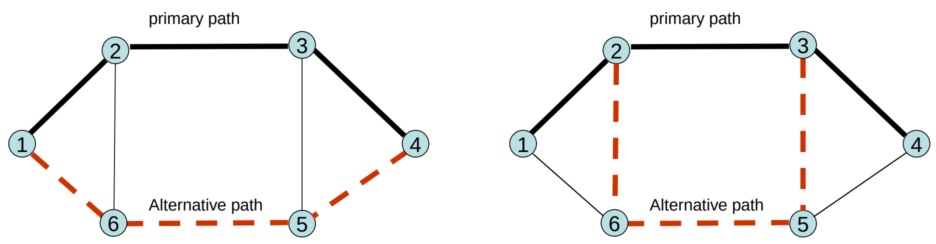
In path-based survivability, the fault notification message informs the source node and the destination node of each path that traverses the failed link. When network failure occurs, the working path is switched to the reserved path which should be disjoint from the corresponding working path. The reserved path is allocated between the end node pairs of path. For example, failed link (2-3) is recovered by the reserved path (1-6-5-4) as shown in Fig 2.6(a). Path based survivability is more efficient in capacity utilization compared to link based survivability, since it only needs spare capacity for the whole reserved path instead of every link along the path [45].

**b) Link protection**

In this method, all alternative paths have already been reserved when the working path is computed. Upon a link failure, the end node pairs of the failed link are immediately switched to the reserved path. In link based survivability, restoration time is faster than path based survivability since path mechanism require a longer time to generate a fault notification message. Fig 2.6(b) illustrates an example of link based survivability. The node pairs of the failed link (2-3) are automatically switched to the reserved path (2-6-5-3).

### 2.2.2 Dynamic restoration

Dynamic restoration dynamically discovers backup paths in the network after a link fails. Typically, this is more efficient in term of capacity utilization than pre-planned protection since it is not necessary to reserve spare capacity. On the other hand, computation time is a




(a) path protection (b) link protection

Figure 2.6: Two basic survivability mechanisms in mesh networks

challenging problem to tackle in dynamic restoration. This mechanism can be classified into path restoration and link restoration. Path restoration is more efficient than link restoration in terms of spare capacity utilization. As a comparison of computation time between path restoration and link restoration, S. Ramamurthy *et al.* [8] shows that link restoration is faster than path restoration.

**a) Path restoration**

In this method, an alternative path is immediately determined from the source node to the destination node when a failure occurs. This technique is not able to guarantee 100% restorability since the path for recovery may be blocked easily when the network capacity is exhausted at failure time. Hence, various researchers have investigated how to find a solution to this problem. In addition, computation time (T) is an important issue and should be given due consideration. It is calculated as [8]:

$$T = F + (m + 1)C + 2(m + 1)D + 2mP. \tag{2.2.1}$$

where $m$ is the number of hops in the restoration path, $F$ is the time to detect a link failure, $C$ is the time to configure, test and setup an OXC, $D$ is the time to process a message at a node, and $P$ is the propagation delay on each link.

**b) Link restoration**

In link restoration, spare capacity is reserved at the time of failure. It dynamically discovers a backup channel around the adjacent nodes of the failed link. Thus, this offers efficient capacity utilization. Network failure can be recovered through any number of reserved channels. The computation time has to be discussed to serve the quality of service in link restoration and the time formulation is given as [8]:

$$T = F + nP + (n+1)D + (m+1)C + 2mP + 2(m+1)D \tag{2.2.2}$$

where $n$ is the number of hops from the source-end node of the failed link to source node of the connection.

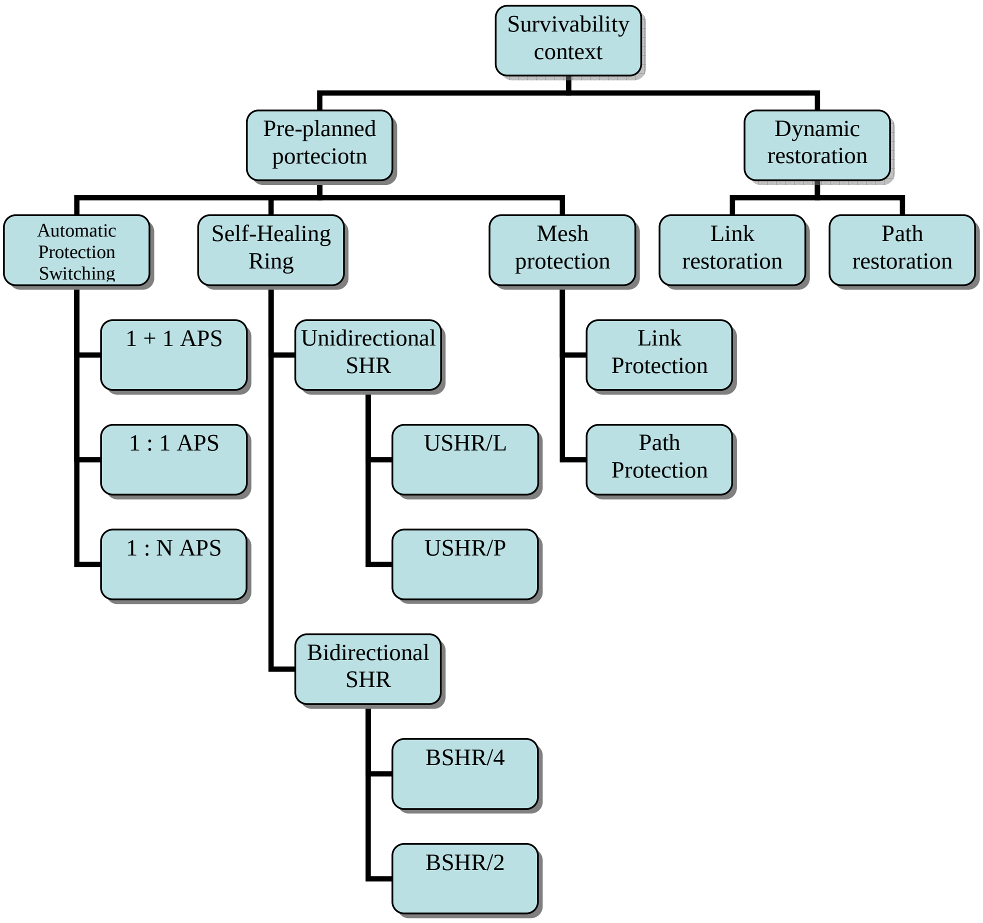



Figure 2.7: *The classification of survivability techniques in optical networks*

### 2.2.3 Shared Backup Path Protection (SBPP)

Shared Backup Path Protection is similar to the 1 + 1 APS scheme in that two disjoint paths, a working and backup path, are routed to transmit the optical signal. However, this scheme is able to share spare capacity over the backup channels from multiple services [46]. For this reason, SBPP can offer efficient capacity utilization. In SBPP, one or more backup paths are found between the source and the destination node of the primary working path, but only one backup path is selected to protect against network failures. The backup path has to be link and node disjoint from the primary path. This is because link or node on a working path may be affected when that working path fails. Fig 2.8 shows a set of three working paths that can share spare capacity in SBPP. Each working path has a backup path that is pre-computed and a number of backup paths can share spare capacity on some spans. For example, every working path shares link E-F for backup in Fig 2.8, this is possible since they are disjoint with their corresponding primary paths. Also, link E-G shares two backup paths. The spare capacity is pre-allocated to protect against network failure.

In the event of a single link failure, the nodes on the working path have to re-configure to switch to the backup path, which affects the service restoration time. Typically, the restoration time is about 200 ms [13, 47]. SBPP is one of the survivability techniques which offers great capacity efficiency, but with a slow computation time.

### 2.2.4 Pre-configured protection cycle

Pre-configured protection cycle (*p-cycle*) is classified as link protection in mesh networks. The basic idea of *p*-cycles is to recover failures using ring network mechanism in a mesh network. This mechanism is a promising approach for solving the network design problem in the context of survivability.

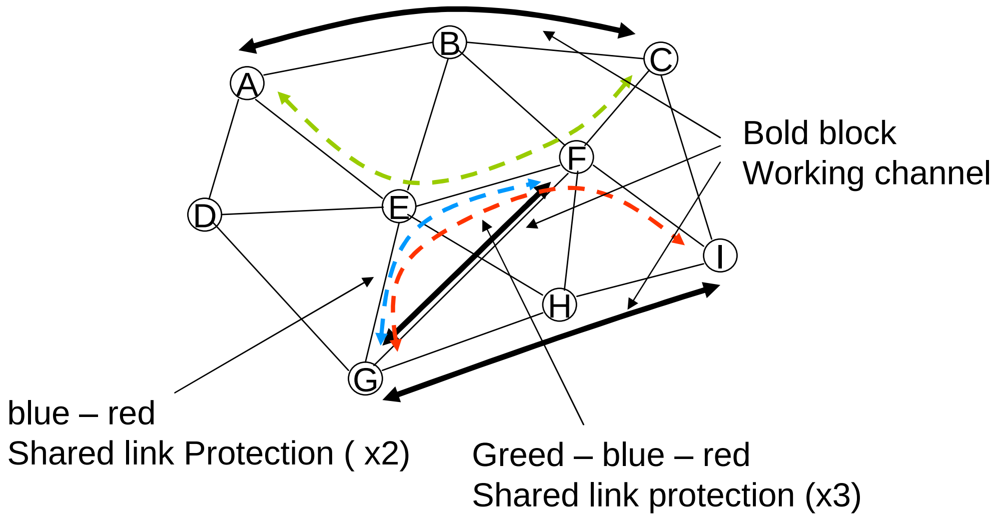



Figure 2.8: *Shared backup path protection*

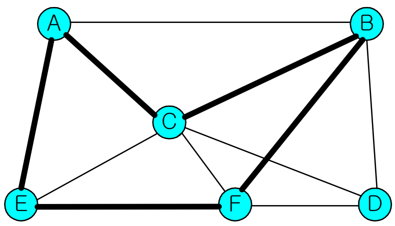
### Basic definitions of *p*-cycle

The basic concept of *p*-cycle has been proposed by Grover and Stamatelakis in 1998 [11, 12, 48]. It has been developed as a hybrid of the ring and the mesh protection mechanisms. It benefits from the fast restoration time of ring protection and the capacity efficiency of mesh topologies. The *p*-cycle method is based on closed cyclic routes. However, unlike the ring mechanism which only protects the working channels on the ring, the *p*-cycle method offers useful backup paths for protecting the straddling links as well as the on-cycle links. A straddling link of a *p*-cycle is a link which does not belong to that cycle but whose end-nodes lie on the *p*-cycle. For example, in Fig 2.9(a), spans C-E and C-F are straddling links of *p*-cycle (A-C-B-F-E). The *p*-cycle method is able to offer two restoration paths for straddling links without requiring any spare capacity. As a result, this technique is a reliable and effective method for utilization of capacity in optical mesh networks. Table 2.1 presents a comparison of the ring, mesh and *p*-cycle protection mechanisms.

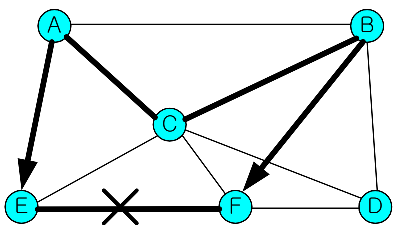
Fig 2.9 shows an example of *p*-cycle contribution. Fig 2.9(a) illustrates a *p*-cycle (A-B-C-D-E-F). An alternative path for the failure of an on-cycle link provides a single restoration

| | **Ring** | **Mesh** | ***p*-cycle** |
|---|---|---|---|
| Restoration time | 50 - 60msec | 100msec - 2sec | 50 - 60msec |
| Redundancy | 100% or more than 200% | 50 - 70 % | 50 - 70 % |
| Network design | simple | complex | simple |
| Capacity efficiency | low | High | High |
| Cost | low | High | low |

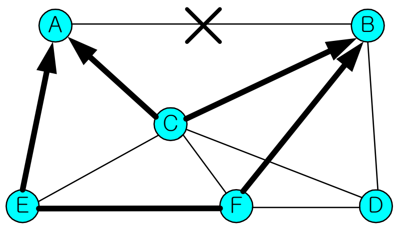
Table 2.1: Comparing Ring, Mesh and $p$-cycle




(a) An individual $p$-Cycle




(b) Failure of on-cycle link




(c) Failure of Straddling link

Figure 2.9: Example of $p$-cycle contribution on a alternative path

path as shown Fig 2.9(b). But when straddling link (E-F) fails, the two nodes spanning the failed link are switched to the alternative path and capacities are reserved in both directions along the cycle by the $p$-cycle. In addition, the $p$-cycle method provides two alternative paths for that failure as shown in Fig 2.9(c).

$p$-Cycle protection offers useful restoration paths depending on the relationship to the failed link. Searching suitable paths is an important issue in $p$-cycle protection since efficient paths can offer better capacity efficiency. The notation $u_{i,j}$ presents the number of useful paths, where the $j^{th}$ $p$-cycle offers paths for working capacity when the failure of span $i$ occurs. The number of useful path $p$-cycles is summarized below [13]:

- $u_{i,j} = 0$ *if one or both end nodes of span $i$ are not nodes of cycle $j$.*
- $u_i, j = 1$ *if both end nodes of span $i$ are nodes of cycle $j$ and span $i$ is not on the cycle.*
- $u_{i,j} = 2$ *if both end nodes of span $i$ are nodes of cycle $j$ and span $i$ is not on the cycle.*

### $p$-Cycle in WDM optical networks

$p$-Cycle protection makes it possible to minimize the redundant capacity by determining an appropriate set of $p$-cycles. However, determining an efficient and sufficient set of $p$-cycles is difficult. Many researchers have investigated how to construct suitable candidate $p$-cycles for solving this optimization problem. Two versions of the optimization have been investigated the non-joint version and the joint version [49]. The objective of the joint version is to minimize the total capacity. It minimizes working capacity and spare capacity jointly by using $p$-cycles while maximizing the restorability. On the other hand, the non-joint version minimizes the working capacity and spare capacity separately. That is, after the distribution of working capacity in the non-joint version is known, a set of candidate $p$-cycles is computed to minimize spare capacity with maximum protection capability. According to the literature, the joint version of optimization may achieve better capacity utilization but has a higher complexity and requires a longer computation time [50]. Currently, ILP models and heuristic methods are being studied and investigated to solve this optimization problem.

