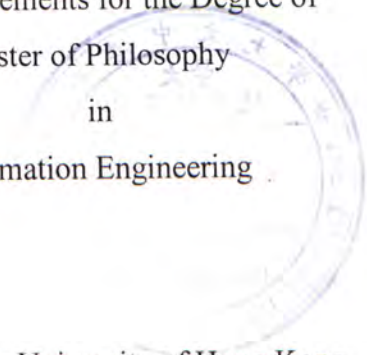


# Survivable Network Design of All-Optical Network

Kwok-Shing Ho

A Thesis Submitted in Partial Fulfilment of  
the Requirements for the Degree of  
Master of Philosophy  
in  
Information Engineering



The Chinese University of Hong Kong

July 2002

The Chinese University of Hong Kong holds the copyright of this thesis. Any person(s) intending to use a part or whole of the materials in the thesis in a proposed publication must seek copyright release from the Dean of the Graduate School.



# Survivable Network Design of All-Optical Network

Submitted by Kwok-Shing Ho

for the degree of Master of Master of Philosophy in

Department of Information Engineering at

The Chinese University of Hong Kong in July, 2002

## Abstract

Future communication networks will carry many WDM channels at very high bit rate. Thus, it is desirable to avoid electronic switching at the core. The all-optical backbone network can be interconnected by optical cross-connects at strategic locations to allow for flexible capacity provisioning and fault-tolerant rerouting. Such an all-optical core layer nicely decouples the long-term capacity planning problem from the short-term dynamic bandwidth allocation problem which can be better tackled in the electronic domain. An essential requirement for the all-optical core layer is that it must be fully fault-tolerant, otherwise, a single failed link can cause a disaster for the entire network.

We consider two all-optical network design problem in this thesis. The first one is the problem of how to allocate the spare capacities on a given all-optical core network topology with working capacity planned, so as to make the network fully single-fault (or multi-faults) tolerant. The objective function is minimizing the total cost of the spare fibers. This problem is NP-complete. We have developed an algorithm called "SCAPE" to tackle this problem and obtain results that are comparable to those obtained by integer programming but with a much smaller complexity. Extensive numerical results are shown. SCAPE can also tackle non-linear objective functions while integer programming cannot efficiently tackle.

The second problem is the extension of the first problem to arbitrary topology. This problem is also NP-complete. We have developed two efficient heuristics, the Modified Drop Algorithm (MDA) and the Genetic Algorithm (GA), to tackle this problem based on the previously developed algorithm SCAPE. Joint optimization for the topology design, working capacity and spare capacity planning is shown for the first time, and either MDA or GA can result in a solution that is 14% to 16% better than the ones obtained without joint optimization. GA can give results that are a few percent better than MDA in general but the computational complexity is about 300 times higher than that of MDA for a ten-node network. Summarizing, a basic framework of how to design a survivable all-optical network is presented in this thesis.

## 摘要

未來通訊網絡將運載許多非常高比特率的波分覆用通信渠道。因此，避免電子開關出現於這個網絡核心中才是理想的情況。在策略性的計劃地點中設置光纖交換機所鋪蓋成的全光纖中樞網絡，使網絡容量的供應更有彈性以及容許容錯改線。這樣的全光纖中樞網絡恰好地將長期性的網絡容量規劃的問題，從於電子領域當中比較易於處理的短期性動態頻寬分配問題當中分開出來。全光纖中樞網絡的一個根本要求是網絡必須有充分容錯的能力，否則，一條光纖電纜的失效就會為整個網絡帶來災難性的影響。

在這份論文當中，我們將會考慮兩個有關全光纖網絡的設計問題。第一個問題考慮到在一個已有拓樸結構以及工作容量的全光纖中樞網絡中，如何分配備用容量，使考慮中的全光纖網絡可以充分容許一個或者多個錯誤的發生。設計的目標是要將備用容量所帶來的額外費用減到最少。這個問題是 NP 完全的。我們開發了一個名為 SCAPE 的算法去到處理這個問題，這算法複雜性遠低於整數編程的算法，卻獲得同整數編程差不多的效果。另一方面，SCAPE 有能力處理整數編程未能有效處理的非線性目標函數。

第二個問題將第一個問題延伸到任意的網絡拓樸結構上面。這個問題也是 NP 完全的。基於先前所開發的 SCAPE 算法，我們開發了兩個算法，修改過的下落數法(MDA)以及基因算法(GA)去到處理這個問題。拓樸結構設計，工作容量同備用容量計劃同時聯合優化的做法在此是首次被提出，並且無論是 MDA 或是 GA 都能夠得出比沒有同時聯合優化時 14%到 16%不等的改善。以十個節點為考慮的網絡設計問題上，GA 一般情況下都表現優於 MDA 幾個百分比，但在運算的複雜程度上卻高於 MDA 三百倍左右。總結，這份論文提出，一個如何設計完全容錯的全光纖網絡的基本框架。

# Table of contents

List of Figures	vi
List of Tables	vii
<b>Chapter 1 Introduction</b>	<b>1</b>
1.1 Overview	1
1.2 Thesis Objectives	6
1.3 Outline of Thesis	8
<b>Chapter 2 The Spare Capacity Planning Problem</b>	<b>9</b>
2.1 Mathematical Model of the Spare Capacity Planning Problem	12
2.1.1 Variable Definitions	12
2.1.2 Objective Function and Constraints	15
2.1.3 Complexity	17
2.2 Greedy Algorithm - Spare Capacity Allocation and Planning Estimator (SCAPE)	19
2.2.1 Working Principle of SCAPE	20
2.2.2 Implementation of SCAPE	22
2.2.3 Improved SCAPE	23
2.3 Experimental Results and Discussion	27
2.3.1 Experimental Platform	27
2.3.2 Experiment about Accuracy of SCAPE	27
2.3.3 Experiment about Minimization of Network Spare Capacity	30
2.3.4 Experiment about Minimization of Network Spare Cost	35
2.4 Conclusions	38
<b>Chapter 3 Survivable All-Optical Network Design Problem</b>	<b>39</b>
3.1 Mathematical Model of the Survivable Network Design Problem	42

3.2	Optimization Algorithms for Survivable Network Design Problem	44
3.2.1	Modified Drop Algorithm (MDA)	45
3.2.1.1	Drop Algorithm Introduction	45
3.2.1.2	Network Design with MDA	45
3.2.2	Genetic Algorithm	47
3.2.2.1	Genetic Algorithm Introduction	47
3.2.2.2	Network Design with GA	48
3.2.3	Complexity of MDA and GA	51
3.3	Experimental Results and Discussion	52
3.3.1	Experimental Platform	52
3.3.2	Experiment about Accuracy of MDA and GA	52
3.3.3	Experiment about Principle of Survivable Network Design	55
3.3.4	Experiment about Performance of MDA and GA	58
3.4	Conclusions	62
	<b>Chapter 4 Conclusions and Future Work</b>	<b>63</b>
	<b>Appendix A The Interference Heuristic for the path restoration scheme</b>	<b>66</b>
	<b>Bibliography</b>	<b>69</b>
	<b>Publications</b>	<b>72</b>

## List of Figures

Figure 1.1	A ring typed network topology with protection scheme	3
Figure 1.2	A mesh typed network topology with restoration scheme	3
Figure 2.1	Link Restoration	9
Figure 2.2	Path Restoration	9
Figure 2.3	An Example Network used to illustrate notations	14
Figure 2.4	An Example Network for Working Principle of SCAPE	20
Figure 2.5	A 4-node Network with a Uniform Traffic Demand Matrix	28
Figure 2.6	Distance Metric of a 4-node Network	29
Figure 2.7	Cost Metric of a 4-node Network	29
Figure 2.8	Network 1 for Experiment 2.3.3	31
Figure 2.9	Network 2 for Experiment 2.3.3	31
Figure 2.10	Network 3 for Experiment 2.3.3	31
Figure 2.11	Network 4 for Experiment 2.3.3	31
Figure 2.12	Topology of Hasegawa Network	36
Figure 2.13	Distance Metric of Hasegawa Network	36
Figure 2.14	Cost Metric of Hasegawa Network	36
Figure 2.15	Spare Capacity Assignment by Herzberg	37
Figure 2.16	Spare Capacity Assignment by Grover	37
Figure 2.17	Spare Capacity Assignment by SCAPE	37
Figure 3.1	A 4-node Network with a Uniform Traffic Demand Matrix	53
Figure 3.2	Distance Matrix for Experiment 3.3.3	56
Figure 3.3	$\beta_{ij}$ Matrix for Experiment 3.3.3	56
Figure 3.4	$\alpha_{ij}$ Matrix for Experiment 3.3.3	56
Figure 3.5	Traffic Matrix for Experiment 3.3.3	56

## List of Tables

Table 2.1	Possible cases of rerouting and flow assignment of the 4-node network	28
Table 2.2	Results of using C program and SCAPE to minimize the total spare capacity of the 4-node network	29
Table 2.3	Results of using C program and SCAPE to minimize the total spare cost of the 4-node network	30
Table 2.4	General information of the four test networks of Experiment 2.3.3	31
Table 2.5	Total Spare Capacity for Link Restoration	32
Table 2.6	Total Spare Capacity for Path Restoration	33
Table 2.7	The running time of the SCAPE	33
Table 2.8	Result of Experiment 2.3.4	37
Table 3.1	All possible combination of working routes for a 4-node network	53
Table 3.2	Result and comparison of Experiment 3.3.2	54
Table 3.3	Comparison of network costs with or without joint optimization	57
Table 3.4	Comparison of MDA and GA with respect to various cost parameters	59
Table 3.5	The average result and running time for all the cases in Table 3.4	59



# Chapter 1 Introduction

## 1.1 Overview

The demand for an ever-increasing bandwidth in telecommunication services is ever-expanding. Services include video on demand, high-resolution medical image archiving and retrieval, multimedia document distribution and many to come. Such communication applications abound that require the high bandwidths are available only with optical fibers. In this way, the high-bandwidth optical network has become the core of the telecommunication networks both currently and in the foreseeable future.

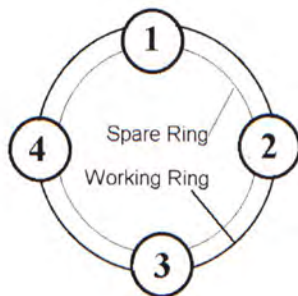
Since future communication networks will carry many WDM channels at very high bit rate, it is desirable to avoid electronic switching at the core. The all-optical backbone network can be interconnected by optical cross-connects (OXC) at strategic locations to allow for flexible capacity provisioning and fault-tolerant rerouting. Such an all-optical core layer nicely decouples the long-term capacity planning problem from the short-term dynamic bandwidth allocation problem which can be better tackled in the electronic domain. Owing to the rapid advance of dense wavelength division multiplexing (DWDM) technology and the cost reduction of optical components, all-optical backbone network has already begun to appear at the core of many telecommunication networks. Therefore, it is important to consider the network architecture design problem. Even though the hardware technology is advancing so fast, the question of how to lay out the most cost effective network with these components is still a wide open problem [1].

A network architecture design problem involves combining the building blocks such as the optical components in an optical network in a specific way to fulfill the traffic demands and other requirements of that network. One essential requirement for an all-optical core layer is that it must be fully fault-tolerant. A network which is single-fault-tolerant should be at least 2-connected and sufficient spare capacity on the links should be available for alternate-route routing. For the high-capacity all-optical network, the high capacity of fiber facilities enables each link to carry larger amounts of traffic. In the event of a link or node failure, the service loss could be very severe. Several highly publicized outages have illustrated that disruption of communication services is very costly to business, governments and the general public. One example is what happened in 1987 when a link that belonged to a major backbone carrier facility was severed, more than 125,000 trunks were put out of service and an estimated 100,000 connections were lost 2 seconds after the cut. With a cost of millions of dollars and time of more than 2 hours, the services were restored manually with the use of some of the capacity on physically diverse routes [1]. That shows that survivability is definitely needed during the network architecture design and the spare capacity planning becomes an important problem at the same time.

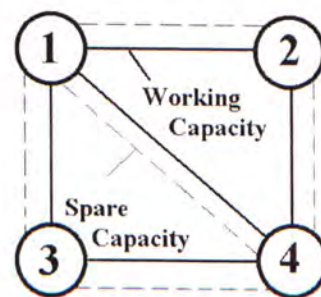
Another important requirement is the cost. If the network cannot be planned cost-effectively, this will waste a lot of network resources. Therefore, a reasonable network architecture design of the all-optical core network should combine the building blocks cost-effectively with the fulfillment of the traffic requirements as well as the fault-tolerant requirement.

Obviously, a ring typed network topology with an automatic protection scheme, shown in figure 1.1, is an attractive solution for this network architecture design problem. This approach enables fast reaction on a failure with simple implementation. However, a fundamental limitation is that survivable ring technology is unable to exploit the very low redundancy levels that are theoretically sufficient for full survivability. As a specialized form of 1:1 protection switching systems, rings fundamentally require a minimum of 100% duplication of working and spare capacities [2]. This is not a cost effective design.

Another possible choice is the mesh typed network with a restoration scheme, shown in figure 1.2. Such kind of network architecture can yield full survivability with redundancy that is proportional to the average nodal degree ( $n$ ) of the network. For real networks, which are generally mesh networks with  $3 < n < 5$ , there is the prospect of full survivability with as little as 30% to 50% redundancy [2]. This is because in meshed networks, rerouting strategies can be applied more efficiently. That is, the spare capacity on any link is not really dedicated to protect the working capacity of that particular link, but is shared among several working



**Figure 1.1** A ring typed network topology with protection scheme.



**Figure 1.2** A mesh typed network topology with restoration scheme.

entities. Theoretically, this network architecture solution not only maintains the survivability but also achieves a cost-effective use of the network resources. However, the planning of an optimized survivable mesh network is a very complex combinatorial optimization problem, as it involves topology design, routing, planning of working capacity, rerouting and planning of spare capacity. These sub-problems are all very complex individually. Even when the topology, routing scheme and work capacity assignment of a network are given, the optimal spare capacity planning of mesh network with a restoration scheme alone is NP-complete. It can be imagined how complex the survivable network design problem for an arbitrary topology mesh network will be. Therefore, many researchers have developed heuristics for solving the problem.

Until now some research works available handle the sub-problem of spare capacity planning only [3-10,12], some others handle the sub-problem of working and spare capacity planning jointly [10] and still some handle the sub-problem of topology design and working capacity planning jointly [3]. However, up to my best knowledge, no research work has addressed the whole survivable mesh network design problem which handles the topology design, working capacity planning and spare capacity planning altogether, as the problem is too complex to tackle. An efficient heuristic for solving this kind of problems is essential.

Summarizing, the network architecture design will be an important issue in the development of the all-optical core communication network. One essential requirement is the survivability. That makes the spare capacity planning an important problem. Another essential requirement is cost optimization. The mesh

type network with a suitable restoration scheme is a potential solution for the network architecture design that satisfies both survivability and cost requirements. However, the designing of a cost-optimized survivable mesh type networks is a very big and complex combinatorial optimization problem that involves the topology design, routing, planning of working capacity, rerouting and planning of spare capacity. This requires an efficient heuristic to solve.

## 1.2 Thesis Objectives

The objective of this thesis is to build up a generic framework for designing mesh topology survivable network with optimal cost and to provide some insights for designing efficient heuristics in dealing with these problems. The first problem to be studied is the spare capacity planning problem: given a mesh network topology which is at least 2-connected, the normal traffic demand (more accurately, the peak traffic demand), and the capacity allocation to meet the traffic demand, how much spare capacity should be provisioned and where it should be located in order for the network to tolerate a single link failure with a minimal total network cost?

Mathematical programming methods have been used to formulate the spare capacity planning problem for link and path restoration schemes such as the Integer Programming (IP) approaches [3,5,7-10]. The objective function adopted is to minimize the spare capacity required for achieving restoration from a specific failure condition. However, the resulting IP formulation is NP-hard and in order to solve the problem, sub-optimal heuristic approaches have been tried. In recent years, many heuristics have been proposed for finding a reasonable spare capacity assignment that meets the fault-tolerance requirement in polynomial time [3-10,12]. It is widely believed that the best approach is to approximate the IP model by a Linear Programming (LP) model, which is polynomial-time bounded and then round the solution to integer values [3,5,7-10]. However, the limitation of this approach is that the complexity of this approach scales with the exponential form of the number of links and nodes of the network. Therefore, the size of the problem has to be reduced by limiting the path sets.

However, it is still an open problem in how to pick the path sets in order to achieve good results. Another open problem is that when considering the real situation in the design of the whole survivable network, nonlinear variables such as nonlinear capacity cost function or quality of service (QoS) variables may be introduced. This problem is not yet tackled. This thesis aims at building a framework for tackling the spare capacity planning problem first and then develop insights to design efficient heuristics which can handle the nonlinear survivable network design problem.

Based on the framework built up, we can tackle the second network design problem for an arbitrary network topology. The problem is stated as follows: given the physical node locations and the normal traffic demand, how to create a mesh network topology and allocate capacity to the links to meet the given traffic condition that can tolerate a single link failure with minimal total network cost? This problem is not yet tackled previously. This thesis will investigate this problem and find an efficient heuristic that can give a good result in a reasonable time.

### **1.3 Outline of Thesis**

The overall structure of the thesis is as follows. Chapter 1 gives an overview of the problems. In Chapter 2, the spare capacity planning problem (the first problem stated) will be tackled. Mathematical model and the optimization techniques used are explained. Simulation for both link restoration and path restoration and the results are presented. The heuristic will be evaluated with a comparison with other heuristics. The influence of this heuristic is discussed too. Chapter 3 describes the second problem. The mathematical model and the optimization technique used are suggested. Simulation results are presented and the performance will be discussed. Finally, in Chapter 4, the future work will be described and a conclusion will be drawn.



## Chapter 2 The Spare Capacity Planning Problem

In this chapter, we investigate the spare capacity planning problem for survivable networks. The need for reliable communication service has become extremely important for high capacity networks. In order to provision for network survivability, spare capacity must be provided on the existing built-up network under a chosen restoration strategy. There are two main restoration strategies: Link restoration and Path restoration. In link restoration, the broken traffic is rerouted between the end nodes of the failed link. In path restoration, the origin

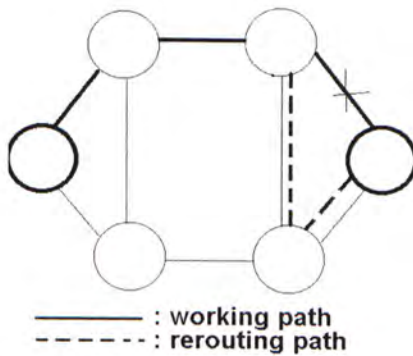


Figure 2.1 Link Restoration

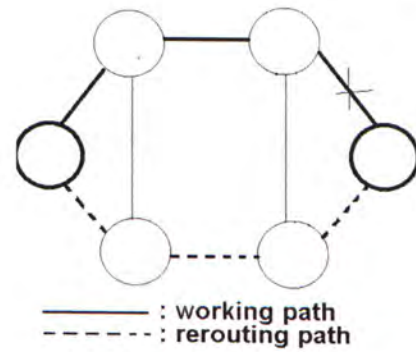


Figure 2.2 Path Restoration

destination node pairs (OD pairs) whose traffic traversed the failed device are responsible for restoration and reroute over the entire path set between each affected origin destination pair. The two restoration strategies are shown in Figure 2.1 and Figure 2.2.

If the spare capacity planning is not optimally done, it will cause a wasteful and inefficient use of resources. However, optimal spare capacity placement in a mesh restorable network has shown to be NP-hard [4]. Based on this reason, many researchers have attempted other approaches to solve this problem [5-11].

Grover suggested the Spare Link Placement Algorithm (SLPA) heuristic for solving the spare capacity placement problem in a mesh network with link restoration [6]. The SLPA is a heuristic approach with polynomial time complexity, and has been implemented in some telecom network for spare capacity planning. In this approach, the spare capacity allocation with full restorability is achieved by iterative link addition in the phase of “forward synthesis” and then the spare capacity will be reduced while maintaining the network restorability in the phase of “design tightening”.