**ILP formulations**

The ILP formulation may be of two types: non-joint optimization and joint optimization. For non-joint optimization, the ILP model has two basic formulations for designing $p$-cycle networks [51, 48]. The first formulation is to maximize restorability within a given capacity and placement of spare capacity. It has been shown that results of close to 100% protection are achievable. The parameters in the models are as follows [48]:

| | |
|---|---|
| $E$ | the set of network spans. |
| $P$ | the set of all candidate cycles in the network. |
| $z_j$ | the number of unrestorable working channels on span $j$. |
| $w_j$ | the number of working wavelength channels on span $j$. |
| $s_j$ | the number of spare wavelength channels on span $j$. |
| $x_{j,k}$ | the number of useful path cycles $k$ can cover after failure of span $j$. |
| $R_j$ | the number of unprotected working channels on span $j$. |
| $n_j$ | the number of unit capacity copies of cycle $j$. |
| $u_{j,k}$ | 1 if span $k$ is on cycle $j$, 0 otherwise. |
| $a_j$ | the number of available protection paths on span $j$. |
| $q_j$ | the cost of span $j$. |

Maximizing restorability with $p$-cycles is expressed as:

$$\min \sum_{j=1}^{|E|} z_j \tag{2.2.3}$$

subject to:

$$s_j \leq \sum_{k=1}^{|P|} x_{j,k} \cdot n_i, \qquad \forall_k = 1, 2, ..., E \tag{2.2.4}$$

$$R_j + \sum_{k=1}^{|P|} u_{j,k} \cdot n_k = w_j + a_j, \qquad \forall_j = 1, 2, ...E \tag{2.2.5}$$

$$0 \leq R_j \leq w_j \qquad \forall_j = 1, 2, ...E \tag{2.2.6}$$

$$n_k \geq 0 \qquad \forall_k = 1, 2, ...E \tag{2.2.7}$$

$$R_j \geq 0 \qquad \forall_j = 1, 2, ...E \tag{2.2.8}$$

The second formulation determines a set of optimal candidate $p$-cycles with minimum spare capacity. The objective of the model is to minimize spare capacity for 100% $p$-cycle restorability [48]. This formulation can basically offer the optimal solution. The objective function here is:

$$\min \sum_{j=1}^{|E|} q_j s_j \tag{2.2.9}$$

subject to:

$$s_j = \sum_{k=1}^{|P|} x_{k,j} \cdot n_k, \qquad \forall_k = 1, 2, ..., E \tag{2.2.10}$$

$$w_j \leq \sum_{k=1}^{|P|} x_{j,k} \cdot n_k, \qquad \forall_j = 1, 2, ...E \tag{2.2.11}$$

$$n_k \geq 0 \qquad \forall_k = 1, 2, ...P \tag{2.2.12}$$

$$s_k \geq 0 \qquad \forall_k = 1, 2, ...E \tag{2.2.13}$$

However, the main problem with ILP is that it requires all possible cycles, which can be a very large number in dense and large scale networks. Thus, the computational complexity in ILP formulation is a complex issue. To overcome this problem, Dominic A. Schupke [52] has proposed a new ILP formulation for non-joint optimization without enumeration of candidates. But the model is very complex, thus they also suggests a four-step heuristic which makes the calculation tractable in the model.

For joint optimization, H. N. Nguyen has proposed a $p$-cycle formulation that can be solved with ILP [53]. The model is based on a new definition of fundamental cycles and straddling links, which are classified into *visible*, *hidden* and *non-shareable straddling links*. *Visible straddling links* are created by jointing two different fundamental cycles, which have only a single common link between them. *Hidden straddling links* are also created by joining two fundamental cycles or more, but the straddling links are not any part of those cycles. *Non-shareable straddling links* is straddling links which disappear by joining fundamental cycles. The results show that the proposed formulation is able to reduce complexity and achieve the required optimality of solution. In the model, $C = \{c_1, c_2, .., c_C\}$ is a set of fundamental cycles in the network and $S = \{s_1, s_2, .., s_S\}$ is defined as the number of *visible straddling links*. $I = \{i_1, i_2, .., i_I\}$ is a set of *hidden straddling links* and $D = \{d_1, d_2, .., d_D\}$ is the set of demands. The set of path candidates between pair-nodes of demands is defined by $P = \{p_1, p_2, .., p_P\}$. $\Lambda = \{\lambda_1, \lambda_2, .., \lambda_n\}$ is set of non-sharable straddling links. The ILP formulation is expressed as [53]:

Constants:

$$\delta_{x,j} = \begin{cases} 1 & \text{If cycle } x \text{ includes span } j;\ j \in \text{E}; x \in C; \\ 0 & \text{otherwise.} \end{cases}$$

$$\xi^{j}_{s,c} = \begin{cases} 1 & \text{If } c = \{x, y | x \cap y = j; x, y \in C\}; j \in E; c \subset C; s \in S; \\ 0 & \text{otherwise, } s \text{ is the } s_{th} \text{ straddling link formed by cycle } x, y. \end{cases}$$

$$\pi^{j}_{i,\theta} = \begin{cases} 1 & \text{If } s \cap E(\theta) = \emptyset \text{ and } V(s) \subseteq V(\theta); j \in E; \theta \subseteq C; i \in I; \\ 0 & \text{otherwise, } i \text{ is the } i_{th} \text{ straddling link formed by set of cycles } \theta. \end{cases}$$

$$\tau^{d}_{i,j} = \begin{cases} 1 & \text{If candidate path } i_{th} \text{ of demand } d \text{ cross span } j; \\ 0 & \text{otherwise.} \end{cases}$$

| | |
|---|---|
| $v_{s,j} = 2$ | the number of useful paths provided by hidden straddling link $s$ to restore span $j \in E; s \in S$ |
| $\alpha_j$ | the cost per channel on span $j \in E$; |
| $h_d$ | volume of demand $d$; $d \in D$ |
| $\phi_j$ | maximum capacity provided by span $j \in E$ |

Objective:

$$\min \sum_{j\in E} \alpha_j (y_j + w_j) - \sum_{i\in S} \sum_{j\in E} 2\xi_{i,c}^{j} u_{i,x}, \qquad x \in c, c \subset C \tag{2.2.14}$$

subject to:

$$\sum_{i\in C} \delta_{x,j} n_x = y_j, \qquad \forall_j \in E \tag{2.2.15}$$

$$\sum_{i\in S} \xi_{i,c}^{k} u_{i,x} - n_x \leq 0 \tag{2.2.16}$$

$$\sum_{j\in I} \pi_{j,c}^{k} m_{j,x} - n_x \leq 0 \qquad \forall_c \subset C; x \in c; k \in E \tag{2.2.17}$$

$$\sum_{i\in P} \sum_{d\in D} p_i^d \tau_{i,j}^d = w_j, \qquad \forall_j \in E \tag{2.2.18}$$

$$\sum_{i\in I} v_{i,j} m_{i,x} + y_j \geq w_j, \qquad \forall_j \in E; x \in C \tag{2.2.19}$$

$$\sum_{i\in S} 2\xi_{i,c}^{j} u_{i,x} - y_j \leq 0, \qquad \forall_j \in E; x \in c; c \subset C \tag{2.2.20}$$

$$(\sum_{i\in S} \xi_{i,c}^{m} u_{i,x} + \sum_{j\in I} \pi_{j,c}^{n} m_{j,x}) - n_x \leq 0, \qquad \forall \{\xi_{i,c}^{m} \pi_{k,c}\} \subseteq \bigwedge; \forall_x \in c; c \subset C; m, n \in E \tag{2.2.21}$$

$$\sum_{i\in P} P_i^d = h_d. \qquad \forall_d \in D \tag{2.2.22}$$

$$\sum_{i\in S} -s\xi_{i,c}^{j} u_{i,x} + y_j + w_j \leq \phi_j, \qquad \forall_j \in E; x \in c; c \subset C \tag{2.2.23}$$

where $y_j$ is the capacity on span $j$ that can offer the cycle; $j \in E$. $n_x$ is the number of unit capacity copies of the cycle $x$; $x \in C$ and $m_{i,x}$ is a set of number of unit capacity copies of the hidden straddling link $i$ created by cycle $x$; $i \in I; x \in C$. $u_{i,x}$ is a set of unit capacity copies of the visible straddling span $i$ created by cycle $x$; $i \in S; x \in C$. $W_j$ is the traffic demand on span $j$ to support the routing of working paths; $j \in E$. $p_i^d$ is a set of unit capacity copy of the $i_{th}$ path candidate chosen to serve demand $d$; $i \in P; d \in D$.

**Heuristic approaches**

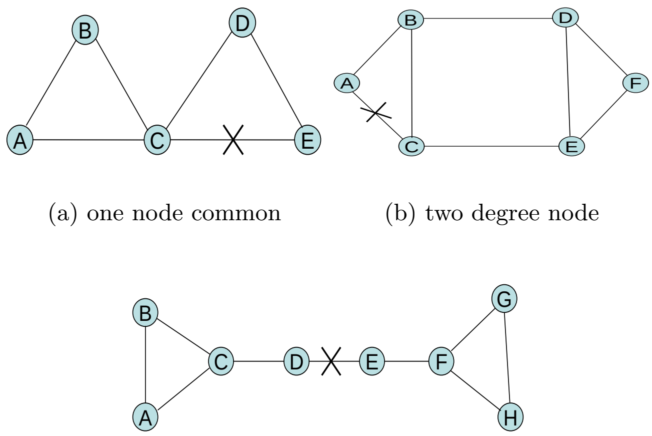
Heuristic approaches can be classified into two types: heuristic approaches based on an ILP model and pure heuristic approaches. The heuristic approach based on ILP first determines a set of limited candidate $p$-cycles and then the ILP formulation is applied to ensure 100% protection. This approach can achieve near optimal solutions, but it is still an optimization problem with the associated computational complexity of ILP. The pure heuristic approach is designed to achieve best performance without using ILP model. This approach can also achieve near optimal solutions within acceptable computational complexity.

In order to reduce the number of candidate $p$-cycles by computing high efficiency cycles, a 'pre-selection' heuristic approach has been investigated in [50] namely *a priori efficiency (AE)*. In this context, a high efficiency cycle is defined as a cycle with high potential capability to protect against network failure. The $AE$ is expressed as:

$$AE(p) \triangleq \frac{\sum\limits_{\forall p \in S} X_{p,i}}{\sum\limits_{(\forall p \in S | X_{p,i}=1)} c_i} \tag{2.2.24}$$

where $S$ is the set of spans in the network, $i$ is a span in $S$. $X_{p,i}$ is the protection potential related to span $p$; $X_{p,i} = 1$ when span $i$ is an on-cycle span and $X_{p,i} = 2$ if span $i$ is a straddling span, $c_i$ is the cost of capacity on span $i$. The $AE$, which is defined as ratio of total working capacity to the total span cost for a cycle, is a measure of the efficiency of $p$-cycle. Theoretically, a $p$-cycle with a high AE value has a high efficiency for protecting the working capacity. However, a 'pre-selection' approach may not consider network circumstances, such as traffic load on the network and nodal degree. As a result, this scheme sometimes leave a lot of wasted working capacity on some spans, not leading to a good solution for protection.

Zhang *et al.* [54] has proposed a Straddling Link Algorithm (SLA) for link protection. A key point is to generate the initial candidate $p$-cycles of the network graph. This approach first finds the shortest path between two end nodes of a span and then searches other shortest path between the same node-pairs that are node disjoint from the previous path.




(a) one node common (b) two degree node

(c) bridge topology

Figure 2.10: Considerations in searching $p$-cycles by SLA

It then constructs a cycle by combining the two shortest node-disjoint paths (each path contains no common intermediate nodes). SLA can only construct a set of $p$-cycles with one straddling span. To generate a set of candidate $p$-cycles in SLA, it prunes off nodes of nodal degree 1 since no cycle could contain such nodes. In addition, it is not able to construct a $p$-cycle in the following network conditions.

- case 1: If a link (C-E) fails, there does not exist two node-disjoint paths as shown in Fig. 2.10(a) since there is only one node common to each path.
- case 2: If a link (A-C) fails, there is one or more paths between the same node. But it is not able to generate $p$-cycles because there is not more than one node-disjoint path, as shown Fig. 2.10(b).
- case 3: If a link (D-E) fails, it is not able to compute a $p$-cycle since there is no path between two nodes at all as shown Fig. 2.10(c).

SLA is simple and fast for generating candidate $p$-cycles, however the number of $p$-cycles is insufficient since only $O(m)$ $p$-cycles are generated. Furthermore, a set of candidate $p$-cycles generated by SLA is inefficient since higher efficiency cycles are not contained in the set and thus this approach is not able to satisfy minimum spare capacity. The pseudocode of the SLA is shown as Algorithm 1.

**Algorithm 1 : Straddling Link Algorithm (SLA)**

```
Input: network topology
Output: Set of candidate p-cycles
Check network condition
S ← number of spans in the network
for i = 1 to |S| do
  Search two shortest disjoint path between two end nodes of span i
  Determine a p-cycle using the two paths found.
end for
```

Another scheme, called a Weighted DFS-based cycles Search (WDCS), has been proposed by Liu and Ruan [49]. The objective of this algorithm is to generate good candidate cycles. WDCS contains two kinds of cycles, one is a high efficiency set of cycles and the other is two sets of short cycles. The high efficiency cycles are generated by the Depth First Search (DFS). DFS in this algorithm selects a neighbor node ($n$) with the highest weight among all outgoing edge from $n$, instead of picking any available neighbor node to extend the path. A set of weights of the directed edges are based on three considerations as follows:

- Large cycles are of higher efficiency since these cycles includes more straddling spans. To avoid going back to the root vertex, this approach assigns a small weight to the directed edges that end at the root. It sets $[weight(s, d) = \Xi]$ for all $d \in N(s)$, where $0 < \Xi < 1$ and s is the root. $N(s)$ is a set of neighboring nodes of node $s$.

- A node with high degree is first selected by DFS and a cycle which includes this node has a high probability of including more straddling edges.

- Nodes of degree 2 are not desirable since they are not able to offer straddling spans.

However, 2 nodal degree nodes are sometimes desirable. For example, a cycle ($r - v_1 - v_2 - r$) with AE of 1 is found by WDCS as shown in Fig 2.11(a). Basically, WDCS finds high efficiency cycles so that they offer special treatment for 2 nodal degree node. The cycle is constructed by extending the path. As shown in Fig 2.11(b), if $v$ is included into the searching path from $v_1$ and node $v$ is able only to extend the path to $v_2$, the nodes with 2 nodal degree can be included into the search path. Therefore, they can yield the higher efficiency cycles.

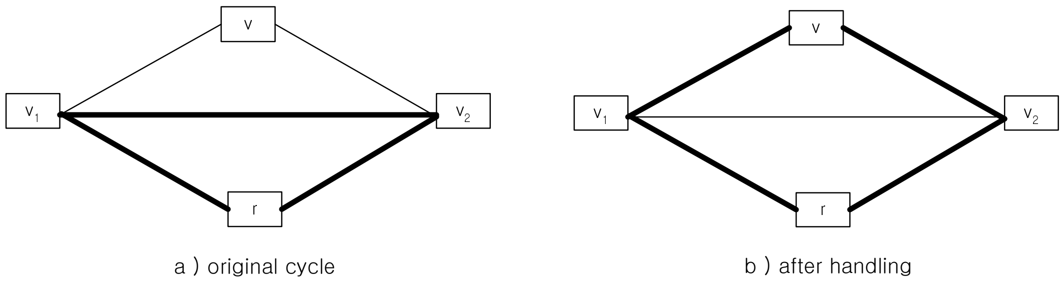



Figure 2.11: *Special handling of nodal degree 2*

In addition, each link has two short cycles. For one of these cycles the link will be an on-cycle link and for the other cycle it will be a straddling link. To generate these cycles, they find two shortest disjoint paths between the end nodes of each span. If they find two paths, they can construct two short cycles. If only one path is found, they only construct an on-cycle link. The number of cycles in WDCS is determined by an input parameter k. The set of candidate $p$-cycles is evaluated based on ILP with uniform and random traffic capacity in their simulation. The results show that a close to optimal solution is able to be achieved when the input parameter k, being the number of shortest paths, is increased to 43. Thus, their method trades off between the number of $p$-cycles and the capacity efficiency. Furthermore, this scheme may not take some important network circumstances into account, such as the distribution of the given working capacity. It only generates candidate $p$-cycles based on the network topology and a suitable set of candidate $p$-cycles is not necessarily generated by this algorithm.