Sakauchi, Herzburg and Venables attempted an IP approach based on max-flow min-cut considerations to solve the spare capacity placement problem [5,7,8]. This approach uses cut sets in the IP formulation and populates the constraint sets with cut sets iteratively until the solution provides a spare capacity placement which is 100% link restorable and minimizes the total spare capacity of the network. This approach can lead to a better spare capacity planning than SLPA while the complexity is higher. However, both of these approaches can be used for link restoration only.

A more recent and advantageous approach is still an IP approach but with flow constraints based on a suitable set of predefined routes over which restoration path sets may be implemented [9]. This is referred to as the flow-based approach. In this approach, the IP is based on eligible restoration routes between each pair of nodes terminating a link. This IP flow-based approach is preferred over the IP cut set approach because only a single IP execution is needed to obtain the spare capacity assignment. The same process also yields the exact routing used to

restore each link failure. This approach has been extended to path restoration and a good result was achieved [10]. However, the IP-based approaches cannot provide any insight in how to pick the path set within a general setting in order to achieve good results while maintaining a computational feasible problem that scales. An additional limitation of the IP-based approaches is that there is still no efficient algorithm for finding the optimal solution when the objective function or constraints are nonlinear [15].

All these studies were interested only in determining the minimum capacity requirements of a given mesh type network topology with the capacity allocation already given to meet the normal traffic demand, under the failure scenario of single link break. Even though they specified the cost parameter in the objective function, they did not take the distance and link cost metric into account. However, in reality there will always be some networks that have more total spare capacity but are less expensive than some others. Without taking into consideration of the distance and link cost metric, the problem-solving model is not complete and cannot represent the real situation. So far, none of the published researches evaluate the effect of cost and distance metric.

In this thesis, the distance and link cost metric will be taken into account in minimizing the total spare capacity cost for the capacity planning with full restoration for a given working mesh network. Here, an efficient polynomial time complexity heuristic for solving this problem under both link restoration and path restoration schemes is provided.

## 2.1 Mathematical Model of the Spare Capacity Planning

### Problem

First, the problem is defined in the following mathematical model. Given a mesh type network topology, the normal traffic demand, and the capacity allocation to meet the traffic demand, how much spare capacity should be assigned for each link so that the network can tolerate a single link failure? The goal of the optimization is to select the best set of restoration routes under link restoration scheme or path restoration scheme, and assign each route a specific flow to survive a single link failure, such that the least expensive network can be found within a reasonable time.

In this problem, it is assumed that the initial given topology is at least two-connected. The normal traffic demand between each node pair is assumed to be symmetrical and the same route is used for both directions.

#### 2.1.1 Variable Definitions:

The mathematical model for this problem and notation, based on [10] are as follows:

$N$  number of nodes in a network;

$E$  number of links in a network;

$C_j$  cost function of a capacity unit assigned to link  $j$ ;

$\beta_j$  cost of a unit capacity assigned to link  $j$  per unit length;

$L_j$  length of link  $j$ ;

$T$  total number of nonzero traffic demand pairs in the normal traffic demand

matrix;

$T_i$  total number of traffic demand pairs affected by link cut  $i$ ;

$d^t$  normal traffic demand between origin-destination(O-D) pair  $t$ ;

$Q^t$  total number of working routes available to satisfy the normal traffic demand between O-D pair  $t$ ;

$g^{t,q}$  the working flow required on the  $q^{th}$  working route for satisfying the demand between node pair  $t$ ;

$\zeta_j^{t,q}$  takes the value of 1 if the  $q^{th}$  working route for demand pair  $t$  uses link  $j$ ; 0 otherwise,

$w_j$  working capacity on link  $j$ ;

$X_i^t$  normal traffic demand affected by O-D pair  $t$  upon the failure of link  $i$ ;

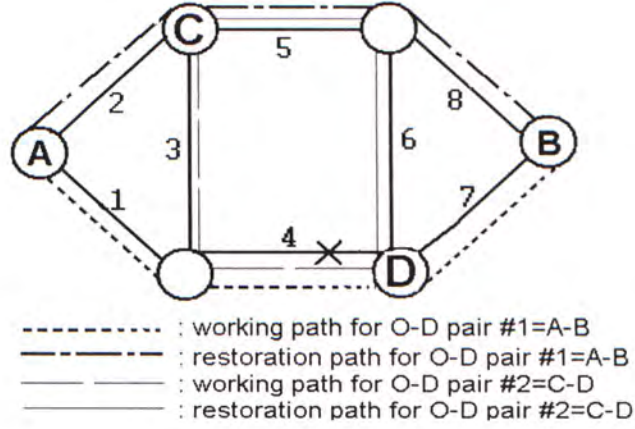
$P_i^t$  total number of possible eligible restoration routes for rerouting the normal traffic demand between O-D pair  $t$  upon the failure of link  $i$ ;

$f_i^{t,p}$  the restoration flow through the  $p^{th}$  restoration route for O-D pair  $t$  upon the failure of link  $i$ ;

$\delta_{i,j}^{t,p}$  takes the value of 1 if the  $p^{th}$  restoration route for O-D pair  $t$  after the failure of link  $i$  uses link  $j$ , and 0 otherwise;

$s_j$  spare capacity on link  $j$ ;

The network shown in Figure 2.3 is used to illustrate the above notations:



**Figure 2.3** An Example Network used to illustrate notations

This network is represented by a graph  $G(N,E) = G(6,8)$ .

The total number of nonzero traffic demand pairs in the normal traffic demand matrix is 2. One is the OD pair #1:A-B. Another one is the OD pair #2:C-D.

Therefore  $T = 2$ .

Traffic demand for both of OD pair #1 and OD pair #2 is 2 units, i.e.  $d^1=2$ ,

$$d^2=2.$$

One working route is available to satisfy the normal traffic demand between OD pair #1 and #2 respectively. They are shown in Figure 2.1, i.e.  $Q^1=1$ ,  $Q^2=1$ .

The working flow required on the 1<sup>st</sup> working route for satisfying the demand between OD pair #1 and OD pair #2 is 2 units respectively, i.e.  $g^{1,1}=2$ ,  $g^{2,1}=2$ .

The 1<sup>st</sup> working route for demand OD pair #1 uses link 1, 4 and 7, so  $\zeta_1^{1,1}=1$ ,

$$\zeta_4^{1,1}=1 \text{ and } \zeta_7^{1,1}=1.$$

The 1<sup>st</sup> working route for demand OD pair #2 uses link 3 and 4, so  $\zeta_3^{1,1}=1$  and

$$\zeta_4^{1,1}=1.$$

In this way, the working capacity is assigned with

$$w_1 = 2, w_3 = 2, w_4 = 2+2=4, w_7 = 2 \text{ and others are } 0.$$

Assume Link 4 is cut then.

Both of traffic demand OD pairs #1 and #2 are affected, so  $T_4 = 2$ .

The normal traffic demand affected by both of OD pair #1 and OD pair #2 after link 4 failure is 2 units, i.e.  $X_4^1 = 2, X_4^2 = 2$ .

Assume that one restoration path is chosen for rerouting the traffic lost, i.e.

$$P_4^1 = 1, P_4^2 = 1.$$

The restoration flow through the 1<sup>st</sup> restoration route for O-D pair #1 and #2 upon the failure of link 4 is 2 units then, i.e.  $f_4^{1,1} = 2, f_4^{2,1} = 2$ .

The 1<sup>st</sup> restoration route for demand OD pair #1 upon link 4 failure uses link 2, 5 and 8, so  $\delta_{4,2}^{1,1} = 1, \delta_{4,5}^{1,1} = 1$  and  $\delta_{4,8}^{1,1} = 1$ .

The 1<sup>st</sup> restoration route for demand OD pair #2 upon link 4 failure uses link 5 and 6, so  $\delta_{4,5}^{2,1} = 1$  and  $\delta_{4,6}^{2,1} = 1$ .

In this way, the spare capacity is assigned with  $s_2 = 2, s_5 = 2+2=4, s_6 = 2, s_8 = 2$  and others are 0.

### 2.1.2 Objective Function and Constraints:

When the problem is transformed to mathematical model, the objective function is:

$$\text{Min} \left\{ \sum_{j=1}^E C_j \cdot s_j \right\} \dots\dots\dots(2.1)$$

where  $C_j = \beta_j \cdot L_j$

For link restoration:

The constraints to be satisfied are:

L1) Restoration flow meets 100% restoration level for each O-D pair  $t$ :

$$\sum_{p=1}^{P_i^t} f_i^{t,p} \geq w_i \quad \forall t = 1, 2, \dots, T_i. \quad \forall i = 1, 2, \dots, E \quad \dots \dots \dots (2.2)$$

L2) Link  $j$ 's spare capacity is sufficient to meet the simultaneous demands of all node pairs affected by any one link failure:

$$\sum_{p=1}^{P_i^t} \delta_{i,j}^{t,p} \cdot f_i^{t,p} \leq s_j \quad \forall t = 1, 2, \dots, T_i. \quad \forall (i, j) = 1, 2, \dots, E, i \neq j \quad \dots \dots \dots (2.3)$$

L3) The flows on restoration paths,  $f_i^{t,p}$ , are non-negative integers.

L4) Spare capacities,  $s_j$ , and working capacities,  $w_j$ , are non-negative integers.

It should be noted that the origin and destination of all working paths cut by a link failure are designated as the immediate end-nodes of the severed link (i.e.,

$$T_i = 1, \text{ and } X_i^t = w_i).$$

For, path restoration:

The constraints to be satisfied are:

P1) restoration flow meets 100% restoration level for each O-D pair  $t$ :

$$\sum_{p=1}^{P_i^t} f_i^{t,p} \geq X_i^t \quad \forall t = 1, 2, \dots, T_i. \quad \forall i = 1, 2, \dots, E \quad \dots \dots \dots (2.4)$$



P2) The total demand lost by OD-pair  $t$  upon the failure of link  $i$  is the sum of the flows over all working routes traversing link  $i$ :

$$\sum_{q=1}^{Q'} \zeta_i^{t,q} \cdot g^{t,q} = X_i^t \quad \forall t = 1, 2, \dots, T_i, \quad \forall i = 1, 2, \dots, E \dots \dots \dots (2.5)$$

P3) Link  $j$ 's spare capacity is sufficient to meet the simultaneous demands of all node pairs affected by any one link failure:

$$\sum_{t=1}^{T_i} \sum_{p=1}^{P'_i} \delta_{i,j}^{t,p} \cdot f_i^{t,p} \leq s_j + \sum_{t=1}^{T_i} \sum_{q=1}^{Q'} \zeta_j^{t,q} \cdot \zeta_i^{t,q} \cdot g^{t,q} \dots \dots \dots (2.6)$$

$$\forall (i, j) = 1, 2, \dots, E, i \neq j$$

P4) The flows on restoration paths,  $f_i^{t,p}$ , and working paths,  $g^{t,q}$ , are non-negative integers.

P5) Spare capacities,  $s_j$ , and working capacities,  $w_j$ , are non-negative integers.

### 2.1.3 Complexity

The problem under study requires  $\sum_{i=1}^E \sum_{t=1}^{T_i} P'_i + 2E$  variables and  $2E$  constraints

for link restoration scheme. For path restoration, the mathematical model consists

of  $\sum_{i=1}^E \sum_{t=1}^{T_i} P'_i + 2E$  variables and  $\sum_{i=1}^E T_i + E$  constraints. The variable  $T_i$  will be

1 for link restoration and scales with the number of nodes of  $O(N^2)$  in the

network. It can be seen that the number of variables and constraints scales with

the number of links and nodes in the network. Furthermore, the number of

variables also scales with the number of restoration routes considered.

Considering that the number of distinct routes in a network of  $E$  links is  $O(2^E)$ .

Finally, the complexity of the whole mathematical model is as follows:

for link restoration, the problem requires  $O(E \cdot 2^E)$  variables and  $O(E)$  constraints; and

for path restoration, the problem requires  $O(E \cdot N^2 \cdot 2^E)$  variables and  $O(E \cdot N^2)$  constraints.

This shows that the problem is NP-hard. In order to solve this NP-hard problem, we have developed an efficient insightful heuristic. The heuristic will be described in the next section.

## **2.2 Greedy Algorithm - Spare Capacity Allocation and Planning Estimator (SCAPE)**

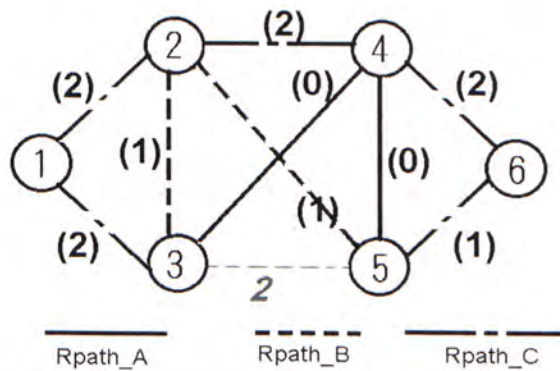
As the spare capacity planning problem is NP-hard, many researchers have attempted different heuristics for solving the problem. Among these heuristics, the most common one is that transforming the spare capacity planning problem to IP formulation and solving with the use of Integer Programming technique, such as Branch and Bound (B & B ) algorithm. Theoretically, in order to obtain the global optimal solution, all distinct restoration routes, between link end-nodes for link restoration or OD pairs for path restoration, must be present in the constraint system of the IP model. However, by doing these, the complexity will be exponential with the total number of link in the network, as shown in the previous subsection.

In order to reduce the complexity and make the integer programming heuristic to be applicable to practical problem, the general approach is to restrict the number of distinct routes entered as constraints with the use of “hop-limited” approach or similar techniques. In this way, it can be expected that Integer programming heuristic no longer ensure the optimality. At the same time, it does not provide any insight for choosing the path set. What makes integer programming worse to apply to solving spare capacity planning problem is that until now, there is no efficient algorithm for solving the nonlinear integer programming problem optimally [15]. That means that the IP heuristic can be effective in optimization only when the objective function and the constraint system of the problem are restricted to be linear function then. However, it can be expected that it is necessary to include some nonlinear variables and constraints, such as QoS

constraints, for the spare capacity planning problem. Therefore, ideally, we need to develop some fairly insightful heuristic which can tackle the above drawbacks while achieving near-optimal or even optimal solution in a reasonable time. With this motivation, we investigate this problem and develop an efficient heuristic to achieve a near-optimal or optimal solution for this NP-hard problem. It is called “Spare Capacity Allocation and Planning Estimator” (SCAPE). In this subsection, the working principle and the implementation of SCAPE will be introduced.

### 2.2.1 Working Principle of SCAPE

In order to understand the basic working principle of the SCAPE, let’s first look at the following simple example. Figure 2.4 shows an example network with 6 nodes and 10 links.



**Figure 2.4** An Example Network for Working Principle of SCAPE.

In this example network, the distance and cost parameters of each link are assumed to be 1 unit. The numbers in the parentheses represent the spare capacity assigned to the link at some time. Assume that for link (3,5), which has a normal traffic demand of 2 units passing through it, is broken. By the link restoration strategy, 2 traffic demand units need to be rerouted through other restoration paths between the end-node pair (3,5). Among these restoration paths, 3 paths are

shown in the figure as example. They are Rpath\_A, Rpath\_B and Rpath\_C respectively. Among these 3 restoration paths, Rpath\_A and Rpath\_B are the shortest one. If we choose any single one of them for rerouting, then either of them should provide the least cost. However, under the condition that some spare capacities already existed on the links, the restoration path set for rerouting will be changed. If Rpath\_A is chosen, 4 spare capacity units will need to be added. For Rpath\_B, 2 spare capacity units need to be added. But for Rpath\_C, just 1 spare capacity units need to be added. Thus, Rpath\_C which is the longest path among these 3 paths turns out to be the least costly route because of the spare capacity that already existed. This example gives us the first key idea of the SCAPE that the placement of new spare capacity on the links to achieve full restorability based on the previous spare capacity placement will lead to the more economical spare capacity planning. We call this the “incremental assignment effect”.

Let’s look at the previous example again. If we use just Rpath\_C for restoration only, 1 more spare unit will need to be added. However, if we split the traffic that need to be rerouted into two portions, each of which is one traffic demand unit. Then if we reroute one unit through Rpath\_C and one unit through Rpath\_B, we can see that no new spare capacity is needed. This fact gives us another key idea that rerouted traffic splitting can help to lower the spare capacity requirement. Since path restoration is similar to link restoration but with more OD pairs to be rerouted each time a link is broken, it can be expected the incremental assignment effect and reroute traffic splitting effect will help both restoration strategies. Based on the above reasoning, we develop the SCAPE to solve the spare capacity

planning problem with the goal of achieving the most economical spare capacity placement in the network.

### 2.2.2 Implementation of SCAPE

Two procedures are used in the heuristic: the **Dijkstra** and **Update** procedure.

Procedure **Dijkstra(G,A,B,n,P)** takes the network structure **G**, and two nodes **A** and **B** of the failed link and the affected traffic **n** as input and then compute out the set of shortest path **P** as well as the flow for these paths as solution. The **Update** procedure will replace the spare capacity in the network for the corresponding link flow through by the set of paths **P**.

The Procedure **Dijkstra(G,A,B,n,P)** in fact is a modified Dijkstra algorithm for our heuristic. As we know that after the first iteration of the algorithm, there will be some spare capacity in some links. For the second iteration of the algorithm, the network has been changed, and those links with spare capacity in the previous failure cases will be a zero cost link in the current iteration. Hence the Dijkstra procedure is based on this principle to find out the shortest path until the spare capacity assigned in the previous cases cannot satisfy the affected traffic in the current iteration. The network structure **G** will be updated. Then a new set of paths will be chosen for assigning the spare capacity of the affect traffic not yet handled in the current iteration. This is the traffic splitting principle and the incremental assignment effect in our heuristic. In the case of paths' length tied, the path will be chosen according to the Dijkstra algorithm. The pseudo code is as follow:

```

Begin
{ G=Load_Netowrk_Information();
  n=All_The_Traffic_Information_For_Each_Link();
  for each failed link case
    { Modified Dijkstra(G,A,B,n,P);
      Update_the_Network(G);
    }
  }
End;

```

It can be seen that as the procedure mainly depends on the Dijkstra algorithm which is of the complexity  $O(N^2)$  where  $N$  is the number of node in the network. Assume that the edge number of the network is  $E$ . The maximum number of shortest paths split for each case is  $k$ , which can be expected to be a small constant. Then the complexity will be  $O(k \cdot E \cdot N^2)$ . This is a very efficient method when compare to the integer programming approach which is of the complexity in exponential form of  $E$  and  $N$ .

### 2.2.3 Improved SCAPE

Although the original SCAPE can achieve a good solution with low complexity, it can be expected that such kind of greedy algorithm cannot achieve the optimal solution in the forward synthesis phase. This is because the interactive effect between the path sets, the spare capacity assignment on each links and the traffic demand has been ignored. The spare capacity planning problem is such a complex combinatorial optimization problem that the interaction between the variables needs to be considered in order to achieve a better result.

To make the SCAPE an optimal or a nearer optimal solution for the spare capacity planning problem, we develop the following improvements.

### **A. Run SCAPE with sorted order**

Based on the traffic splitting and incremental assignment effect, we form the basic SCAPE. However, there must be a starting point for the algorithm to run through. If the algorithm starts with a bad choice, it will lead to a low quality solution. It is important to choose a good starting point that leads to a high quality solution.

As the SCAPE is based on the incremental assignment effect, it is reasonable to expect that the starting point should be the OD pair that lead to the least incremental assignment effect. By this assumption, if the algorithm runs with a sorted incremental assignment effect, SCAPE should lead to a high quality solution. However the incremental assignment effect depends on the total path cost and the traffic demand. But the path we need to choose is not known yet for all OD-pair and if we try to search out all possible paths for all OD-pair and then do sorting, it will be a complex problem. Hence, we try to sort the traffic demand that is already known for all OD pair and then start the SCAPE from the least-load OD-pair. This is the first enhancement.

### **B. After the SCAPE implementation, perform backtracking**

Backtracking has long been a standard method in computer search problems when everything else fails. After backtracking procedure, the solution will be usually improved [13].