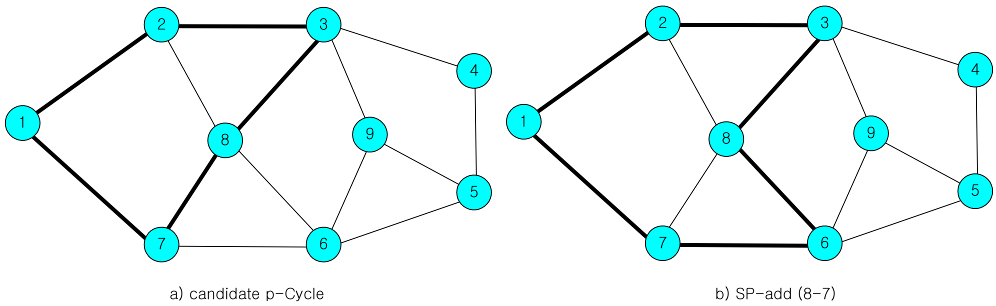
Other heuristic approaches, which are not dependant on the ILP model, have been proposed [55, 56, 57]. Different qualities of performance are achieved depending on the flexibility of candidate $p$-cycles selection after enumerating the set of $p$-cycles. A popular heuristic algorithm used is Capacitated Iterative Design Algorithm (CIDA) proposed by Doucetee *et al.* [55]. CIDA evaluates cycle efficiency using *acutal efficiency* $E_W(p)$, which is defined as the ratio of total protected capacity to the total span cost for each cycle. $E_W(p)$ is a modified version of $AE$ and is expressed as :

$$E_w(p) = \frac{\sum_{\forall i \in S} w_i X_{p,i}}{\sum_{(\forall i \in S | X_{p,i}=1)} cost_i} \tag{2.2.25}$$

where $w_i$ is the amount of unprotected working capacity on span $i$ at that time. CIDA constructs a primary set of candidate $p$-cycles using the SLA approach. Then, more complex $p$-cycles are generated by three algorithms, namely 'SP-add', 'Expand' and 'Grow'.

The 'Grow' and 'Expend' algorithm are based on the 'SP-add' algorithm. 'SP-add' constructs $p$-cycles by using the shortest path between pair-nodes of a selected span on a $p$-cycle. For example, if span 7-8 is selected as shown in 2.12(a), 'SP-add' searches for the disjoint shortest paths between a pair of nodes of the span. The shortest path is replaced on the original cycle to make a more complex $p$-cycle as shown in 2.12(b). Then, other cycles of the primary set are processed in similar fashion. But, the 'Expand' algorithm continues searching for shortest paths of other spans in the same cycle. That is, it continues seeking until every span in the original cycle is visited. The shortest paths of every span should be a disjoint path from every previous route. The 'Grow' algorithm is similar to 'Expend', except that the operation moves to the first span to find a new shortest path after a $p$-cycle is generated.

After a set of $p$-cycles has been constructed, CIDA selects a cycle with high $E_w(p)$ among a set and then removes the working capacity protected by that selected cycle. The




(a) Candidate $p$-cycle (b) SP-add (7-8)

Figure 2.12: *Operation of SP-add*

remaining working capacity is updated and the process is repeated until all working capacity is protected. Unlike AE, this formulation is calculated according to the distribution of the working capacity of each span on a cycle. Hence, this approach can achieve a better performance than AE. Finally, the paper reports that the 'Grow' algorithm based on CIDA is the most efficient. The pseudocode of CIDA is shown in Algorithm 2.

**Algorithm 2 : Capacitated Iterative Design Algorithm (CIDA)**

**Input:** set of candidate $p$-cycles, unprotected working capacity
**Output:** Set of $p$-cycles
$W \leftarrow$ unprotected working capacity
**while** Sum of $W$ is not zero **do**
  calculate $E_w(p)$ of candidate $p$-cycles using 2.2.25
  select maximum $E_w(p)$ of a set
  remove the working capacity of the selected $p$-cycle from $W$
**end while**

An *ER-based unity-p-cycle* algorithm has been proposed by Zhang *et al.* [56]. This algorithm considers unidirectional $p$-cycles and the traffic in the network. First, they construct all possible cycles in the network and then compute the efficiency ratio (ER), which is the ratio of the number of actual protected working units to the number of spare units of a unity-$p$-cycle. A *unity-p-cycle* with a high ER is a more efficient cycle. Therefore, they select the maximum *ER-based unity-p-cycles* to reduce the unprotected capacity until all working capacity is protected. It is found that they are able to achieve near optimal solution without the use of an ILP model.

Zhang *et al* [57] has proposed a heuristic algorithm to minimize the total spare capacity. The proposed algorithm selects cycles according to the redundancy of a $p$-cycle, which is the ratio of the spare capacity of the cycle to the working capacity protected by the cycle. The redundancy ($\Re$) is expressed as [13]:

$$\Re = \frac{\sum_{i \in S} p_i}{\sum_{i \in S} w_i} \tag{2.2.26}$$

where $w_i$ is the number of units of working capacity on span $i$ and $p_i$ is the corresponding number of spare capacity units on span $i$. A cycle with a small redundancy is more efficient for protecting the working capacity. However, when the working capacity protected by a cycle is zero, the algorithm is in trouble since the redundancy is infinity in this situation. The heuristic algorithm is summarized below:

- step 1. Generate all cycles in the network.
- step 2. For each candidate cycle, calculate $\Re$ by 2.2.26.
- step 3. Select maximum $\Re$ of a set.
- step 4. Remove the working capacity protected by the selected cycle and update the working capacity.
- step 5. Go back to step 3 until the working capacity on every span is zero.

In another work related to limiting the $p$-cycles, Schupke *et al* [58] introduce a new approach that limits the circumference of the largest cycle. This approach limits the physical length of the $p$-cycle, and circumference limitation reduces the redundancy of a $p$-cycle by reducing the spare capacity consumption. The paper reports that a redundancy of 34% is achieved when the $p$-cycle length is limited to 6000km. Hence, longer $p$-cycles can achieve a better efficiency. However, the computation time increases dependent upon the maximum allowed $p$-cycle length. Kodian *et al* [59] also applies circumference and hop limitation to generate a set of the $p$-cycles. With hop limit, the total cost is reduced and much capacity efficiency can be achieved. The paper shows that it is quite possible to design efficient $p$-cycle networks, but the algorithm has a problem on how to best control the limitation on the path length.

### Hamiltonian *p*-cycle

Hamiltonian $p$-cycles have been investigated by Stamatelakis and Grover [60], but they still use $p$-cycles of a variety of circumferences in the efficiency generalized designs. A

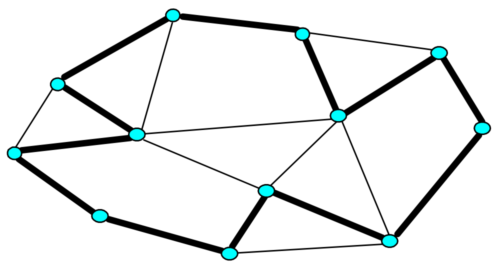
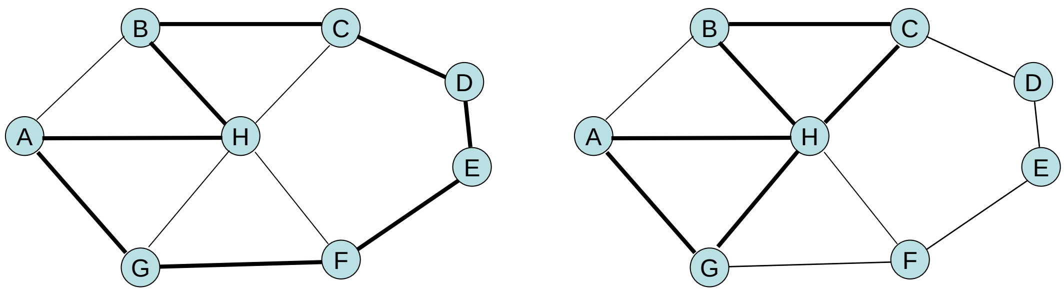


Figure 2.13: *A single Hamiltonian* p-*cycle*

Hamiltonian $p$-cycle is a single large $p$-cycle that travels over every node exactly once as shown figure 8 [61]. That means the circumference is N in a network of N nodes. The Hamiltonian $p$-cycle can achieve lower bounding redundancy by $(1/\bar{d} - 1)$ for any type of span restorable mesh network [62], where $\bar{d}$ is the average nodal degree. Thus, a Hamiltonian $p$-cycle is the most efficient overall solution, theoretically. However, a set of $p$-cycles including small cycles in general provides better solution for reducing the spare capacity. For example, where there exists two spans with unprotected working capacity in the network, a Hamiltonian $p$-cycle wastes other spare capacity to protect two spans. But, these two spans may be protected by a single small cycle. Thus, small cycles are needed to achieve the best performance. Heydari [63] investigates why a Hamiltonian $p$-cycle is far from optimal. It was found that a Hamiltonian $p$-cycle requires over 100% redundancy.

### Non-simple $p$-cycles

Non-simple $p$-cycles have been proposed by Grubern [64]. They are needed in order to achieve good performance when there is not enough capacity available for full protection. Unlike conventional $p$-cycles, non-simple $p$-cycles permit a node to be visited twice. Fig 2.14(b) shows an example of non-simple $p$-cycle protection.




(a) conventional $p$-cycle (b) non-simple $p$-cycle

Figure 2.14: *Example of non-simple cycles*

When link A-B fail, the conventional $p$-cycle is not the best choice for protection. This is because only the conventional $p$-cycle (A-H-B-C-D-E-F-G-A), which should traverse every node, is available. However, the non-simple $p$-cycle is able to find a small cycle (A-H-B-C-H-G-A) to cover the failure so that it reduces the spare capacity. Their simulation results show that non-simple $p$-cycles reduce the redundancy when compared to conventional $p$-cycles. However, the number of candidate $p$-cycles needed for protection is increased, therefore the computation time required is longer.

## 2.3 Conclusion

In this chapter, we have provided an overview of the various approaches that can be used for survivability protection. Specially, $p$-cycle protection can offer an excellent performance in terms of optimality of solution and computational complexity. In $p$-cycle network design, the optimization problem has been formulated as an integer linear program (ILP). Various conventional heuristic approaches have been proposed. However, these heuristic approaches still have an optimization problem in terms of a balancing between optimality of solution and computational complexity. SLA, WDCS and CIDA approaches offer faster computation time but has problem in capacity utilization in the dense and large networks. 'Pre-selection' approach has computational complexity problem but shows good capacity utilization.

# Chapter 3

# Analytical Framework

This chapter defines the various notations used in this thesis and provides an overview of the basic concepts of graph theory to aid the work that will follow. The rest of chapter 3 introduces the networks employed for evaluating our approach and describes the assumption used in the implementation of the work. Typically, approaches for $p$-cycle design first generate the set of all cycles from the network graph. An optimal $p$-cycle is then computed by choosing the number of cycles to be configured as a $p$-cycle. To find efficiency cycles, we need to understand graph theory since a graph may represent certain aspects of a network, usually its topology [13].

## 3.1 Graph Theory

The general solution for network design can be implemented using graph theory algorithms. These algorithms are based on reference to actual networks. In general, graphs are divided into two types depending on the direction of transmission capacity on spans: directed and undirected graphs. With these graphs, a given graph is represented by $G(V, E)$, where $V = \{v_1, v_2, ..., v_N\}$ is the set of $N$ vertices and $E = \{e_1, e_2, ..., e_M\}$ is the set of $M$ spans that make connections between vertices $v_i$ and $v_j$. The difference between the two graph types is only the direction of the spans. For example, if a path exists from $v_i$ to $v_j$, the direction of the path is indicated by an arrow in directed graphs while there exists automatically a path from $v_j$ to $v_i$ in an undirected graph [65]. An edge is denoted by a set $(v_i, v_j)$, where $v_i, v_j \in V$ and $v_i \neq v_j$. Two edges are said to be *adjacent* or *neighboring* if they connect to the same vertex. Similarly, two vertices are said to be *adjacent* or *neighboring* if they are on the same edge. Furthermore, the degree of a vertex is determined by the number of distinct edges that are incident to it [66]. The out-degree and in-degree of a vertex are the number of edges that come out or in from the vertex, respectively. The corresponding network average over all vertices is termed *nodal degree*, $(\bar{\mathrm{d}})$.

For the purpose of describing the work of this thesis, we shall define the following notations:

| | |
|---|---|
| $C$ | The set of channels (capacity) over a span |
| $\varpi$ | The set of unprotected capacity in the network |
| $\mathbb{A}$ | The set of candidate $p$-cycles in the network |
| $\Phi$ | The potential protection capacity |
| $\Phi_i$ | The potential protection capacity of cycle $i$ |
| $\Delta_i$ | The distribution of unprotected capacity after removing capacity protected by cycle $i$ |
| $\Lambda$ | An average contribution of unprotected capacity |
| $\Lambda_i$ | An average contribution of unprotected capacity of cycle $i$ |
| $\Xi$ | The standard deviation of the average unprotected capacity |
| $\Psi$ | Wasted spare capacity |
| $\Psi_i$ | Wasted spare capacity of cycle $i$ |
| $M$ | Maximum capacity of $\varpi$ |
| $\mathbb{L}$ | A span with maximum unprotected capacity |
| $cycle^s$ | the $s_{th}$ candidate $p$-cycle in a set $(\mathbb{A})$ |
| $S_{p,i}$ | The distribution of cycle $p$ on span $i$ |
| $W_i$ | The number of working channels on span $i$ |
| $\mathbb{C}_i$ | The number of spare channels on span $i$ |

### 3.1.1 Survivability based on graph algorithm

In graph theory algorithms, much research has been done to tackle the single shortest path problem for finding a path with a minimum cost from a source to a destination through a connected network [67]. This issue is important because of its wide range of applications in transportation. Furthermore, the shortest paths are a possible way to provide alternative solutions for network design. ILP model generally computes a given traffic demand on each span based on the shortest path algorithm to gain an optimal solution [68]. There are several algorithms to determine a shortest path between a source node and the destination node [66]: e.g. Dijkstra, Breadth First Search (BFS) and Bellman-Ford. In some cases, it is also necessary to know the k shortest paths between a pair nodes. This is because the $k$ shortest paths are required for specific purposes in terms of routing and improving blocking performance.

- Dijkstra algorithm

  The Dijkstra algorithm finds the shortest paths from a source to a destination by weighted in terms of distance. All weights must be positive. The source node must be a single, but the destination may be all other nodes. If the shortest path to node A is desired, this algorithm is terminated when the shortest path to node A has been found. The time required by Dijkstra algorithm is $O(N^2)$ [13], where $O$ is the order of complexity of the algorithm. The basic idea of this algorithm is first to find the closest node from a initial node. The first closest node must be a neighbor of the initial node. The next node must then be a neighbor of the first closest node. The algorithm keeps searching until the destination node is found. The pseudocode of the Dijkstra algorithm is presented below.

- Breath First Search (BFS) algorithm

  The Breath First Search (BFS) algorithm is one of simplest algorithms to search for the shortest path in a given graph. BFS explores all adjacent nodes from the source node and stores the nodes as they are found in a queue. It then searches deeper in one of the adjacent nodes. BFS continues moving on through the next adjacent nodes

**Algorithm 3 : Dijkstra algorithm**

**Input:** $G(V, E)$
**Output:** Shortest path found
$S \leftarrow$ source node
$D \leftarrow$ destination node
$path \leftarrow S$
**while** last node of *path* is not equal to $D$ **do**
  scan all neighbors from last node of *path*
  determine a node with smallest weight or cost
  place selected node into last position of *path*
**end while**

until the shortest paths are found. In general, BFS computes the shortest path using a spanning tree, and different shapes of spanning tree are generated depending on the order in which adjacent nodes are placed into the queue. Fig 3.1 presents the spanning tree of BFS in a given graph.

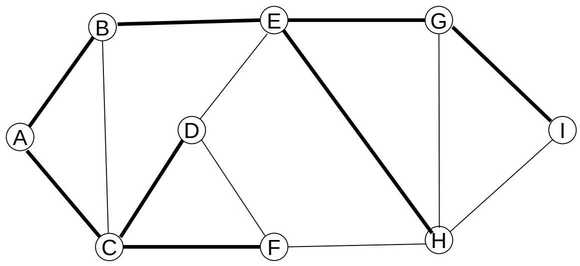



Figure 3.1: *Breadth-First Searching spanning tree of the example graph*

In addition, BFS can be used to achieve four different objectives:

- Testing for connectivity,
- searching a spanning tree,
- searching for a shortest path,
- searching all distinct cycles.

In this thesis, we employ the Breadth First Search (BFS) algorithm to evaluate one of our proposed algorithms. The objective is to determine all distinct cycles. For this objective, the pseudocode of BFS is presented as Algorithm 4.

**Algorithm 4 : Breadth First Search (BFS) algorithm**

**Input:** $G(V, E)$, $s$ source node
**Output:** all distinct cycles ($P$)
$e \leftarrow$ set of neighbor nodes from $s$
$queue \leftarrow$ [s, node.e]
**while** $queue \neq \emptyset$ **do**
  $path \leftarrow head(queue)$
  $queue \leftarrow queue - path$
  $d \leftarrow$ last node of $path$
  **if** $d == s$ **then**
    $P \leftarrow path$
  **else**
    $nei \leftarrow$ set of neighbors of $d$
    $queue \leftarrow$ [queue, node.nei]
  **end if**
**end while**

- Bellman Ford algorithm

  The Bellman Ford algorithm is similar to the Dijkstra algorithm, which is for solving the shortest path problem, but the difference is that this algorithm can have both positive and negative weights. This algorithm searches the shortest path through a process of path extension with cost or weight, like the Dijkstra algorithm. Bellman's computational complexity is $O(n)$ when the nodal degree or the network is small, but worst case is $O(n^3)$. If all edges have equal value of one, the BFS algorithm is a more efficient alternative. If we search the shortest paths for positive weights, however Dijkstra's algorithm is a more efficient alternative than the Bellman Ford algorithm to find the shortest paths.

- k-shortest path algorithm

  The $k$-shortest path algorithm produces the $k$ shortest paths from a given source to the destination node. This algorithm finds the next shortest path such that it does not use any edge or span of the previous path after the first shortest path is found.

The number of shortest paths is controlled by an input parameter $k$. This algorithm is terminated when the cost of a second shortest path is larger than that of its previous paths. The steps are summarized in Algorithm 5

**Algorithm 5 : k-shortest path algorithm**

**Input:** $G(V, E)$, the number of shortest paths ($k$)
**Output:** the $k$ shortest paths found
1. $s$ and $d$ are the source and destination nodes, respectively.
2. Find shortest tree from source node $s$ to other nodes in the network.
3. Define the shortest path $P_1 = (s, s_1, s_2, ..., s_h, d)$ as the first shortest path.
4. Remove links $(s_1, s_2, ..., s_h)$ on the shortest path $P_1$ from a given network.
5. Find second shortest path from source node to destination node if there exists an other shortest path
6. Keep searching until the number of shortest path is equal to $k$ or no further shortest path is found.

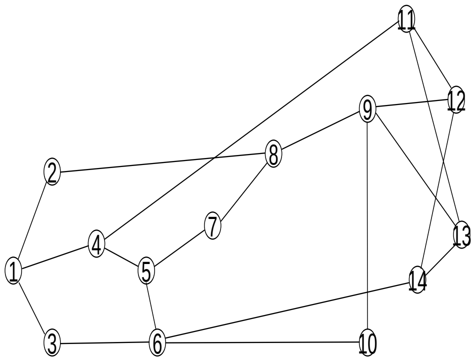
The $k$-shortest path algorithm can be classified into three particular classes [13]:

- k-shortest link disjoint paths: the set of shortest path, in which each path can share spans but is link disjoint.
- k-shortest distinct routes: the set of routes over the spans, in which each of the routes are disjoint from the previous routes.
- k-shortest span-disjoint routes: the set of routes over the spans, in which each of the routes are disjoint from the spans on the previous routes.

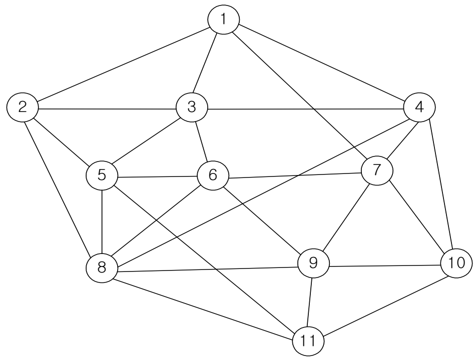
However, these shortest path algorithms sometimes require large spare capacity to protect all of the given traffic in the network. This is because many shortest paths may need to be utilized to provide 100% protection upon a network failure.

### 3.1.2 The test networks

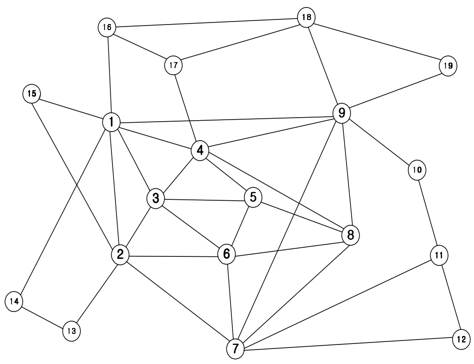
Our proposed algorithm has been implemented and evaluated on four different test networks. These test networks, namely the National Science Foundation Network (NSFNet),



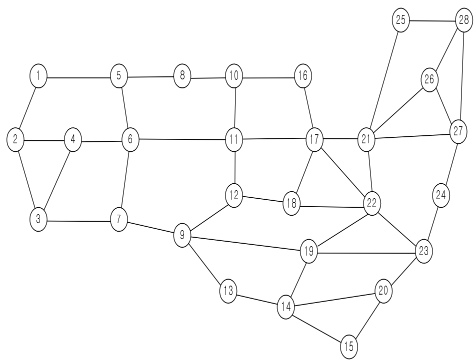

(a) NSFNet Network (N = 14, M = 21)




(b) COST239 network (N = 11, M = 26)




(c) EON network (N = 19, M = 38)




(d) USA network (N = 28, M = 45)

Figure 3.2: The four test networks used in this research

COST239, European Optical Network (EON) and USA networks, are employed as shown in Fig. 3.2. Much research in this area employs these network topologies for implementation and testing of algorithms. This is because these network configurations are based on reference to real national backbone networks. Hence, we also use these network topologies for our study. In addition, NSFNet and COST239 are existing networks of medium size and the other two networks are large size networks so that we can evaluate our algorithm across various network size.

The first study network is the (NSFNet), which is reproduced in Fig 3.2(a), with 14 nodes and 21 spans. The network has a nodal degree of $\bar{\mathrm{d}} \approx 3$. The initial NSFNet consisted of the US network backbone capable of providing 56kbps. This network was not able to support multimedia services.

The COST239 network (Fig 3.2(b)) contains 11 nodes and 26 spans, and has a high nodal degree of $\bar{\mathrm{d}} \approx 4.7$. In addition, the nodal degree of each node is equal to or larger than 4 so that redundancies as low as 30 % are achieved. This network is a medium size network as mentioned, but is also a very dense network.

The European Optical Network (EON) topology has 19 nodes and 38 spans as shown in Fig 3.2(c). The nodal degree is 4 and the topology comprises 1380 unidirectional connection requests [69].

The USA topology (fig 3.2(d)) contains 28 nodes and 45 spans, and has an nodal degree of $\bar{\mathrm{d}} \approx 3.21$. This network is based on reference to the US IP backbone network. Finally, there exists a total of 139, 3531, 8857 and 7321 cycles in each network, respectively. Table 3.1 summarizes these test networks information.

The complexity of non-joint ILP models in dense networks is much higher than the complexity of it in small networks as discussed. This is because the number of variables is high in dense networks. The complexity of the ILP model is $2^{139}$, $2^{3531}$, $2^{8857}$, $2^{7321}$ in each network, respectively [53]. In short, dense networks, which have high complexity, are more

Table 3.1: Four test networks information

| **Network** | NSFNet | COST239 | EON | USA |
|---|---|---|---|---|
| No. of nodes | 14 | 11 | 19 | 28 |
| No. of spans | 21 | 26 | 38 | 45 |
| Average nodal degree | 3 | 4.72 | 4 | 3.21 |
| All possible cycles | 139 | 3531 | 8857 | 7321 |
| Average AE | 1.42 | 2.81 | 2.57 | 1.70 |
| cycles by SLA | 17 | 26 | 27 | 31 |

complicated to gain optimal solution.

## 3.2 Assumptions

In general, traffic connection are offered according to two different types of traffic demands in networks: static traffic and dynamic traffic. With static traffic, the network traffic is assigned in advance and the network should satisfy all traffic requests as well as minimize the total network cost. ILP model is used to solve optimization problem. With dynamic traffic, each connection in the network requests traffic randomly and this is seen as an opportunity to improve blocking probability by dynamically reacting to dynamically varying traffic. Dynamic traffic must be considered because it is important in reality. In our thesis, however, we only consider the static situation and leave the dynamic situation for future work since the static situation is basically first research area to jump into next research area in terms of dynamic situation. Each node is allowed to contain the same amount of network switching resources and network transmission resources. Furthermore, all WDM spans have the same weight, which is set to one, and each fiber span can offer up to $C = 32$ wavelength channels. Routing and Wavelength Assignment (RWA) of the lightpaths is an important issue as mentioned before. The channels composing the lightpath are classified into two cases depending on the availability of wavelength conversion [69]: Virtual Wavelength Path (VWP) and Wavelength Path (WP). VWP is able to convert an incoming wavelength to any outgoing wavelength since wavelength converters are available at every node while a WP is not able to convert any wavelength. As discussed, we do not consider the RWA problem so we assume full wavelength conversion is available in all WDM networks. We defined a span usage (%), which is the ratio of the working capacity in each span to the maximum capacity provided in that span, is randomly generated to be between 25% $\sim$ 40%. Due to the low probability of multiple failures in the networks, all our simulations are based on single link failures of the network. Finally, our computing platform is a 3.0GHz P4 with 1GB of RAM running windows XP and the software platform is Matlab. All simulation running time is limited to 6 hours. That is, the result achieved by ILP is a near optimal solution when computation time is 21000(s). In this thesis, we employ ILP model for the non-joint version.

## 3.3 Conclusion

This chapter has described the basic definitions of graph theory and then introduced some survivability algorithms based on graph theory for finding shortest paths. These algorithms are good way to solve optimization problem like optimality of solution since shortest path can reduce spare capacity required to protect against network failures in the context of survivability. Shortest path algorithms have been used in conventional heuristic approaches, such as SLA and CIDA, for generating a set of candidate $p$-cycles. In our work, we also use one of shortest path algorithms, BFS. Some notations to express our proposed algorithms have been shown and the four real networks employed to evaluate our approach have been detailed. The last part of the chapter has introduced the assumptions used for the $p$-cycle network design in this thesis.

# Chapter 4

# An Approach to Generating an Efficient Set of Candidate $p$-Cycles

In this chapter, we propose a new framework for computing an efficient set of $p$-cycles, which can lead to superior performance in terms of capacity utilization in network protection. We first propose an algorithm for finding all fundamental cycles. Then, the set of candidate $p$-cycles is constructed by merging the fundamental cycles. The number of proposed candidate $p$-cycles is small, but they can construct more straddling links and have high *a priori efficiency*. Finally, an ILP model is employed to select an adequate set of $p$-cycles to ensure 100% protection while minimizing the total spare capacity.

## 4.1 Preliminary Theory

The set of fundamental cycles in the literature is defined as the set of simple cycles obtained from the spanning tree [70, 71, 72]. It is observed that these cycles may contain straddling links. In addition, the set of fundamental cycles as defined in the literature cannot construct all possible cycles that may be solutions in $p$-cycle design. As mentioned above, ILP solution requires enumeration of all possible cycles. In this chapter, we define a new set of fundamental cycles, which can construct all possible cycles. The definitions for our proposed set of fundamental cycles and background theories are presented here.

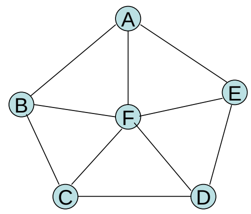



Figure 4.1: 6 nodes, 10 spans topology

- **Definition 1:** A straddling link of a cycle is a link which does not belong to that cycle but whose end-nodes lie on the cycle.
- **Definition 2:** An on-cycle link of a cycle is a link belonging to that cycle.
- **Definition 3:** A fundamental cycle is a cycle which does not contain any straddling links.

An example is shown in Fig. 4.1, where a set of fundamental cycles as per the definition in the literature [72] is $FC = \{c_1, \ldots, c_5\}$, where $c_1 = (A-B-F-A)$, $c_2 = (A-F-E-A)$, $c_3 = (C-F-D-C)$, $c_4 = (A-B-C-F-A)$ and $c_5 = (A-F-D-E-A)$. It is easy to see that cycles $c_4$ and $c_5$ contain links B-F and E-F as straddling links, respectively. In addition, $FC$ cannot construct a cycle $c_6 = (A-B-C-D-E-A)$ and two small cycles ($c_7 = (B-C-F-B)$, $c_8 = (D-E-F-D)$), hence this set may not be able to construct all possible cycles in the network. However, our set of fundamental cycles contains $c_6$, $c_7$, $c_8$ because these do not include any straddling links. In fact, our set contains all possible fundamental cycles that contain no straddling links. Thus, our set of fundamental cycles can construct all possible cycles of the network by joining these fundamental cycles.

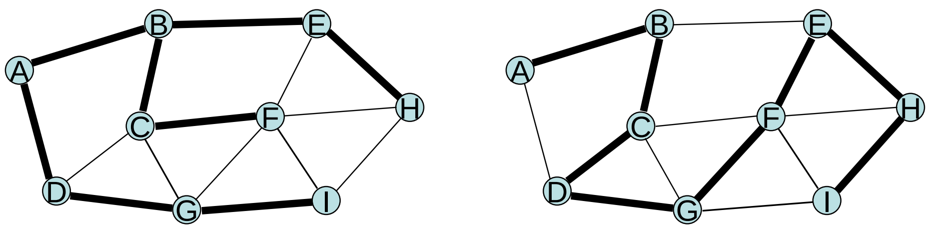
## 4.2 Generation of Candidate p-Cycles

Our method for generating candidate $p$-cycles is a two-stage process. The first stage generates a set of fundamental cycles. The second stage determines a set of efficient candidate

$p$-cycles from these. After the set of candidate $p$-cycles is generated, an ILP formulation for minimizing spare capacity is applied to select an optimal set of $p$-cycles that will ensure 100% protection of working capacity. The ILP model was introduced in chapter 2. To generate fundamental cycles, we propose a new method called the Breadth-Depth First Search (BDFS).

### 4.2.1 Breadth-Depth First Search (BDFS) algorithm

In general, there are two possible approaches for generating candidate cycles: the Breadth First Search (BFS) and the Depth First Search (DFS). Basically, these approaches generate all cycles based on spanning trees. The BFS checks each possible path from each node to determine if it forms a cycle, exploring the shortest cycles first. On the contrary, DFS explores each possible path from the first node. If the path contains no destination node, another path is explored by backtracking until a cycle is found. Fig 4.2 presents the spanning tree of these approaches.




(a) Spanning tree of BFS (b) Spanning tree of DFS

Figure 4.2: Spanning tree of BFS and DFS

In our proposed method, we explore first all possible fundamental cycles traveling over an initial node A. We then remove the node A from the given network to avoid enumeration of repeated cycles. Every node is then progressively explored until no further nodes remain in the network. The pseudocode for generating fundamental cycles using our method is shown as Algorithm 6.

**Algorithm 6 : Finding fundamental cycles**

**Input:** An undirected graph $G(V, E)$
**Output:** Set of all fundamental cycles
$i \leftarrow 1$
$NoN = |V|$
**while** $NoN \geq 3$ **do**
  Find all fundamental cycles passing through a node $i$ using BDFS (Algorithm 7)
  Remove the node $i$ from $G(V, E)$
  i = i+1
  NoN = NoN - 1
**end while**

In Algorithm 6, all fundamental cycles traveling over each node are explored by BDFS. The following two conditions are used to form our fundamental cycles. First, BDFS needs at least two nodes neighboring the root node to form a cycle. This is because a cycle cannot be formed if there is only a single neighboring node. Secondly, the algorithm checks for all expanded nodes to find a node connected to the prior node. This is because this node should be on a straddling link of the cycle. The BDFS algorithm removes these nodes from the expanded nodes. The pseudocode of the BDFS algorithm is shown as Algorithm 7.