The backtracking procedure is very simple in the SCAPE. After getting the solution from the basic SCAPE, reduce the spare capacity unit of each link in the



network and see whether it is still a feasible solution. If yes, then update the network and then use that network for another iteration of backtracking until there is no further improvement. The pseudo code of the backtracking is as follows:

```

do {
    set improvement (imp) to zero
    For each link (A,B) in network (G)
    {
        if spare capacity exists in link (A,B)
        {
            decrement the spare capacity of link (A,B) by one
            if network (G) is still a feasible configuration.....(*)
            {
                increment (imp) by one
                update the network (G)
            }
        }
        else
        { restore spare capacity of link (A,B)
        }
    }
} while (imp) is non-zero

```

Again, for both link and path restoration, the backtracking procedure are the same. From the pseudo code, it can be seen that most of the processing time in the above procedure is consumed in line (\*), the part “check feasibility”. The way for the “check feasibility” part for link restoration is simpler. As it is just a single commodity maximum flow (SCMF) problem which can be solved by the maximal-flow algorithm with complexity  $O(k \cdot E)$  where  $k$  is the maximum flow value and  $E$  is the edge number [14]. However, as the simplicity of the link restoration, we decide to use shortest path for checking feasibility then.

The “check feasibility” part for path restoration is not so simple, because each link failure will cause several OD-pair traffic to be rerouted, and each OD-pair will have several possible paths for supporting its traffic demand. That means we cannot choose any single path of any OD-pair for checking so easily, as we need to consider all the other traffics which are affected by the “check feasibility” procedure. However, this is not an easy task, as this will involve another integer programming problem, the integer multicommodity maximum flow (MCMF) problem. So we try to use the interference heuristic proposed by Iraschko which is an efficient heuristic for solving MCMF problem [11]. This heuristic is of complexity  $O(N^4)$  where  $N$  is the node number. With this heuristic, we solve the “check feasibility” procedure of path restoration quite successfully and make backtracking procedure work for the path restoration scheme. Both the maximal-flow algorithm and interference heuristic are given in Appendix A.

## **2.3 Experimental Results and Discussion**

This section reports experimental results exploring the suitability of the SCAPE for spare capacity planning design. Firstly, we will do an experiment to show that SCAPE we developed can work accurately and optimally for spare capacity planning. Although SCAPE tries to minimize the network cost, many experiments are done to show the result of minimizing the total spare capacity in order to compare the result with those in the literature.

In another set of experiment, we will show that our algorithm can achieve the purpose of minimizing the network cost, which is not shown by other researchers in their own experiments. Finally, from these two sets of experiment, we hope to draw a conclusion that SCAPE is a generic and robust heuristic for spare capacity planning.

### **2.3.1 Experimental Platform**

The SCAPE was written with C++ language and executed on Pentium III 733 MHz PC with 128 MB RAM running Windows 2000 Professional.

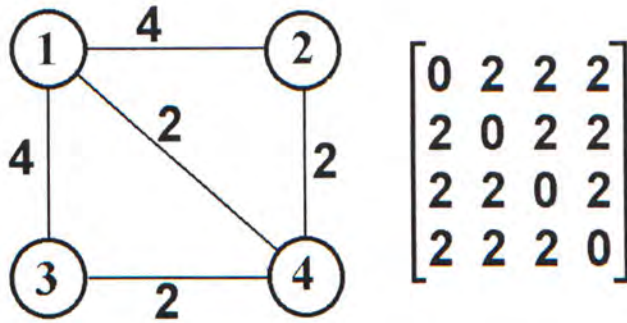
### **2.3.2 Experiment about Accuracy of SCAPE**

In order to test the accuracy and the ability of SCAPE in solving the spare capacity planning problem, a small experiment is designed for this purpose. The experiment performed is as follows:

A 4-node network topology and its traffic demand matrix is shown in Figure 2.5.

The number on the link represents the working capacity obtained with the

shortest path routing scheme for each OD pair traffic. The cost and distance parameters are simply taken to be one. Under a single link failure, with link restoration, all the possible cases of rerouting and flow assignment are show in Table 2.1.



**Figure 2.5**  
A 4-node Network with  
a Uniform Traffic  
Demand Matrix

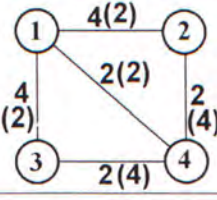
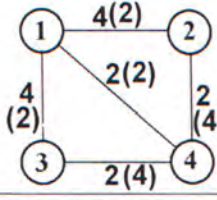
Failed Link	Possible Rerouting route	Possible flow assignment combination				
		A	B	C	D	E
1—2	1→4→2	0	1	2	3	4
	1→3→4→2	4	3	2	1	0
1—3	1→4→3	0	1	2	3	4
	1→2→4→3	4	3	2	1	0
1—4	1→2→4	0	1	2	/	/
	1→3→4	2	1	0	/	/
2—4	2→1→4	0	1	2	/	/
	2→1→3→4	2	1	0	/	/
3—4	3→1→4	0	1	2	/	/
	3→1→2→4	2	1	0	/	/

**Table 2.1** Possible cases of rerouting and flow assignment of the 4-node network.

From Table 2.1, it can be seen that all the possible combination for rerouting and spare capacity assignment with link restoration scheme of this 4-node network under a single link failure scenario is  $(5*5*3*3*3)=675$ .

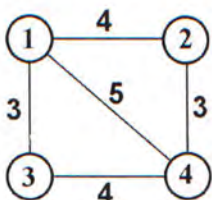
A C program is written to check out all these 675 combinations and then find the

optimal spare capacity assignment with the minimal total spare cost, i.e. the total spare capacity in that case. And then we use SCAPE to find the solution of that network too. The results are shown in the Table 2.2.

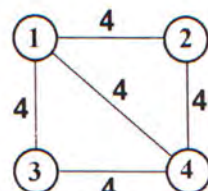
Failed Link	Possible Rerouting route	Flow assignment	
		C program	SCAPE
1 — 2	1 → 4 → 2	2	2
	1 → 3 → 4 → 2	2	2
1 — 3	1 → 4 → 3	2	2
	1 → 2 → 4 → 3	2	2
1 — 4	1 → 2 → 4	2	2
	1 → 3 → 4	0	0
2 — 4	2 → 1 → 4	2	2
	2 → 1 → 3 → 4	0	0
3 — 4	3 → 1 → 4	2	2
	3 → 1 → 2 → 4	0	0
Spare Capacity Assignment (shown in parentheses)			
Optimal Total Spare Capacity		14	14

**Table 2.2** Results of using C program and SCAPE to minimize the total spare capacity of the 4-node network.

Then we use the C program and SCAPE to do the experiment again with distance metric and cost metric as show in Figure 2.6 and Figure 2.7, and try to minimize the total network cost then. The results are shown in Table 2.3.



**Figure 2.6**  
Distance Metric  
of 4-node  
Network



**Figure 2.7**  
Cost Metric  
of 4-node  
Network

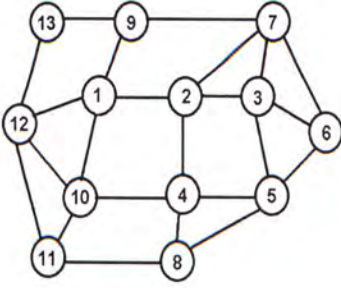
Failed Link	Possible Rerouting route	Flow assignment	
		C program	SCAPE
1 — 2	1→4→2	2	2
	1→3→4→2	2	2
1 — 3	1→4→3	2	2
	1→2→4→3	2	2
1 — 4	1→2→4	2	2
	1→3→4	0	0
2 — 4	2→1→4	2	2
	2→1→3→4	0	0
3 — 4	3→1→4	2	2
	3→1→2→4	0	0
Spare Capacity Assignment (shown in parentheses)			
Optimal Total Spare Cost		208	208

**Table 2.3** Results of using C program and SCAPE to minimize the total spare cost of the 4-node network.

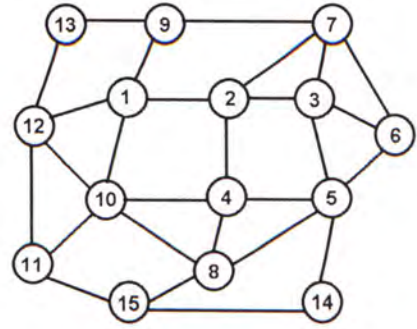
From this experiment, it can show that the SCAPE we developed work accurately and optimally. However, as the network we considered in that case is too small, we do the following two sets of experiments with a larger size of network to further examine the capability of SCAPE.

### 2.3.3 Experiment about Minimization of Network Spare Capacity

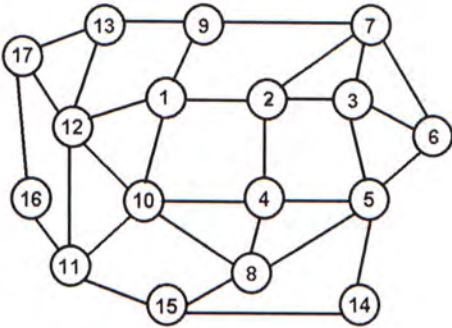
In this set of experiments, simulations will be done for the four medium-sized networks which are proposed by A. Al-Rumaih [12]. The topologies of the four networks are shown in Fig 2.8 - 2.11. In each network, it is assumed that there are two traffic demand units between each node pair in the network. Table 2.4 summarizes the parameters of the four networks. The working capacity for each network was determined using the shortest path routing for each traffic demand



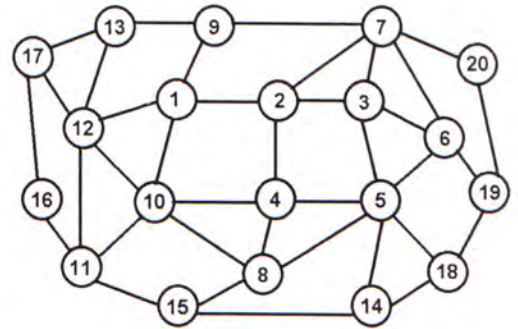
**Figure 2.8** Network 1  
for Experiment 2.3.3



**Figure 2.9** Network 2  
for Experiment 2.3.3



**Figure 2.10** Network 3  
for Experiment 2.3.3



**Figure 2.11** Network 4  
for Experiment 2.3.3

OD-pair, that means  $Q^l = 1$  for all cases in our mathematical model.

Network	Number of nodes	Number of links	Average node degree	Number of O-D pairs	Total network load
1	13	23	3.54	78	156
2	15	27	3.60	105	210
3	17	31	3.65	136	272
4	20	37	3.70	190	380

**Table 2.4** General information of the four test networks for experiment 2.3.3

Al-Rumaih has done the experiments on these four networks with the use of three methods from the literature: (1) the Spare Link Placement Algorithm (SLPA)

[6]; (2) Link restoration using Integer Programming (Link IP) [9]; and (3) Path restoration with link disjoint routes using Integer Programming (Path IP) [10]. All of these methods are used with the objective to minimize the total spare capacity in the network and with hop count limit at 7. The result of simulation for the above four networks by the 3 heuristics just introduced is adopted from Al-Rumaih as he claimed that the implementation of the three methods above is validated by reproducing published results from the literature [12]. We will then use SCAPE with link restoration to compare with the result from method 1, 2 and use SCAPE with path restoration to compare with the result from method 3. For SCAPE, in order to minimize the spare capacity of each network, the link cost parameter and distance parameter for each link in each network is taken to be equal to 1 unit. The results are shown in Table 2.5 and Table 2.6. The running time of the SCAPE for spare capacity planning of these four network is shown in Table 2.7.

For link restoration:

In Table 2, the total spare network capacity for 100% restoration for any single link failure as determined by different algorithms is given.

Network	Working Capacity	Total Spare Capacity				
		SLPA	Link IP	Basic SCAPE	Improved SCAPE w/o backtracking	Improved SCAPE with backtracking
1	324	230	199	241	236	200
2	464	322	264	332	324	268
3	640	444	353	483	430	351
4	966	684	535	756	641	528

**Table 2.5** Total Spare Capacity for Link Restoration



For path restoration:

Network	Working Capacity	Total Spare Capacity		
		Path IP	Basic SCAPE	Improved SCAPE with backtracking
1	324	135	160	128
2	464	176	208	170
3	640	236	281	249
4	966	350	437	367

**Table 2.6** Total Spare Capacity for Path Restoration

Network	Running time of Link Restoration (ms)		Running time of Path Restoration (ms)	
	Basic SCAPE	Improved SCAPE	Basic SCAPE	Improved SCAPE
1	20	110	100	2054
2	40	160	170	3635
3	50	260	271	6440
4	60	461	561	25296

**Table 2.7** The running time of the SCAPE

From the result shown in Table 2.5, Table 2.6 and Table 2.7, we observe the following:

1. When we compare the result obtained in Basic SCAPE and Improved SCAPE, we see that Improved SCAPE can outperform Basic SCAPE from 15% to 30% in all the cases. It provides the strong base that the incremental assignment sorting and backtracking procedure can bring a high quality solution with great improvement.
2. From the running time table, we can see that, the backtracking procedure is about 5 times as complex as the basic SCAPE. It can be seen that even with the backtracking procedure, the running time is very short for a 20-node network, that means the SCAPE is overall an algorithm with low complexity.
3. For link restoration strategy, when compared to the SLPA results, it can be

seen that even the Improved SCAPE without any backtracking procedure performed, can bring up to 6% improvement. It should be noted that the SLPA is based on the cut-based procedure, which cannot provide any routing and flow information while SCAPE can. And SLPA is itself a forward synthesis and backward tightening heuristic which will be more complex than Improved SCAPE without backtracking.

4. When compared with Link IP method for link restoration strategy, if with the backtracking procedure, SCAPE can perform within 1.5% of the result obtained for Link IP. However, we can find that for the network size to become larger and larger, SCAPE can even outperform the Link IP up to 1.3% for the largest network. It is because in order to reduce the complexity and running time of the Link IP procedure, there is a limit on the size of the restoration paths with hop limit, and some long paths which may be helpful in the optimization process are excluded. While for SCAPE, it tries to assign the flow and the routing information based on the previous iteration until all the traffic demand is routed or rerouted, so the problem in limiting the path size in the execution does not exist. Hence, SCAPE can cover a large search space and produce a better result for larger networks.
5. For path restoration strategy, when compared to the Path IP results, the SCAPE still can perform well, within 5.5% of the result obtained by Path IP. In the smaller network, SCAPE performs better. For the smallest network, SCAPE even outperform the Path IP by 5%. It is because for the path restoration strategy, a single link failure will cause several OD pair traffic demand to be rerouted, that means more interaction effort needs to be considered between these OD pair traffic in order to achieve the optimal

solution, which is not the case for link restoration. So path size was not a dominant factor then. In this way, Path IP will perform better than SCAPE, because SCAPE is not so strong in handling the interactive effect between the assignment. However, SCAPE can provide the insight of choosing the Path Set and with further software development, it can achieve real-time processing with good result.

Through the experiment results, it can be concluded that the principles of traffic splitting effect and incremental assignment effect work and form the basic SCAPE. With the sorting of incremental assignment effect and a simple backtracking procedure, improved SCAPE can outperform the SLPA and have a similar performance with integer programming but with a much lower complexity. What SCAPE contribute is the insight for choosing the path set while providing the flow and routing information directly. This is still an open question and cannot be obtained by the integer programming approach. The SCAPE is also useful in developing real time processing software for spare capacity planning.

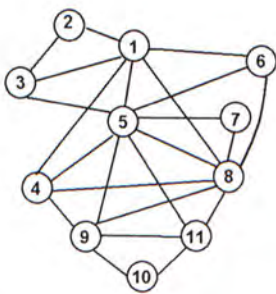
#### **2.3.4 Experiment about Minimization of Network Spare Cost**

As said before, the SCAPE is developed to optimize the selection of the best routes according to the dimensioning in order to find the least cost network within a reasonable time. In the first set of experiments, it has been shown that SCAPE can achieve optimization of a single dimension, i.e. the spare capacity. Here we show that SCAPE can also handle the distance and cost metrics and we perform the second set of experiments to test the SCAPE performance.

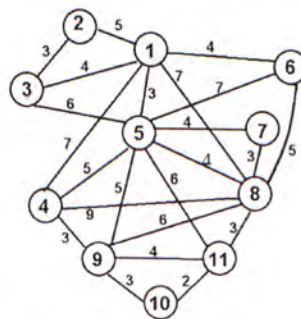
Unlike the first set of experiment, until now and to the best of our knowledge, no research work exists that handles these two metrics and produce a guideline for reference even though the researchers try to include the cost parameter in the objective function. Therefore, we develop our own experiments and guidelines to show that SCAPE can be used for cost minimization as well as the spare capacity assignment.

The experiment is simple. We apply a set of distance and cost metrics to the same network topologies and then use the SCAPE heuristic to find the capacity assignment and then compare with the assignment based on the other heuristic to see whether the SCAPE heuristic can lead to a more economical result.

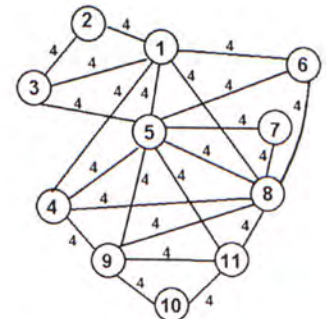
The network we use for the second set of experiment is from Hasegawa [9]. The network topology is shown in Figure 2.12. We assign a distance metric that scales with the drawing distance of the network and a cost metric to the network. They are shown in Figure 2.13 and Figure 2.14 respectively. The spare capacity assignment (S.C.A.) by Herzberg [7], Grover [6] and SCAPE are shown in Figure 2.15, Figure 2.16 and Figure 2.17. Table 2.8 gives the total cost for the spare capacity planning by these different sets of assignment.



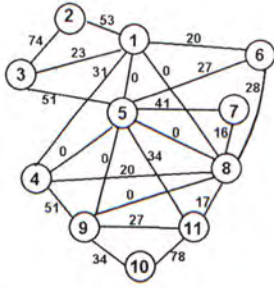
**Figure 2.12** Topology of Hasegawa Network



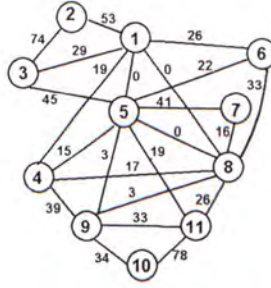
**Figure 2.13** Distance Metric of Hasegawa Network



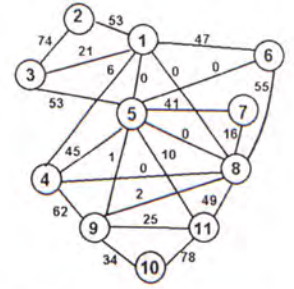
**Figure 2.14** Cost Metric of Hasegawa Network



**Figure 2.15** S.C.A by Herzberg



**Figure 2.16** S.C.A by Grover



**Figure 2.15** S.C.A by SCAPE

	Herzberg	Grover	SCAPE
Total Spare Capacity	625	625	672
Total Spare Cost	10708	10404	10396

**Table 2.8** Result of Experiment 2.3.4

From the results shown in Table 2.8, we can find that even though the total spare capacity found by SCAPE is more than those found by the Herzberg and Grover, the total cost is less than either of them. This shows that SCAPE is better than other heuristics in the way that it can handle the cost metric and distance metric successfully for obtaining the lowest cost network. Based on the adaptive property during the execution of SCAPE, it can be expected that it can handle an adaptive cost function too. This is better than Link IP and Path IP which need to keep the cost parameter constant during the iteration to ensure linearity due to their inability to handle nonlinear integer programming problem.

## **2.4 Conclusions**

It can be seen that SCAPE is a generic and robust heuristic for the spare capacity planning with both link restoration and path restoration strategies. Not only can it achieve the optimization of the total spare capacity, it also minimizes the total cost in a reasonable time. From the experimental results, it can be seen that the SCAPE performance is almost as good as all existing heuristics in the literature while SCAPE contributes to the insight of choosing the path set as well as its capability in handling nonlinear cost function based on its adaptive property.

## Chapter 3 Survivable All-Optical Network Design

### Problem

All optical networks will play a key role in the future worldwide telecommunication infrastructure due to the advances in fiber optic technology and the cost reduction of the corresponding optical components. The high capacity of fiber enables each link to carry large amounts of traffic that make various broadband services available. However, it also means that service loss due to link failures is more probable and severe. A fully survivable fiber optic communication network is definitely needed.