### 4.2.2 Generating the set of candidate *p*-cycles

In our method, an efficient set of $p$-cycles is computed by merging fundamental cycles. Existing approaches for merging cycles can be classified into two groups, the 'Add' algorithm and the 'Join' algorithm [55]. Both of these algorithms create $p$-cycles with more straddling links so that the $p$-cycles are more efficient. The 'Add' algorithm is only possible when a straddling link on cycle A is a cycle link on cycle B and a straddling link on cycle B is a cycle link on cycle A [55]. However, our fundamental cycles have no straddling links on the cycles as mentioned. Thus, the 'Add' algorithm is not suitable for generating $p$-cycles in our method. On the other hand, the 'Join' algorithm is used when pairs of fundamental

**Algorithm 7 : Breadth-Depth First Search (BDFS)**

```
Input: An undirected graph G(V, E), root
Output: Candidate fundamental cycles
Nei ← set of neighbors of node root
if |Nei| ≤ 2 then
  return
end if
for i = 1 to |Nei| − 1 do
  for j = i + 1 to |Nei| do
    s ← Nei(i)
    d ← Nei(j)
    if d is a neighbor of s then
      Create fundamental cycle
    else
      pathlist = [s]
      while Pathlist is not empty do
        currentpath ← pathlist(1)
        s ← last node of currentpath
        remove s from pathlist
        if d is a neighbor of s then
          Create fundamental cycle
        else
          for each k neighbor node of s do
            if d is neighbor node of k then
              Create fundamental cycle
            else
              Newpath = [currentpath, k]
              Put newpath in the pathlist
            end if
          end for
        end if
      end while
    end if
  end for
end for
```

cycles have one common link and the common link becomes a straddling link for the $p$-cycle. However, two adjacent cycles with more than one link in common cannot form a new $p$-cycle. This is because the two cycles cannot build a straddling link when the common links of these cycles are adjacent.

In our method the 'Join' algorithm is employed for generating a set of $p$-cycles. At first, the method chooses one fundamental cycle A from our set of fundamental cycles. It then establishes a list of all other cycles that have one common link with cycle A. The smallest such cycle is chosen to join cycle A and form a new $p$-cycle. If such a cycle does not exist, we then proceed to find another fundamental cycle within our set of fundamental cycles. This procedure is repeated until no further $p$-cycles are found. The pseudocode of the merge algorithm is given as Algorithm 8.

**Algorithm 8 : Operation of merge algorithm**

**Input:** Set of fundamental cycles
**Output:** Candidate $p$-cycles
$listpcycle \leftarrow \emptyset$
$Fc \leftarrow$ set of fundamental cycles
**for** $i = 1$ to $|Fc|$ **do**
  $pcycle \leftarrow Fc(i)$
  **while** $p$cycle is not empty **do**
    Find fundamental cycles which have one common link with $p$cycle from Fc
    **if** Exist **then**
      Join $p$cycle with smallest cycle to create new $p$cycle
      Add new $p$cycle to list$p$cycle
      $pcycle \leftarrow$ new $p$cycle
    **else**
      $pcycle \leftarrow \phi$
    **end if**
  **end while**
**end for**
Candidate $p$-cycles $\leftarrow listpcycle \cup Fc$

## 4.3 Results and Analysis

In this section we examine the efficiency of our proposed algorithm in terms of the optimality of the solution and the computational time. We also compare how different the performance of our proposed algorithm is from the optimal solution by $Diff(\%)$ [55].First of all, we examine the set of our defined fundamental cycles. The number of fundamental cycles and the computational time of our proposed algorithm are verified in different network topologies. Next, we verify the optimality of the solution and the complexity of our approach. For medium and large network, two test case networks, namely the COST239 and USA networks, are employed as shown in Fig. 3.2. The COST239 network is also a dense network, as discussed previously. Since the aim of our approach is only to minimize the total spare capacity used, the working capacity in each span is randomly generated with around 24% $\sim$ 36% of the maximum capacity provided in that span.

### 4.3.1 Generating the set of fundamental cycles

The first task is to generate all possible fundamental cycles for the two test networks. To validate the reliability of our set of fundamental cycles, we compare them with the set of fundamental cycles generated by BFS. BFS first constructs a set of all possible cycles in the network and then we select our new fundamental cycles from the set. In the COST239 network, there are a total of 3531 cycles and all of the 42 fundamental cycles are generated by BFS. The USA network has 47 fundamental cycles out of a total of 7321 cycles. Our algorithm finds the same number of fundamental cycles as BFS in both of these networks. Table 4.1 presents our results and network information for the test networks. Note that our set of fundamental cycles is different from other works as explained in section 4.1. It is found that our simulation results are in excellent agreement with those generated by BFS. Since we have verified that the BDFS algorithm is reliable, we can construct any possible set of cycles for the test networks based on our set of fundamental cycles. Also, the computation time for the BDFS algorithm was found to be 0.07(s) and 0.5(s) for the COST239 and USA network topology, respectively, whereas BFS requires 1800(s) and 10800(s) of computation time.

Table 4.1: All possible fundamental cycles in COST239 and USA

| **Network** | COST239 | USA |
|---|---|---|
| All possible cycles | 3531 | 7321 |
| All fundamental cycles by BDFS | 42 | 46 |
| Fundamental cycles by BFS | 42 | 46 |
| Computation time(s) for BDFS | 0.07 | 0.5 |
| Computation time(s) for BFS | 1800 | 10800 |

### 4.3.2 Optimality of solution and computational complexity of our proposed approach

In the next simulation, we verify the optimality of the solutions and the computational complexity of our proposed approach. We compare the results of our solutions to the optimal ILP model using all cycles as the set of candidates. The complexity is compared through the number of candidate $p$-cycles put into the ILP model.

We first compute the set of efficient candidate $p$-cycles using the merge algorithm. The ILP model is then used to achieve the best performance. Table 4.2 shows the performance of our algorithm for the COST239 network. The range of total traffic demand is set to be from 200 to 300. To evaluate the validity of our algorithm, we perform ILP with full cycle enumeration with the same traffic demand on the test network.

It is found from these simulation that the results achieved are near optimal as shown in Table 4.2. The ILP uses 3531 $p$-cycles to converge to the optimal solution. In the best case (traffic demand of 220), the redundancy from our algorithm is 33.2%, which is the same result as the ILP solution. In the worst case (traffic demand of 300), the optimal result

Table 4.2: Simulation results for COST239 topology

| | | **COST239** (C = 32, NoP:3531) | | | | | |
|---|---|---|---|---|---|---|---|
| | **WC** | 200 | 220 | 240 | 260 | 280 | 300 |
| all cycles | Re | 36.5 | 33.2 | 32.1 | 30.8 | 30 | 30.3 |
| | time(s) | 915 | 19378 | 17134 | 16120 | 21600 | 7340 |
| our candidates | Re | 37 | 33.2 | 32.1 | 32.3 | 32.5 | 33.7 |
| | time(s) | 2 | 368 | 21 | 41 | 104 | 71 |
| | NoP | 253 | 253 | 253 | 253 | 253 | 253 |
| usage(%) | | 24 | 26.4 | 28.9 | 31.3 | 33.7 | 36 |

WC: working capacity, Re: Redundancy, NoP: no. of $p$-cycle

is 29.5 whereas ours is 33.7. Thus, given the same traffic demands, we are able to achieve solutions within 4% of the optimal solutions from a set of only 253 $p$-cycles. This number of candidate $p$-cycles is only 7% of all possible cycles. Also, the computation time required is less than 2% of that required to determine the fully optimal solutions for each traffic demand. Note that we define the computation time to be the ILP runtime.

Table 4.3: Simulation results for USA topology

| | | **USA** (C = 32 , NoP:7321) | | | | | |
|---|---|---|---|---|---|---|---|
| | **WC** | 410 | 430 | 450 | 470 | 490 | 510 |
| all cycles | Re | 69.5 | 64.1 | 68.7 | 69.6 | 68.8 | 68.6 |
| | time(s) | 1699 | 7752 | 8386 | 1068 | 868 | 4686 |
| our candidates | Re | 70 | 64.9 | 70.4 | 71.7 | 70.6 | 71 |
| | time(s) | 27 | 2 | 3 | 30 | 47 | 78 |
| | NoP | 302 | 302 | 302 | 302 | 302 | 302 |
| usage(%) | | 27.9 | 29.2 | 30.6 | 31.9 | 33.2 | 34.6 |

WC: working capacity, Re: Redundancy, NoP: no. of $p$-cycle

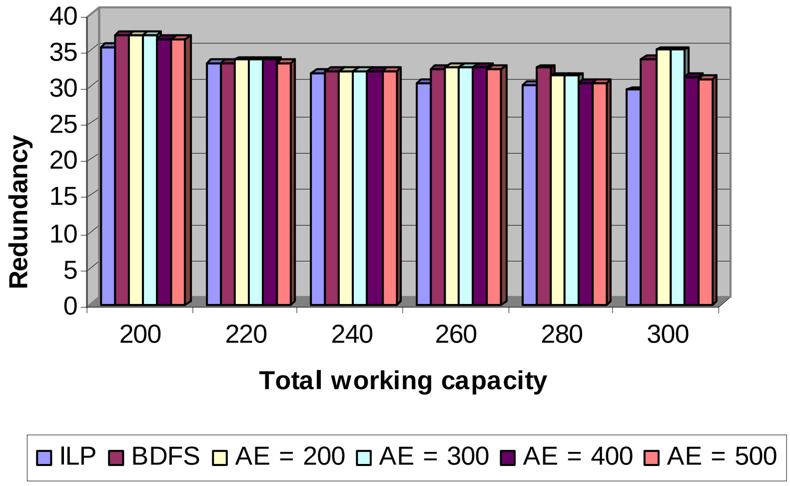
Table 4.3 presents the corresponding results for the USA network topology. The total traffic demand is set to be between 410 and 510. These results also indicate near optimal solutions. The redundancy in our results is only slightly higher than the optimal solutions. The simulation shows that the results of our algorithm in only 0.5% off the optimal solution in the best case. The total number of $p$-cycles found by our algorithm is less than 4% of all possible cycles in this case. In addition, the computation time required is less than 1% of that taken by ILP if all cycles are considered.

To evaluate our simulation results, we compare them to the AE and WDCS approaches. With the AE, the number of candidate $p$-cycles is determined from high values of AE. The number of $p$-cycles is set to be from 200 to 500. Fig 4.3(a) presents the simulation result for COST239. It is observed that the performance achieved by AE is close to the full ILP solution and our approach. Interestingly, every result is similar when we increase the number of cycles. That is because candidate $p$-cycles does not contain small cycles. Small cycles are, however, sometimes necessary. For example, if a span with unprotected capacity is left after we protect working capacity using a high efficiency cycle, the high efficiency cycle wastes network capacity and this case may worsen our results

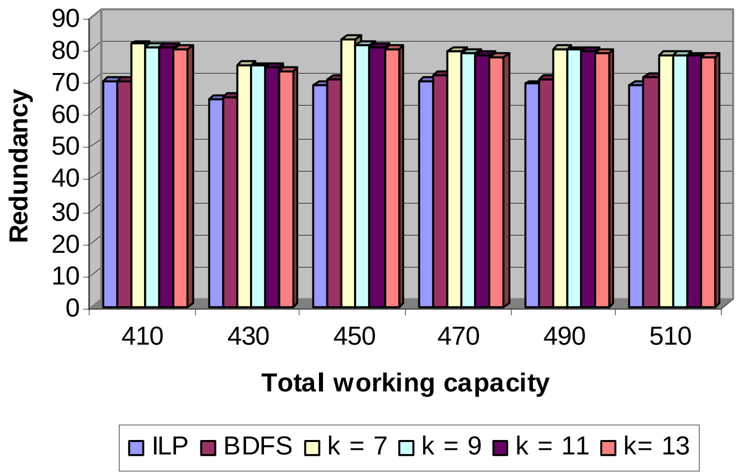
For the USA network, the performance of AE is close to the ILP solution as shown in Fig 4.3(b). However, we are able to observe that our set of $p$-cycles can improve the performance more than the set of $p$-cycles derived using AE. When the number of cycles is 800 (not shown in Fig. 4.3(b)), the AE approach is able to achieve performance within 3% of the optimal for all working capacities.

With the WDCS approach, the number of $p$-cycles is increased dependant on the parameter $k$ as mentioned before. Fig. 4.4 shows the change in the number of candidate $p$-cycles as $k$ is increased. The simulation results showing the number of cycles as a function of $k$ is presented in Fig 4.5. For the COST239 topology, our simulation results are better than WDCS with the same number of candidate $p$-cycles. In the best case, our result is 1.9% from the optimal solution whereas WDCS is 5%. However, WDCS achieves slightly better results when increasing $k = 10$ in traffic demand of 280. This is expected because increasing $k$ constructs more candidate $p$-cycles.

For the USA topology, our candidate $p$-cycles can achieve performance within 3% of the ILP solution, while WDCS only achieves results within 12% of the optimal. In the

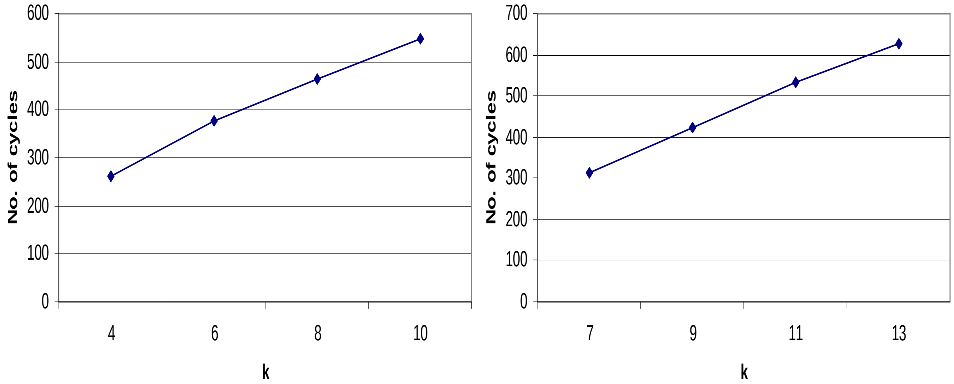



(a) COST239 network




(b) USA network

Figure 4.3: Simulation comparing AE and our BDFS algorithm for two test networks




(a) COST239 network

(b) USA network

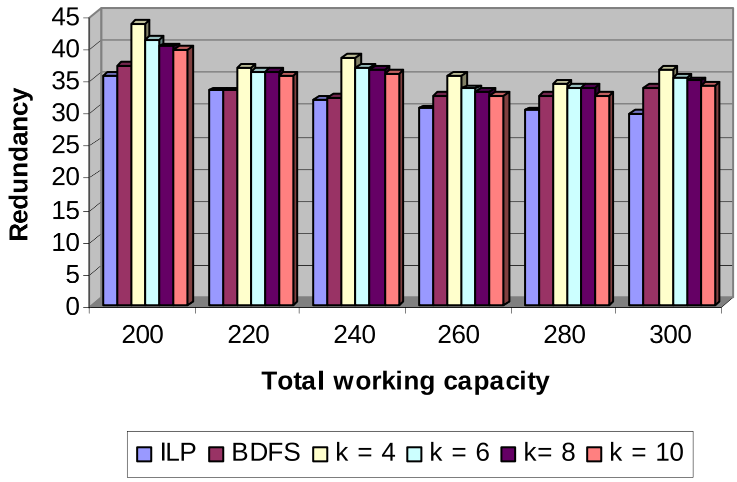
Figure 4.4: Number of candidate $p$-cycles achieved by WDCS

worst case, our results are within 2.4% from the optimal whereas WDCS is within 14% as shown in Fig 4.6(b). Fig 4.6 shows Diff(%) results, when AE and $k$ are 300 cycles and 8, respectively. These results clearly show that our proposed approach is able to achieve a better solution for the USA network and, therefore, is effective for dense networks.

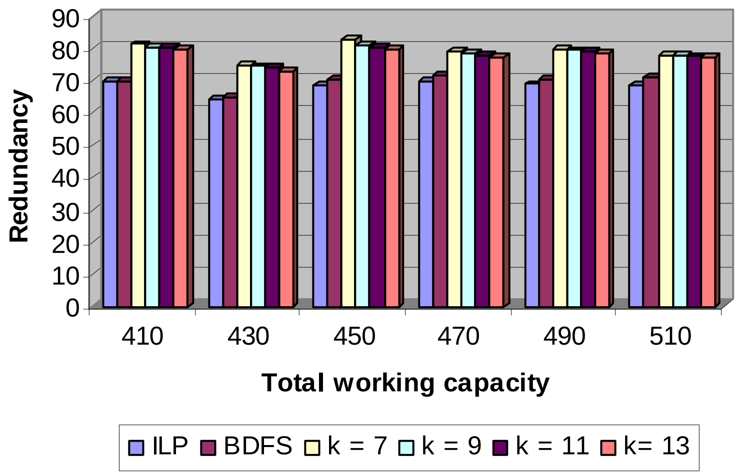
In the candidate $p$-cycles generated by our proposed BDFS algorithm, the maximum AE is 3.73 and the average AE is 2.39 for COST239. For the USA network, the maximum/average AE values are 2.21/1.59, respectively. Furthermore, the algorithm produces cycles with a minimum AEs of 1, since fundamental cycles also belong to the candidate set of $p$-cycles.

## 4.4 Conclusion

We have developed an effective algorithm to compute a sufficient set of $p$-cycles to ensure 100% working capacity protection with a minimum spare capacity requirement using the ILP formulation. The algorithm consists of two stages. The first stage generates all possible fundamental cycles, and the second stage computes sufficient candidate $p$-cycles from

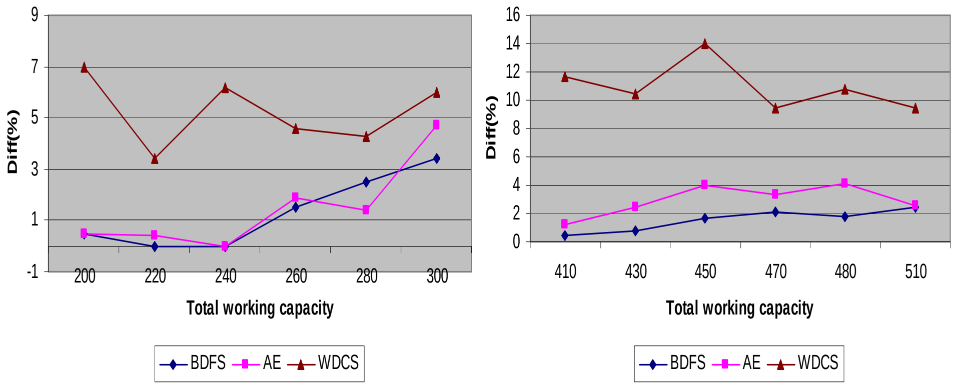



(a) COST239 network




(b) USA network

Figure 4.5: Simulations comparing WDCS and ours for two test networks




(a) COST239 network (b) USA network

Figure 4.6: The performance in terms of Diff(%) for two test networks

these. Our generated fundamental cycles can be used to construct all possible cycles of the networks.

For the two test networks simulated, it has been found that our method can achieve near optimal solutions using ILP. The number of candidate $p$-cycles generated is less than 7% of all possible cycles for the test networks and the solutions are close to optimal. That is, our method is able to achieve satisfactory results with only a small number of candidate $p$-cycles. The method in this paper is, however, still based on the ILP formulation with its associated computational complexity. In the next chapter, we shall discuss a pure heuristic approach for $p$-cycle network design to avoid the computational complexity of the ILP models.

# Chapter 5

# A Pure Heuristic Approach for $p$-Cycle Network Design

## 5.1 Proposed Heuristic Approach

In this chapter, we propose a new pure heuristic approach for designing survivable WDM networks based on $p$-cycles. The objective of our algorithm is to ensure 100% protection while minimizing the total spare capacity and reducing the computation time. To achieve our objective, we first construct all possible cycles using a proposed Pre-compute all Cycles Approach (PCA). An adequate set of candidate $p$-cycles is then selected using a matching criteria between the distribution of the working channels and the distribution of the protected channels on the selected set of cycles.

### 5.1.1 Heuristic approach for Efficient Cycle Selection (HECS)

In the proposed algorithm, all cycles are pre-computed. This is because if we construct all cycles for large dense networks, the computation time will be very long. This would mean that the heuristic approach would have the same computational complexity problem as the ILP model. Thus, we pre-compute all possible cycles according to PCA before we run our algorithm. Then, to determine a suitable cycle, we calculate the potential-protection capacity ($\Phi$) of each cycle. Each cycle has different $\Phi$ since each cycle has different straddling links and on-cycle links. The potential-protection capacity is defined as:

$$\Phi_p = \sum_{i=1}^{M} cycle_{p,i} \tag{5.1.1}$$

where

$$cycle_{p,i} = \begin{cases} 2 & \text{If } i \text{ is a straddling span of } cycle_p \\ 1 & \text{If } i \text{ is an on-cycle span of } cycle_p \\ 0 & \text{otherwise} \end{cases}$$

In our algorithm, $\Phi$ is also pre-computed. The reason is to avoid computational complexity problem. Suitable cycles are determined using the CSSD approach with three factors, average contribution of unprotected capacity ($\Lambda$), standard deviation of unprotected capacity ($\Xi$) and wasted spare capacity ($\Phi$). The algorithm keeps on searching cycles until 100% protection of the whole working capacity is achieved. The design of this heuristic algorithm is summarized as follows:

- step 1. Pre-compute all cycles in the networks using PCA.
- step 2. For each candidate cycle, pre-calculate $\Phi$ of the cycle usning 5.1.1.
- step 3. Select the best cycle using CSSD.
- step 4. remove the working capacity protected by selected cycle in step 3 and update the working capacity.
- step 5. Go to step 3 until no further working capacity is on each span.

**Pre-compute all Cycles Approach (PCA)**

DFS and BFS are the most basic algorithms to determine all cycles. As mentioned before, these algorithms store the nodes as they are searched in a queue and explore them in that order. However, in our algorithm, PCA first explores all possible cycles traveling over an initial node. Each node is then progressively explored until every node is discovered. The

**Algorithm 9 : Pre-compute all Cycles Approach (PCA)**

**Input:** A undirected graph $G(V, E)$
**Output:** Set of all cycles
$i \leftarrow 1$
$No = |V|$
**while** $No \geq 3$ **do**
  Find all cycles passing through a node $i$ using TEN (Algorithm 10)
  Remove node $i$ from $G(V, E)$
  i = i+1
  No = No - 1
**end while**

objective of PCA is to construct all possible cycles in the given network. The pseudocode of PCA is shown as Algorithm 9.

All cycles passing through a node are explored by a proposed traveling over Each Node (TEN) algorithm. The TEN algorithm is similar to the BDFS algorithm expect that when it encounters the first fundamental cycle where a destination node connects only to a source node, it does not stop after constructing the fundamental cycle. Instead, TEN continues searching cycles traveling over the node. There exists two conditions that must be satisfied to implement PCA. First, we remove nodes with a nodal degree 1 ($\bar{d}$=1) from the network topology since a cycle cannot be formed from these nodes. Secondly, we disconnect every link which connects to a root node after we find all cycles through a node. This is in order to avoid enumeration of repeated cycles. Algorithm 10 presents the pseudocode of the TEN algorithm.

**Cycle Selection based on Standard Deviation (CSSD) Approach**

The main objective of CSSD is to select a suitable set of cycles for protection. Among the existing heuristic approaches, one feasible scheme for determining a set of $p$-cycles is to compute the capacity utilization protected for each candidate cycle. Our algorithm also measures the capacity utilization protected by computing the average unprotected capacity after removing the capacity protected by the selected cycle. This average shows how much

**Algorithm 10 : Traveling over Each Node (TEN)**

**Input:** An undirected graph $G(V, E)$, root
**Output:** Candidate cycles
$Nei \leftarrow$ set of neighbors of node root
**if** $|Nei| < 2$ **then**
  *return*
**end if**
**for** $i = 1$ to $|Nei| - 1$ **do**
  $s \leftarrow Nei(i)$
  pathlist = [root s]
  **while** Pathlist is not empty **do**
    $currentpath \leftarrow pathlist(1)$
    $parentnodes \leftarrow currentpath$
    $N \leftarrow$ last node of currentpath
    remove *currentpath* from pathlist
    $list \leftarrow$ neighbor node of $N$
    remove *parentnodes* from *list*
    **for** each $k$ node of *list* **do**
      **if** $k$ is *root* **then**
        Create cycle
      **else**
        Newpath = $[currentpath, k]$
        Put newpath in the pathlist
      **end if**
    **end for**
  **end while**
  disconnect $s$ from *root*
**end for**

the distribution of unprotected capacity is reduced by the selected cycle. The average ($\Lambda$) is expressed as:

$$\Lambda_j = mean(\Delta_j) \tag{5.1.2}$$

constraint:

$$\Delta_j = \varpi - \Phi_j \tag{5.1.3}$$

If the number of cycles selected by $\Lambda$ is one, we select the cycle. If not, we apply the standard deviation of $\Lambda$ ($\Xi$) in order to select a next best cycle. $\Xi$ gives us more information to select a cycle. For example, if a cycle selected by $\Lambda$ only protects some unprotected capacity, it will sometimes leave high wasted working capacity on some spans. However, if we select more spread cycles for protection, it is possible to avoid wasted working capacity. That is, $\Xi$ shows how spread up each cycle is to cover unprotected capacity. Thus, we use $\Xi$ and select its minimum value from a set of $\Xi$. $\Xi$ is defined as:

$$\Xi = \sum_{s=1}^{M} (\Delta_s - \Lambda_s)^2 \tag{5.1.4}$$

After $\Xi$ and $\Lambda_s$ are known, our last decision criterion is applied if the number of cycles is still not one. This criterion is used to reduce the amount of spare capacity. The wasted spare capacity is working capacity for on-cycle spans, those that do not protect any working capacity but should occupy the spare capacity. If we can determine a cycle with a smallest degree of spare capacity but the same capacity utilization protected, we achieve better performance. Thus, we determine a cycle which has the minimum value from a set of $\Phi$. The wasted spare capacity $\Phi_p$ is computed as:

$$\Phi_p = \frac{\displaystyle\sum_{i=1}^{L} \mathbb{W}_{p,i}}{\displaystyle\sum_{i=1}^{L} \mathrm{C}_{p,i}} \tag{5.1.5}$$

where

$$\mathbb{W}_{p,i} = \begin{cases} 1 & \text{if } \mathbb{S}_{p,i} \text{ is less than zero} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbb{C}_{p,i} = \begin{cases} 1 & \text{if } i \text{ is an on-cycle span of cycle}_p \\ 0 & \text{otherwise} \end{cases}$$

Algorithm 11 shows the pseudocode of CSSD. The CSSD approach only determines one best cycle from all candidate $p$-cycles.

**Algorithm 11 : CSSD**

**Input:** working capacity, $\Phi$
**Output:** *cycle*
Unprotected capacity($\varpi$) $\leftarrow$ working capacity
Compute ($\Delta$, $\Lambda$) of $|\Phi|$ by Eqs. 5.1.2, 5.1.3
Determine set of cycles ($\eta$) with minimum value of $\Lambda$
Compute $\Xi$ of $\eta$ by Eqs. 5.1.4
Check set of cycles($\Theta$) which is minimum value in $\Xi$
Calculate spare capacity($\Psi$) of set $\Theta$ using Eq. 5.1.5
Select a cycle with minimum $\Psi$ of set $\Theta$

## 5.2 Simulation Results and Analysis

In this section, we present and discuss the results obtained by performing HECS. Four well-known networks have been considered: the NSFNet, COST239, EON and USA networks. Data regarding these topologies has been introduced in chapter 3. We compare the performance of the proposed heuristic algorithm to optimal solutions and conventional heuristic approaches to evaluate the effectiveness of the proposed approach.

### 5.2.1 Performance comparison

First of all, we implement the PCA algorithm to compute all cycles in the given networks. Table 5.1 shows these results. They are presented as number of cycles and computation time. It is observed from the results that our algorithm is reliable within a reasonable running time.

Table 5.1: All cycles generated by PCA for test networks

| **Network** | NSFNet | COST239 | EON | USA |
|---|---|---|---|---|
| All possible cycles | 139 | 3531 | 8857 | 7321 |
| All cycles by PCA | 139 | 3531 | 8857 | 7321 |
| time(s) for PCA | 0.68 | 12.42 | 63.81 | 2384 |

After determining candidate cycles, we compare the performances to the ILP formulation, the heuristic algorithm CIDA given in [55] and Zhang's algorithm provided in [57] to evaluate our simulation. From now on, we will refer to Zhang's algorithm as Zhang. Every algorithm in this simulation is performed with all cycles. The computational time and the percentage difference from the optimal solution (*Diff(%)*) are shown for each network. For each network, the simulation results are shown in Tables 5.2, 5.3, 5.4 and 5.5. The average results are described in Table 5.6 and Fig. 5.1 presents the differences from the ILP solutions. Interestingly, the comparison shows that the results from CIDA and Zhang are exactly the same. The only difference between them is the computation time. The running time of Zhang is on average is slightly longer than CIDA as shown in Table 5.6. Thus, we do not discuss Zhang's algorithm in these simulation results.

**Simulation results for NSFNet topology**

The total working capacity is varied from 193 to 287. We observe from the numerical results that the performance of HECS is very close to CIDA in this test network. In the best case, CIDA is 1.1% worse that the optimal solution, whereas HECS is 2.4%. The computation time only requires 0.03(s) for both of these approaches. That is, we observe an outstanding improvement in computational time of our proposed approach compared to the ILP solution. Furthermore, the redundancy on average results shows that HECS and CIDA are 2.87% and 3.68% of the optimal solution within 0.1(s) of computation time.

**Simulation results for COST239 topology**

Table 5.2: Simulation results for NSFNet topology

| | **NSFNet** (C = 32) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | **WC** | 193 | 204 | 212 | 225 | 237 | 243 | 257 | 264 | 271 | 287 |
| pure ILP | Re | 68.4 | 64.2 | 69.8 | 64 | 67 | 60.9 | 68.1 | 66.3 | 65.3 | 65.5 |
| | time(s) | 2 | 9 | 17486 | 12 | 11.8 | 2.1 | 92.9 | 12 | 1 | 19.3 |
| CIDA | Re | 72 | 68.1 | 73.1 | 68.4 | 70.3 | 65 | 72 | 70.1 | 69.7 | 66.6 |
| | time(s) | 0.03 | 0.02 | 0.02 | 0.02 | 0.03 | 0.04 | 0.03 | 0.03 | 0.03 | 0.03 |
| Zhang | Re | 72 | 68.1 | 73.1 | 68.4 | 70.3 | 65 | 72 | 70.1 | 69.7 | 66.6 |
| | time(s) | 0.16 | 0.07 | 0.12 | 0.16 | 0.07 | 0.11 | 0.09 | 0.12 | 0.08 | 0.09 |
| HECS | Re | 69.4 | 69.1 | 72.2 | 70.2 | 70.3 | 63.4 | 70.8 | 70.1 | 68.3 | 67.9 |
| | time(s) | 0.03 | 0.02 | 0.02 | 0.02 | 0.03 | 0.04 | 0.03 | 0.03 | 0.03 | 0.03 |
| usage(%) | | 28.7 | 30.4 | 31.6 | 33.5 | 35.3 | 36.2 | 38 | 39.3 | 40.3 | 42.7 |

WC: working capacity, Re: Redundancy

The number of traffic demand is varied from 240 to 340. It is found from the simulation results that on average the solutions achieved by CIDA are close to the optimal solution with just a 3.99% difference from the ILP solution. In the best case, CIDA is 0.3% different from optimal results and with a computation time of 0.2(s). In the worst case (traffic demand of 295), CIDA is 7.1% off. This result shows that the implementation of CIDA can lead the worst case for protection.

the worst case occurs with the implementation of CIDA.