The most probable failure is a single link failure. To make a network fully survivable, the network must be at least two-connected (or  $N$ -connected) and there must also be enough spare capacity to guarantee the level of survivability. However, if the design is not done well, it will waste a lot of resources. Therefore, the main objective of this chapter is to design a cost-effective survivable all-optical communication network against a single link failure.

In Chapter 2, it was assumed that a two-connected network existed and we concentrated on the spare capacity allocation and planning problem. That means only part of the network design problem was tackled. In this chapter, we will tackle a more general problem - the design of the least cost survivable all optical mesh network with arbitrary topology where only the traffic demand matrix is given. For the generalized problem, there will be more conflicting, inter-dependent constraints that need to be met, such as the connectivity

constraints, working capacity constraints, routing consideration and topology arrangement etc. These additional constraints increase the design complexity, thus requiring multi-constraint combinatorial optimization. That is, the network topology, the routing assignment and the link capacity assignment need to be optimized together. No doubt that such a problem is NP-complete, and we must use heuristics to handle the complexity and solve practical problems.

Many heuristics have been proposed for designing survivable fiber optic communication network of both SONET [23,24] and mesh network. Groetschel et.al. employed a cutting plane method to obtain the solutions [16]. Koh and Lee proposed a tabu search method for optimization and generating an effective solution [17]. However, all the above studies just concentrated on the connectivity constraint without considering the routing and capacity assignment, which are also important factors when designing an optimal and cost-effective network.

Recently, Van Caenegem et.al. [3] provided a method for designing a survivable WDM network and considering the connectivity constraints, routing and capacity planning together. A branch and bound (B & B) procedure was provided and a heuristic based on simulated annealing (SA) was proposed for solving the problem. However, the spare capacity planning problem was not considered in the design process. From our point of view, the spare capacity planning should be considered in the whole design. Without it, the design is incomplete and the result obtained may not be the most cost-effective.



Based on the above reasons, we try to map the problem into a mathematical model that jointly consider the topology design, routing, capacity assignment, and rerouting and spare capacity assignment under a single fault situation and develop a number of heuristics to find the lowest cost network in a reasonable time. In fact, there are many possible heuristics proposed for data networks. Inspired by these methods, we develop two heuristics here, namely, the modified drop algorithm and genetic algorithm.

The Drop Algorithm [20] is a simple and rather standard design method for data networks. We modify the algorithm to handle the constraints in our problem and develop a suitable and efficient heuristic for solving the survivable fiber optic network design problem.

The Genetic Algorithm is a global optimization technique that has attracted much attentions in recent years. Due to its ability in handling the multi-constraint problem, nonlinear function and discrete value, it appears to be a very suitable method for solving the network design problem. Many researchers have proposed to apply genetic algorithms for solving the design problem in data network and even in ATM ring network [18,19]. Here we apply the genetic algorithm to design survivable fiber optic mesh network and show that it can solve the design problem quite well.

## 3.1 Mathematical Model of the Survivable Network

### Design Problem

First, we define the problem in a mathematical model. We consider the problem of designing a survivable fiber optic communication network. In this problem, the geographical information (the location of nodes), the predicted traffic requirement matrix as well as a cost structure which is dependent on the distance and capacity among the nodes are given. The goal of optimization is to select the overall best set of routes and restoration routes, with proper dimensioning, to result in the least cost survivable network that is at least two-connected, and which can be found within a reasonable computation time.

In this problem, the normal traffic demand between each node pair is assumed to be symmetrical and both directions follow the same route.

This problem basically follows the same mathematical model presented in the last chapter but with a different objective function. In addition, some variables and constraints are added for this design problem. The added variables are as follows:

$\alpha_j$  fixed installation cost per unit length assigned to link  $j$ ;

$\delta_j$  takes the value of 1 if link  $j$  is included in the network, and 0 otherwise;

$\delta_{j,n}$  takes the value of 1 if link  $j$  is incident to node  $n$ , and 0 otherwise.

The objective function becomes:

$$\text{Min} \left\{ \sum_{j=1}^E \alpha_j \cdot \delta_j \cdot L_j + C_j \cdot (w_j + s_j) \right\} \dots\dots\dots(3.1)$$

where  $C_j = \beta_j \cdot L_j$

The following constraints need to be satisfied:

A1) the total capacity on the working routes allocated to any OD traffic pair  $t$  can carry all the demand of the OD pair  $t$ :

$$\sum_{q=1}^{Q^t} g^{t,q} = d^t, \quad \forall t = 1, 2, \dots, T; \dots\dots\dots(3.2)$$

A2) link  $j$ 's working capacity is sufficient to meet the pre-failure demands of all OD pairs which cross it:

$$\sum_{t=1}^T \sum_{q=1}^{Q^t} \zeta_j^{t,q} \cdot g^{t,q} = w_j, \quad \forall j = 1, 2, \dots, E; \dots\dots\dots(3.3)$$

A3) the node degree must be at least two to assure recovery from any single link failure:

$$\sum_{j=1}^E \delta_{j,n} \geq 2, \quad \forall n = 1, 2, \dots, N \dots\dots\dots(3.4)$$

As before, the formulation can be adapted to a link-restorable network by adding the constraints L1-L4 and by designating the origin and destination of all working paths cut by a link failure as the immediate end-nodes of the severed link (i.e.,  $T_i = 1$ , and  $X_i' = w_i$ ). For path-restorable network, the constraints P1-P5 should be added to form the complete model.

## 3.2 Optimization Algorithms for Survivable Network

### Design Problem

The problem under study requires  $\sum_{i=1}^E \sum_{t=1}^{T_i} P_i' + 2E + T \cdot \sum_{q=1}^{Q'} g^{t,q} + N \cdot E$  variables

and  $3E + N$  constraints for the link restoration scheme. For path restoration, the

mathematical model consists of  $\sum_{i=1}^E \sum_{t=1}^{T_i} P_i' + 2E + T \cdot \sum_{q=1}^{Q'} g^{t,q} + N \cdot E$  variables and

$\sum_{i=1}^E T_i + 2E + N$  constraints. The variable  $T_i$  and will be 1 for link restoration.

$T$  and  $T_i$  scale with the number of nodes, i.e.,  $O(N^2)$  in the network. It can be

seen that the number of variables and constraints scales with the number of links

and nodes in the network. Furthermore, the number of variables also scales with

the number of working routes and restoration routes considered. The number of

distinct routes in a network of  $E$  links is  $O(2^E)$ . That means the variable  $P_i'$

and  $Q'$  scale with  $O(2^E)$  in this problem.

Finally, the complexity of the whole mathematical model is as follows:

For link restoration, the problem requires  $O(E \cdot 2^E + N^2 \cdot 2^E)$  variables and

$O(E + N)$  constraints.

For path restoration, the problem requires  $O(E \cdot N^2 \cdot 2^E + N^2 \cdot 2^E)$  variables and

$O(E \cdot N^2)$  constraints.

Since the problem is NP-complete, we develop two more heuristic techniques for

SCAPE to handle the complexity and solve the practical problems. They are the

Modified Drop Algorithm and the Genetic Algorithm.

## **3.2.1 Modified Drop Algorithm (MDA)**

### **3.2.1.1 Drop Algorithm Introduction**

The Drop Algorithm is a low complexity algorithm for designing cost-effective mesh network. Starting with a fully connected network, all links are marked “deletable”. Then the “deletable” link with the highest cost is chosen and temporarily deleted. The capacity then is reassigned by redistributing the traffic. If the resulting network is improved, then the link will be actually deleted. If not, the link will be marked “undeletable” and will become a component of the final network. This process will be repeated until all the links are either marked undeletable or deleted [20].

### **3.2.1.2 Network Design with MDA**

From Section 3.2.1.1, it can be seen that the drop algorithm cannot be directly applied to solve our problem. Therefore, we modify the drop algorithm and develop it as an efficient method for building a survivable network.

As the design problem we are handling is very big and complex, the general approach is to break the big problem into several small sub-problems for insightful optimization respectively and then recombine the results together to get the overall result. This can solve the problem efficiently but the trade off is a less than optimal solution. Our approach here is based almost on the same principle by decomposing the design problem into three sub-problems: a) topology design, b) routing with working capacity planning, and c) rerouting with spare capacity planning. The main difference is that after dealing with the three sub-problems, we recombine the solutions to get the total network cost and use the total cost for

optimization. This approach intuitively takes advantage of the efficient handling of small sub-problems to result in an effective joint optimization

The modified drop algorithm is as follows:

First, we tackle the network topology problem. We start by assuming that the network is a fully connected mesh network. Then we repeat to choose the most expensive link by the objective function  $\alpha_j \cdot \delta_j \cdot L_j + C_j \cdot (w_j + s_j)$  and determine whether a link is deleted according to the cost reduction resulted provided that the constraints of connectivity is not violated.

In order to reduce the complexity of the problem, we assume that the traffic demand between each OD pair is routed by one single path. We do the routing and working capacity planning using the single shortest path. That means the working capacity planning is performed by the shortest path routing algorithm. Finally after the topology is formed and the working capacity of the whole network is planned by choosing the shortest path for each OD traffic demand pair, we use the efficient SCAPE to solve the spare capacity planning problem. In this way, the topology, the working and spare capacity plan of the whole network can be found. The whole algorithm will be repeated until the lowest cost network is achieved. The following is the pseudo code :

- Step 1:** Create a fully connected network.
- Step 2:** All the links are marked “deletable”
- Step 3:** Initialize the algorithm by performing the working capacity planning with the shortest path routing once and then the spare capacity planning

with SCAPE.

- Step 4:** Get the total cost and mark it “**old\_cost**”.
- Step 5:** Temporarily delete the most expensive “**deletable**” link with cost function  $\alpha_j \cdot \delta_j \cdot L_j + C_j \cdot (w_j + s_j)$ .
- Step 6:** If the connectivity constraint is violated, recover the network, mark the link “**undeletable**” and then go to **Step 5**, else go to **Step 7**.
- Step 7:** Do the working capacity planning with the shortest path routing and spare capacity planning with SCAPE again.
- Step 8:** Get the total cost and marked as “**new\_cost**”.
- Step 9:** If “**new\_cost**” is less than “**old\_cost**”, then update the network and update the “**old\_cost**” with “**new\_cost**”, else recover the network and mark the link as “**undeletable**”.
- Step 10:** If all the links are either marked as “**undeletable**” or deleted, calculate the total improvement and marked as “**imp**”, then go to **Step 11**, else go back to **Step 5**.
- Step 11:** Repeat **Step 2** to **Step 10** until no improvement can be made (i.e. “**imp**” is equal to zero).