Table 5.3: Simulation results for COST239 topology

| | **COST239** (C = 32) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | **WC** | 244 | 254 | 266 | 273 | 287 | 295 | 302 | 318 | 322 | 331 |
| pure ILP | Re | 31.6 | 29.1 | 30.1 | 30 | 30.3 | 29.2 | 29.1 | 30.5 | 32.9 | 29.3 |
| | time(s) | 21600 | 17194 | 21600 | 21600 | 2137 | 21600 | 3871 | 21600 | 21600 | 4872 |
| CIDA | Re | 35.7 | 32.3 | 35 | 33.3 | 33.8 | 36.3 | 33.1 | 34.6 | 33.2 | 34.7 |
| | time(s) | 0.24 | 0.2 | 0.2 | 0.22 | 0.2 | 0.2 | 0.22 | 0.3 | 0.2 | 0.22 |
| Zhang | Re | 35.7 | 32.3 | 35 | 33.3 | 33.8 | 36.3 | 33.1 | 34.6 | 33.2 | 34.7 |
| | time(s) | 0.32 | 0.2 | 0.26 | 0.22 | 0.21 | 0.19 | 0.25 | 0.32 | 0.24 | 0.25 |
| HECS | Re | 32.4 | 29.5 | 30.1 | 30.4 | 31.4 | 29.2 | 29.1 | 30.2 | 30.1 | 29.9 |
| | time(s) | 0.3 | 0.24 | 0.3 | 0.25 | 0.3 | 0.29 | 0.34 | 0.35 | 0.31 | 0.34 |
| usage(%) | | 30.3 | 31.5 | 33 | 33.9 | 35.6 | 36.6 | 37.5 | 39.5 | 40 | 41.1 |

WC: working capacity, Re: Redundancy

On the other hand, we have shown that the performance of HECS is much better than that of CIDA. HECS can achieve -2.8% of optimal solution within 0.3(s) when traffic demand of 295 is used. Please note that we restrict the computation time of ILP to a maximum of 21000(s), hence the better than optimal result. In the worst case, HECS is only 1.1% from the optimal solution. Our performance on average is 184 times better than CIDA and the running time of HECS is around $2 * 10^{-5}(s)$ of that required by the optimal solution. In short, our approach can achieve near optimal solution within acceptable computation time.

**Simulation results for EON topology**

We now simulate the EON topology and then compare the performance of this topology to other approaches such of the CIDA and ILP models. The simulation results show that CIDA offers a significantly faster computation time compared to the ILP model. However, we find that the solutions generated using CIDA has a large discrepancy compared to the ILP model.

Table 5.4: Simulation results for EON topology

| | | **EON** (C = 32) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | **WC** | 355 | 367 | 377 | 384 | 392 | 405 | 411 | 427 | 436 | 445 |
| pure ILP | Re | 52.9 | 58.6 | 53.5 | 51 | 53.6 | 54.8 | 53 | 55 | 53.2 | 49.9 |
| | time(s) | 3915 | 21600 | 3828 | 197 | 1644 | 8631 | 21600 | 214 | 8686 | 21600 |
| CIDA | Re | 64.4 | 66.9 | 63.3 | 60.2 | 64 | 64.4 | 65 | 65.8 | 63.5 | 60 |
| | time(s) | 1.8 | 1.9 | 1.8 | 1.7 | 2 | 2 | 2.2 | 2.1 | 2.1 | 2 |
| Zhang | Re | 64.4 | 66.9 | 63.3 | 60.2 | 64 | 64.4 | 65 | 65.8 | 63.5 | 60 |
| | time(s) | 2.2 | 2.34 | 2.14 | 2.21 | 2.73 | 2.67 | 2.8 | 2.91 | 2.89 | 2.6 |
| HECS | Re | 56.3 | 59.2 | 55.1 | 56.3 | 56.1 | 56.3 | 54.7 | 57.4 | 56.4 | 52.8 |
| | time(s) | 1.9 | 1.9 | 1.8 | 2.3 | 2.2 | 2 | 2.3 | 2.5 | 2.6 | 2.2 |
| usage(%) | | 30.1 | 31.2 | 32 | 32.6 | 33.3 | 34.4 | 34.9 | 36.3 | 37 | 37.8 |

WC: working capacity, Re: Redundancy

For example, in the worst case CIDA is 11.5% from the optimal solution and the best found solutions of CIDA are 8.3% away. If we focus on running time, however, CIDA can make the ILP model solvable with a much lower computational complexity (4595 times faster than ILP). With the HECS approach, the simulation results show that our approach can strike the balance between optimality of the solution and computational complexity.

HECS is within 0.6% of the optimal solution in the best case (traffic demand of 367). Table 5.6 shows that HECS is 2.51% from optimal solution within 2.17(s) on average.

**Simulation results for USA topology**

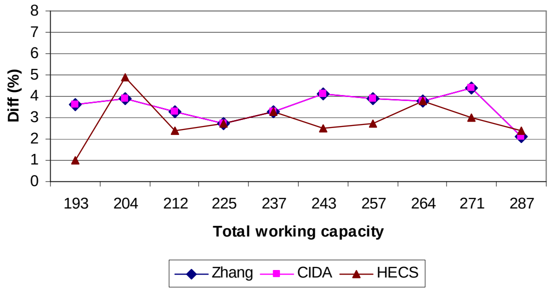
These results show that with our algorithm, an almost optimal spare capacity consumption can be achieved with much less running time than that required by ILP. It is found that we reduce computation time used by around $2 * 10^{-4}\%$ of that required to find the optimal solution. In the distribution of traffic demand between 520 and 530, HECS is 1.1% from the ILP solution and is around 8 times better than CIDA solution. We observe from Table 5.6 that our performed results are 2.92% of the optimal on average whereas CIDA is 7.34%. Furthermore, the results show that the average running time for HECS is slightly faster than that required by CIDA

Table 5.5: Simulation results for USA topology

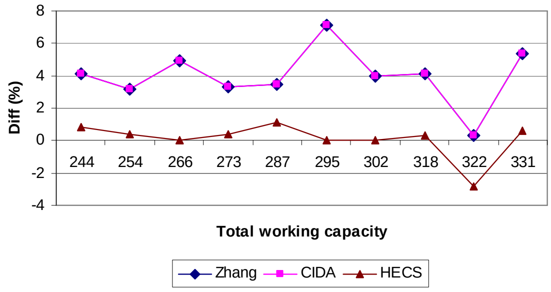
| | | **USA** (C = 32) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | **WC** | 458 | 467 | 476 | 485 | 494 | 507 | 515 | 526 | 537 | 544 |
| pure ILP | Re | 68.6 | 67.2 | 71 | 65.2 | 67.8 | 67.9 | 65.8 | 67.5 | 69.5 | 68.6 |
| | time(s) | 1098 | 21600 | 944 | 399 | 5109 | 872 | 593 | 2294 | 16775 | 21600 |
| CIDA | Re | 77.1 | 73.9 | 76.9 | 72.4 | 75.5 | 74 | 73.4 | 75.9 | 75.8 | 77.6 |
| | time(s) | 1.8 | 1.8 | 1.5 | 1.3 | 1.5 | 1.5 | 1.5 | 1.6 | 1.7 | 1.9 |
| Zhang | Re | 64.4 | 66.9 | 63.3 | 60.2 | 64 | 64.4 | 65 | 65.8 | 63.5 | 60 |
| | time(s) | 2.2 | 2.34 | 2.14 | 2.21 | 2.73 | 2.67 | 2.8 | 2.91 | 2.89 | 2.6 |
| HECS | Re | 72.7 | 71.3 | 75 | 68 | 70.5 | 70 | 67.6 | 68.6 | 72.4 | 72.2 |
| | time(s) | 1.4 | 1.3 | 1.6 | 1.4 | 1.4 | 1.6 | 1.4 | 1.6 | 1.5 | 1.7 |
| usage(%) | | 32.8 | 33.5 | 34.1 | 34.8 | 35.4 | 36.3 | 36.9 | 37.7 | 38.5 | 39 |

WC: working capacity, Re: Redundancy

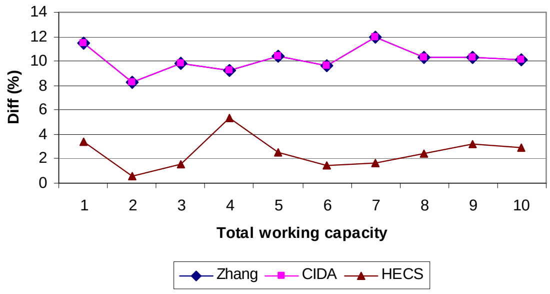
From the results, it is observed that the proposed HECS algorithm possesses a balance between performance of capacity redundancy and computational time. Therefore, HECS is more attractive for dense and large networks. As for the COST239, HECS can achieve near optimal solution within 1(s). For EON and USA, our results are superior to existing approaches and with very reasonable computation time.



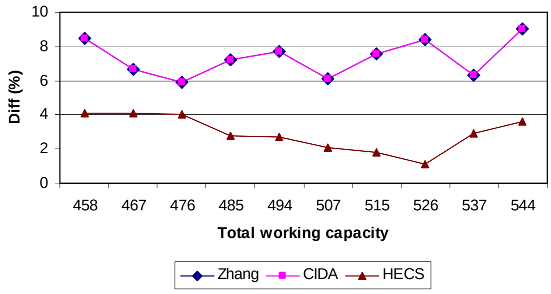

(a) NSFNet network




(b) COST239 network




(c) EON network




(d) USA network

Figure 5.1: Difference from the optimal solution for the four test networks

Table 5.6: Average results in each test network

| Performance | | Pure ILP | CIDA | Zhang | HECS |
|---|---|---|---|---|---|
| NSFNet | Re | 65.95 | 69.63 | 69.63 | 68.82 |
| | time | 1764 | 0.03 | 0.11 | 0.05 |
| | Diff | 0 | 3.68 | 3.68 | 2.87 |
| COST239 | Re | 30.21 | 34.2 | 34.2 | 30.23 |
| | time | 15767 | 0.22 | 0.22 | 0.3 |
| | Diff | 0 | 3.99 | 3.99 | 0.02 |
| EON | Re | 53.55 | 63.75 | 63.75 | 56.06 |
| | time | 9191 | 1.96 | 2.55 | 2.17 |
| | Diff | 0 | 10.2 | 10.2 | 2.51 |
| USA | Re | 67.91 | 75.25 | 75.25 | 70.83 |
| | time | 7128 | 1.61 | 2.96 | 1.49 |
| | Diff | 0 | 7.34 | 7.34 | 2.92 |

Re : Redundancy, Diff : Diff(%)

## 5.3 Conclusion

This chapter has presented and discussed a heuristic approach to $p$-cycle selection that avoids the use of the ILP model. We term this algorithm the Heuristic approach for Efficient Cycle Selection (HECS). The results show that the proposed algorithm can achieve near optimal solution in dense and large networks. Furthermore, this algorithm greatly reduces computational complexity in a dense networks when compared to the ILP model. We thus conclude that HECS is suitable to design $p$-cycle network because it has the balance between computational complexity and optimality of solution with 100% protection.

# Chapter 6

# Conclusion and Future Work

Optical networks based on WDM technology can carry very high bandwidths over fiber links in the network so that the capacity utilization can be dramatically increased. However, the high bandwidths carried by such links have the potential problem of the loss of a tremendous amount of capacity when a failure of a node or link occurs. Thus, network survivability is a very important issue in optical networks to prevent such huge losses. Various mechanisms have been investigated and examined by other researchers as discussed in chapter 2. The most promising is the pre-configured protection cycle ($p$-cycle) approach, which can offer the computation speed of ring networks and the capacity efficiency of mesh networks. To achieve optimal solution for $p$-cycle designs, the initial step requires all possible cycles in the network to be identified and then an ILP formulation is usually used to selects an optimized set of $p$-cycles. However, the ILP model has the drawback of high time complexity in dense networks since the number of cycles grows exponentially depending on the number of nodes and spans. In practice, however, also a few cycles are needed to achieve an optimal solution. There exists some heuristic approaches to tackle the problem as discussed in literature review. The heuristic methods are limited to a manageable number of cycles to achieve near optimal solutions within reasonable computation times.

In Chapter 1, we opened the thesis with a background of optical network and gave the basic concept of survivability. Chapter 1 also introduced our motivation and purpose in this thesis. Chapter 2 provided the explanation of network survivability mechanisms. The mechanisms were pre-planned protection, dynamic restoration, shared backup path protection (SBPP) and pre-configured protection cycle ($p$-cycle). These various existing methods for $p$-cycle design were reviewed. The existing methods were classified into two types: ILP formulation and heuristic approach. Basically, the ILP model offers optimal solutions using all cycles, but this has a computation time problem when the number of $p$-cycles grow exponentially in large networks. This issue gave us motivation to investigate heuristic approaches in this thesis. Heuristic approaches provide near optimal solution, but this introduces the problem of balancing between capacity utilization and time complexity. This issue in the heuristic approach has been the focus of our work.

Chapter 3 presented graph theory and the mathematical notations for the corresponding network design used in this thesis. Four test networks, named NSFNet, COST239, EON and USA, were introduced. We closed Chapter 3 with the assumptions used in the implementation of our designed approaches.

This thesis has proposed new heuristic techniques for balancing between capacity efficiency and computation time. To achieve our main objective, we have investigated two kinds of heuristic approaches: a heuristic approach based on an ILP formulation and a pure heuristic approach. Both have been evaluated by comparing them with a full ILP solution and existing heuristic approaches. In Chapter 4, we proposed our first approach to enumerate good candidate cycles for use by ILP to ensure 100% protection. It is based on a modified definition of network fundamental cycles. A set of candidate $p$-cycles is created by joining different fundamental cycles and the algorithm can achieve the best performances in terms of capacity utilization and time complexity. The simulated results show that the performance is close to optimal within reasonable computation time. For COST239 network, design efficiencies as low as 30% redundancy are achieved. In addition, we found from the results that small cycles are needed in order to achieve better performance as expected. In Chapter 5, we proposed a second approach based on a pure heuristic technique that does not use the ILP model. All cycles in the network under consideration have been pre-computed using PCA and then a set of suitable $p$-cycles are determined based on three proposed three factors. The results show that the proposed method outperforms the exist heuristic approach. Average redundancy in all test networks is within 3% of optimal solution and 2.5(s) of computation time.

Thus, we can conclude that both approaches are of great use in achieving the best performance. For dense and large networks, the proposed pure heuristic approach performs very close to ILP solution within acceptable computation times. It can meet our objective in terms of balancing between the optimality of solutions and the time complexity.

## 6.1 Future Works

In this thesis, we consider the logical topology design in the context of network survivability with the objectives of balancing between utilization of network capacity and restoration time. However, the topology design has been researched under various assumptions. Thus, there are some possibilities for further extensions to the current work and these are itemized below.

- This thesis on $p$-cycle design has been limited to static traffic. It aims to balance between fast restoration time and high capacity efficiency. However, survivability in dynamic networks is more complex and it will be hard to achieve this objective. This is because the traffic demands arrive randomly and we do not have concise information on the traffic pattern. This can lead to high blocking probability and reduce the protection against network failure. Thus, dynamic traffic in real network is an important field of study for the design of survivable networks.

With dynamic traffic, every link may be required to carry many different traffic demands. In general, some of the capacity on the link is reserved to provide 100% protection. It is then desired to minimize the total the reserved capacity for protection.

- In optical WDM networks, the *wavelength continuity* problem is an important issue to set up lightpaths as discussed earlier in this thesis. But, in this thesis we made the assumption that wavelength convertors (WCs) are available at every node of the network. This assumption makes it possible to provide good performance in terms of blocking probability. However, every node in real networks are unlikely to be equipped with full wavelength conversion. This is because WC is still costly equipment. WCs trade off between cost and capacity utilization. A challenge in WC is to search a node location which can be equipped with wavelength conversion. Hence, another extension is to optimize wavelength converter placement, i.e.,how many to use and where to place them.

- In reality, network survivability should provide 100% restorability against a single link failure or dual link failures. However, most researchers have focused on protection under single link failure since dual link failure protection is more difficult to solve and find the optimal solution. This thesis also considered single link failure in the context of the network survivability. A few studies have proposed a new framework to protect against dual link failures [73, 74]. These aim to offer capacity optimization for surviving dual link failures in mesh optical WDM networks. However, this research is still at single link failure. Therefore, another extension to this thesis is to investigate the network capacity design for protection against a mix of dual link failures and single link failure

# Bibliography


[1] M. llyas and H. T. Mouftah, *The handbook of optical communication networks.* New york: CRC press, 2003.

[2] e. a. A. R. Mikler, "An object-oriented approach to modeling and simulation of routing in large communication networks," tech. rep., Department of computer science, 1998.

[3] R. Ramaswami and K. N. Sivarajan, *Optical networks : a practical perspective.* Morgan kaufmann, second ed., 2002.

[4] D. Medhi and D. Tipper, "Multi-layered network survivability - models, analysis, architecture, framework and implementation: an overview," in *The DARPA, Information Survivability Conference and Exposition*, 2000.

[5] E. Karasan and E. Ayanoglu, "Effects of wavelength routing and selection algorithm on wavelength conversion gain in wdm optical networks," *IEEE/ACM Transactions on networking*, vol. 6, no. 2, pp. 186 – 196, 1998.

[6] B. Ramamurthy and B. Mukherjee, "Wavelength conversion in wdm networking," *IEEE Journal on selected areas in communications*, vol. 16, no. 7, pp. 1061 – 1073, 1998.

[7] J. M. yates and M. P. rumsewicz, "Wavelength convertes in dynamically reconfigurable wdm networks," 1999.

[8] S. Ramamurthy, L. Sahasrabuddhe, and B. Mukherjee, "Survivable wdm mesh networks," *Lightwave Technology, Journal of*, vol. 21, no. 4, pp. 870–883, 2003.

[9] S. Ramamurthy and B. Mukherjee, “Survivable WDM mesh networks. Part I-protection,” in *Proceedings of 8th Annual Joint Conference of the IEEE Computer and Communications Societies. INFOCOM’99*, vol. 2, (New York, USA), pp. 744–751, 1999.

[10] S. Ramamurthy and B. Mukherjee, “Survivable WDM mesh network.II. restoration,” in *IEEE International Conference on Communications*, vol. 3, (Vancouver, BC Canada), pp. 2023 – 2030, 1999.

[11] D. S. W. D. Grover, “Bridging the ring-mesh dichotomy with p-cycles,” in *Design of Rerliable Communication Networks (DRCN 2000)*, (Munich, Germany), pp. 92 – 104, 2000.

[12] D. S. W. D. Grover, “Self-organizing closed path configuration of restoration capacity in broadband mesh transprot networks,” in *Canadian Conference on Broadband Research(CCBR)*, (Ottawa, Canada), June, 1998.

[13] G. W. D., *Mesh-Based Survivable Networks.* Prentice Hall, 2003.

[14] P. Green, “Optical networking update,” *IEEE Journal on selected areas in communications*, vol. 14, no. 5, pp. 764 – 779, June, 1996.

[15] D. Cotter, J. K. Lucek, and D. Marcenac, “Ultra high bit rate networking: From the transcontinental backbone to the desktop,” *IEEE communications magazine*, pp. 90–95, April 1997.

[16] M. Marhic, “Coherent optical cdma networks,” *Journal of Lightwave Technology*, vol. 11, pp. 729–735, 1993.

[17] O. Gerstel and S. Kutten, “Dynamic wavelength allocation in all optical ring networks,” in *IEEE International Conference on Communications (ICC)*, vol. 1, (Canada), pp. 432–436, 1997.

[18] HuiZang, *WDM mesh networks: Management and survivability.* 2003.

[19] K. Chan and T. Yum, "Analysis of least congested path routing in wdm lightwave networks," in *IEEE INFOCOM*, vol. 2, (Canada), pp. 962 – 969, 1994.

[20] L. Li and A. K. Somani, "Dynamic wavelength routing using congestion and neighborhood information," *IEEE/ACM Transactions on networking*, vol. 7, pp. 779 – 786, 1999.

[21] I. Chlamtac, A. Ganz, and G. Karmi, "Purely optical networks for terabit communication," in *IEEE INFOCOM*, vol. 3, (Washington DC), pp. 887 – 896, 1989.

[22] A. Birman and A. Kershenbaum, "Routing and wavelength assignment methods in single hop all optical networks with blocking," in *IEEE INFOCOM*, vol. 2, (Boston), pp. 431 – 438, 1995.

[23] E. Karasan and E. Ayanoglu, "Effects of wavelength routing and selection algorithms on wavelength conversion gain in wdm optical networks," *IEEE/ACM Transactions on networking*, vol. 6, pp. 186 – 196, 1998.

[24] X. Zhang and C. Qiao, "Wavelength assignment for dynamic traffic in multi-fiber wdm networks," in *International Conference on Computer Communications and Networks*, (Lafayette), pp. 479 – 485, 1998.

[25] H. Harai, M. Murata, and H. Miyahara, "Performance of alternate routing methods in all optical switching networks," in *IEEE INFOCOM*, vol. 2, pp. 516 – 524, 1997.

[26] B. Li and X. Chu, "Routing and wavelength assignment vs. wavelength converter placement in all optical networks," *IEEE Communication Magazine*, vol. 41, pp. S22 – S28, August 2003.

[27] R. A. Barry and S. Subramaniam, "The max-sum wavelength assignment algorithm for wdm ring networks," in *Optical Fiber Communication Conference and Exhibit (OFC)*, (Dallas), pp. 121 – 122, 1997.

[28] G. Jeong and E. Ayanoglu, "Comparison of wavelength interchanging and wavelength selective cross connects in multiwavelength all optical networks," vol. 1, (San Francisco), pp. 156 – 163, 1996.

[29] S. Subramaniam and R. A. Barry, "Wavelength assignment in fixed routing wdm networks," in *IEEE International Conference on Communications (ICC)*, vol. 1, (Canada), pp. 406 – 410, 1997.

[30] e. a. T. Durhuus, "All optical wavelength conversion by semiconductor optical amplifiers," *Journal of Lightwave Technology*, vol. 14, pp. 942 – 954, 1996.

[31] Z. Chu, J. Liu, and Z. Zhang, "Analysis of sparse partial wavelength conversion in wavelength routed wdm networks," in *IEEE INFOCOM*, vol. 2, (USA), pp. 1363–1371, 2004.

[32] M. A. S. Subramaniam and A. Somani, "All optical networks with sparse wavelength conversion," *IEEE/ACM Transactions on networking*, vol. 4, pp. 544 – 557, 1996.

[33] A. S. Arora and S. Subramaniam, "Converter placement in wavelength routing mesh topologies," in *International Conference on Communication (ICC)*, vol. 3, (New Orleans), pp. 1282 – 1288, 2000.

[34] X. Chu, B. Li, and Z. Zhang, "A dynamic rwa algorithm in a wavelength routed all-optical network with wavelength converters," in *IEEE INFOCOM*, vol. 3, pp. 1795 – 1804, 2003.

[35] M. A. S. Subramaniam and A. K. Somani, "On the optimal placement of wavelength converters in wavelength routed networks," in *IEEE INFOCOM*, vol. 2, pp. 902 – 909, 1998.

[36] X. Chu, B. Li, and C. I., "Wavelength converter placement under different rwa algorithms in wavelength routed all optical networks," *IEEE/ACM Transactions on networking*, vol. 51, pp. 601 – 217, 2003.

[37] G. Xiao and Y. W. Leung, "Algorithms for allocating wavelength converters in all optical networks," *IEEE/ACM Transactions on networking*, vol. 7, no. 4, pp. 545 – 557, 1999.

[38] D. Zhou and S. Subramanian, "Survivability in optical networks," *IEEE Networks*, 2000.

[39] T. Ho and L. Rucgard C, "A class of self-healing ring architectures for sonet network applications," in *GLOBECOM*, pp. 444–449, 1990.

[40] H. Luss, M. Rosenwein, and R. Wong, "Topological network design for SONET ring architecture," *Systems, Man and Cybernetics, Part A, IEEE Transactions on*, vol. 28, no. 6, pp. 780–790, 1998.

[41] J. Doucette and W. D. Grover, "Comparion of mesh protection and restoration schemes and the dependency on graph connectivity," in *Internation Workshop on Design of Raliable Communication Networks (DRCN 2001)* (3th, ed.), (Budapest, Hungary), pp. 121–128, 2001.

[42] B. Fortz, P. Soriano, and C. Wynants, "A tabu search algorithm for self healing ring network design," tech. rep., Universite Libre de Bruxelles, May 23 2000.

[43] G. Morley and W. Grover, "Current approaches in the design of ring-based optical networks," (Edmonton, Alberta, Canada), pp. 220 – 225, IEEE canadian Coference on Electrical and Compute Engineering Shaw Conference Center, 1999.

[44] W. D. Zhong, Z. Zhang, and B. Mukherjee, "Fast and dynamically pre-configured protection cycles for survivable wdm networks," in *9th Optoelectonics and Communications Conference (OECC 2004)*, (Yokohama, Japan), pp. 12 – 16, July, 2004.

[45] R. R. Iraschko, *Path Resotrable Networks.* PhD thesis, University of Alberta, Spring 1997.

[46] A. Kodian and W. D. Grover, "Failure-independent path protecting p-cycles: Efficient and simple fully preconnected optical path protection," *Lightwave Technology*, vol. 23, no. 10, pp. 3241 – 3259, 2005.

[47] F. J. Blouin, A. Sack, W. D. Grover, and H. Nasrallah, "Benefits of p-cycles in mixed protection and restoration approach," *IEEE Communications Networks*, 2003.

[48] D. S. Wayne D. Grover, "Cycle-oriented distributed preconfiguration bing-link speed with mesh-like capacity for self-planning network restoration," in *IEEE ICC*, pp. 537–543, June 7-11, 1998.

[49] C. Liu and L. Ruan, "Finding good candidate cycles for efficient p-cycle network design," in *13th international Conference on Computer Communication and Networks(ICCCN 2004)*, pp. 321 – 326, 2004.

[50] W. Grover and J. Doucette, "Advances in optical network design with p-cycles: Joint optimization and pre-selcetion of candidate p-cycles," tech. rep., 2002.

[51] D. A. Schupke, C. G. Gruber, and A. Autenrieth, "Optimal configuration of p-cycles in wdm networks.," in *IEEE International Conference on Communication (ICC)*, vol. 5, (USA), pp. 2761 – 2765, 2002.

[52] D. A. Schupke, "An ilp for optimal p-cycle selection without cycle enumeration," in *Eighth IFIP Working conference on optical Network Design and Modleing )ONDM 2004)*, (Ghent, Belgium), 2003.

[53] H. N. Nguyen, D. Habibi, V. Q. Phung, S. Lachowicz, K.Lo, and B. Kang, "Joint optimization in capacity design of networks with p-cycle using the fundamental cycle set," in *IEEE GLOBECOM*, (USA), 2006.

[54] H. Zhang and O. Yang, "Finding protection cycles in dwdm networks," in *IEEE International Conference on Communication*, vol. 5, pp. 2756–2760, 2002.

[55] J. Doucette, D. He, W. D. Grover, and O. Yang, "Algorithmic approaches for efficient enumeration of candidate p-cycles and capacitated p-cycle network design," *Design of Reliable Communication Networks (DRCN) 2003*, pp. 212 – 220, 2003.

[56] Z. Zhang, W.-D. Zhong, and B. Mukherjee, "A heuristi method for design of survivable wdm networks with p-cycles," *IEEE Communications Letters*, vol. 8, no. 7, pp. 467 – 469, 2004.

[57] Z. Zhang, W. Zhong, and B. Mukherjee, "A heuristic algorithm for p-cycles configuration in wdm optical networks," in *Opto-Elecrtoinics and Communications Conference(OECC)*, vol. 23, pp. 568 – 569, 2003.

[58] A. Kodian, A. Sack, and W. D. Grover, "p-cycle network design with hop limits and circumference limits," in *First International Conference on Broadband Networks(BROADNETS 2004)*, (USA), pp. 224 – 253, Oct. 2004.

[59] C. G. Gruber and D. A. Schupke, "Capacity-efficient planning of resilient networks with p-cycles," in *in 10th international Telecom. Network Strategy and Planning Symposium*, (Munich, Germany), Jun, 2002.

[60] D. Stamatelakis and W. Grover, "Theoretical underpinnings for the efficiency of restorable networks using preconfigured cycles ($p$-cycles)," *Communications, IEEE Transactions on*, vol. 48, no. 8, pp. 1262–1265, 2000.

[61] H. Huang and J. A. Copeland, "A series of hamiltonian cycle-based solutions to provide simple and scalable mesh optical network resilience," *IEEE communications magazine*, pp. pp. 46 – 51, Nov. 2002.

[62] A. Sack and W. D. Grover, "Hamiltonian $p$-cycles for fiber-level protection in homogeneous and semi-homogeneous optical networks," *IEEE Network magazine*, pp. 49 – 56, March/April 2004.

[63] S. S. Heydari, W. Mardini, and O. Yang, "Theoretical analysis of restorability of hamiltonian protection cycles in random mesh networks," in *9th IEEE Symposium on Computers and Communication (ISCC 2004)*, (Alexandria, Egypt), pp. 921 – 926, July 2004.

[64] C. G. Gruber, *Deployment of p-cycles in WDM Networks.* PhD thesis, Technische University Munchen, August, 2001.

[65] R. Bhandari, *Survivable Networks: Algorithms for Diverse Routing.* USA: Kluwer Academic, 1999.

[66] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms.* New York: BcGraw-Hill, 1990.

[67] Y. Lim and H. Kim, "A shortest path algorithm for real road network based on path overlap," *Journal of the Eastern Asia Society for Transportation Studies*, vol. 6, pp. 1426 – 1438, 2005.

[68] K. Lo, D. Habibi, Q. V. Phung, H. N. Nguyen, and B. Kang, "Dynamic p-cycles selection in optical wdm mesh networks," in *International Conference on Networks (ICON2005)*, (Malaysia), 2005.

[69] M. Tornatore, G. MAier, and A. Pattavina, "Wdm network optimization by ilp based on source formulation," in *IEEE INFOCOM*, vol. 3, pp. 1813 – 1821, 2002.

[70] M. P. N. Deo and M. S. Krishnamoorty, "Algorithms for generating fundamental cycles in a graph," *ACM transactions on Mathematical Software*, vol. 8, no. 1, pp. 26 – 42, 1982.

[71] C.C.Gotlieb and D. Corneil, "Algorithms for finding a fundamental set of cycles for an undirected linear graph," *communications of the ACM*, vol. 10, pp. 780 – 783, 1967.

[72] K. Paton, "An algorithm for finding a fundamental set of cycles of a graph," *communications of the ACM*, vol. 12, pp. 514 – 518, 1969.

[73] H. CHoi, S. Subramaniam, and H. A. Choi, "On double link failure recovery in wdm optical networks," in *The 21st Annual Joint Conference of the IEEE Computer and Communications Societies*, (New York), pp. 808 – 816, 2002.

[74] W. He, M. Sridharan, and A. K. Somani, "Capacity optimization for surviving double-link failures in mesh-restorable optical networks," in *Photonic Network Communications*, vol. 9, pp. 99 – 111, 2005.