## 3.2.2 Genetic Algorithm (GA)

### 3.2.2.1 Genetic Algorithm Introduction

The Genetic Algorithm (GA) is a simple global optimization technique for complex multi-dimensional search spaces. GA has attracted growing interest in recent years for solving complex problems in many fields, such as resources allocation, scheduling and telecommunication [22]. GA is a stochastic search techniques that mimic the survival of the fittest (or best) paradigm observed in

nature [21]. In GA, each possible solution in the form of a set of parameters is treated as an individual which is usually encoded as a string of binary or real numbers. A set of individuals forms a population. GA will implement the genetic operators, which are selection, crossover and mutation, to the parent population to form offspring for the next generation. GA then evolve successive generations of a population in a manner such that only the best solutions survive from generation to generation while the initial population can be generated randomly or by some other efficient heuristics. The evaluation of the solution quality is done by a fitness function, which is generally the objective function of the problem. GA will repeat the process of evolution until they reach a desired termination criterion, such as the number of iteration. In fact GA is problem dependant. The solution encoding method and the fitness function will depend on the problem and which GA operators are suitable are also problem dependant. How to solve the problem with GA, and what the operators and parameters should be will be the main area to be investigated next.

#### **3.2.2.2 Network Design with GA**

Before we can apply GA to our problem, we must find an efficient method to encode the solution in an appropriate form. We again take the divide-and-conquer approach to break the big problem into three smaller sub-problems for insightful optimization and then recombine the result to get the overall result with some some minor changes. The GA design approach is as follows:

We start with a fully connected network, assuming that the fixed node locations and the traffic demand between the node pairs are given. We assume only one



routing path is used for the working capacity planning. With this simplification, we can concentrate on handling the routing and working capacity planning sub-problem first. We then use GA for optimization of this sub-problem. The possible eligible routes for routing the normal traffic demand between the node pairs are found and stored with an index assigned to the routes for each node pairs. Then we encode the individual  $R = \{r_1, r_2, \dots, r_{n(n-1)/2}\}$  where  $r_x$  represents the index of the working route used for the traffic requirement between the node pair  $x$ . The working capacity for each link will then be assigned from the routing scheme represented by the individual  $R$ . The routing and working capacity planning sub-problem can then be locally optimized.

After handling the routing and working capacity planning problem, we can use the previous developed SCAPE for handling the rerouting and spare capacity planning problem efficiently to obtain a high quality solution. However, if SCAPE is used for each individual during each generation, the running time for converging to the optimal solution will be very long. Based on the property that poor solutions appear in the beginning of the genetic algorithm, we can selectively apply SCAPE to find the overall optimal solution only in the last few steps of the iteration. Intuitively this can reduce the convergence time while maintain the joint optimization concept.

Finally, depending on whether there are working capacity or spare capacity on a link, as well as whether the connectivity constraint will be violated, we can decide whether to eliminate the link or not. After the elimination, the final topology will be formed. After all the three sub-problems are solved, we obtain

the total network cost by calculating the fixed cost, the working capacity cost and the spare capacity cost. The total network cost will then be used for evaluating the fitness to get the best solution in this generation.

In our GA, we will generate the initial population of individuals randomly. Then at each generation, the population will undergo tournament selection, uniform recombination and real value mutation to form a new set of offsprings for the next generation. After the fitness evaluation and the elitist reproduction scheme, a new population of the next generation will be formed from the parent population and offspring population. The GA and corresponding procedure will be repeated until a specific number of iterations has been reached. The pseudo code is as follows:

- Step 1:** Create a fully connected network.
- Step 2:** Find the  $k$ -shortest paths for routing traffic between all the OD pairs and index the paths for each OD pair.
- Step 3:** Generate an initial population of  $M$  individuals randomly, where each individual is  $\mathbf{R}=\{r_1, r_2, \dots, r_{n(n-1)/2}\}$  where  $r_x$  represents the index of the working route used for the traffic requirement between the node pair  $x$ .
- Step 4:** Do the genetic operator “selection” with the binary tournament selection scheme and select out 2 parents.
- Step 5:** Do the genetic operator “crossover” with the uniform recombination scheme for the 2 parents and form 2 offspring.
- Step 6:** Do the genetic operator “mutation” with the random mutation scheme for  $m\_rate$  portion of the whole population.

**Step 7:** If the number of generations reaches a specific level  $s$ , apply SCAPE to find the spare capacity planning for the network formed.

**Step 8:** Under the constraint of two-connectedness, find the total cost of the network formed by each individual with the objective function

$$\sum_{j=1}^E \alpha_j \cdot \delta_j \cdot L_j + C_j \cdot (w_j + s_j).$$

**Step 9:** Repeat Step 4 to Step 8 until a specific number of the generations  $N$  is reached.

### 3.2.3 Complexity of MDA and GA

For both MDA and GA, the main processing time is used by SCAPE which is of complexity in the order of  $O(N^3)$  for link restoration and of complexity in the order of  $O(N^4)$  for path restoration. The overall complexity of MDA and GA depends on the number of iteration then. For MDA, it depends on the total edge number which is of order  $O(N^2)$ . For GA, the iteration depends on the number of generation and the population size. We do believe that they should be with order of complexity of  $O(N^2)$  respectively in order to achieve a high quality solution with GA. Then the number of iteration of our GA will be of complexity of  $O(N^2) \cdot O(N^2) = O(N^4)$ .

### **3.3 Experimental Results and Discussion**

The experimental results comparing the Modified Drop Algorithm and the Genetic Algorithm for the whole network design problem is shown here. First, we perform an experiment to show the accuracy of the MDA and GA in solving the network design problem. Then a second experiment is performed to demonstrate the concept of joint optimization of the rerouting and spare capacity planning. Finally a set of experiments is performed to compare the performance of MDA and GA. We then draw some insightful conclusion through these sets of experiments.

#### **3.3.1 Experimental Platform**

Both of the MDA and GA were written with C++ language and executed on a Pentium III 733 MHz PC with 128 MB RAM running Windows 2000 Professional.

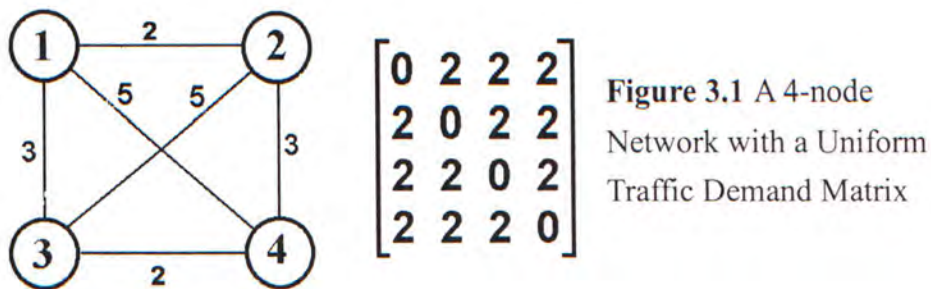
#### **3.3.2 Experiment about Accuracy of MDA and GA**

In order to test the accuracy and the ability of MDA and GA in solving the network design problem, a small experiment is designed for this purpose. The experiment done is as follows:

A 4-node network with an arbitrary topology and traffic demand matrix is shown in Figure 3.1. The number on the link represents the distance. The fixed cost  $\alpha$  for each link is assumed to be 50 units and the cost for adding a unit of capacity  $\beta$  for each link is assumed to be 4 units. The design is performed for single link failure protection with link restoration. We assume that only one working path is

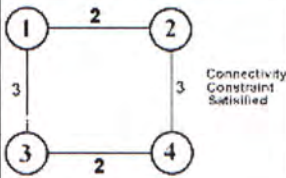
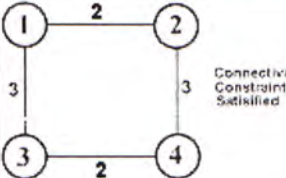
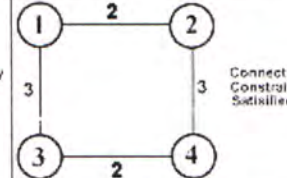
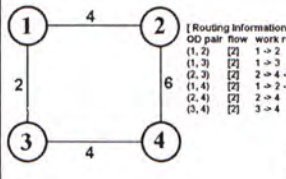
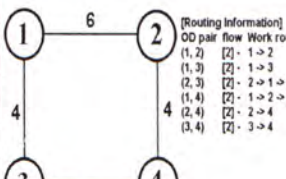
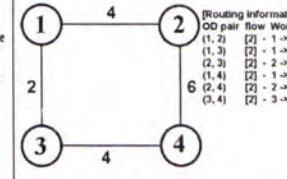
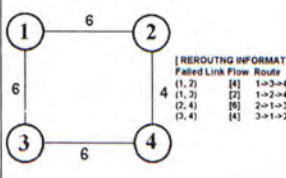
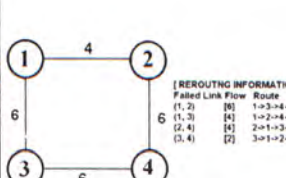
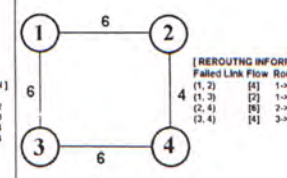
used for routing the traffic of each OD-pair and SCAPE is used for the spare capacity planning under a single-fault scenario.

For this 4-node network, all the possible working paths for routing the normal traffic between each OD-pair is shown in Table 3.1. The number of possible combinations of working capacity planning is  $5^6 = 15625$ . In this way, we can write a C program to simulate all of these combinations and then use SCAPE to find the spare capacity planning for each combination. Finally, we can find out the total network cost of all possible 4-node survivable networks formed and then get the lowest cost network. We can then compare the results found from the MDA and GA. The results found by the C program, MDA and GA are shown in Table 3.2.



OD-pair	Working Path 1	Working Path 2	Working Path 3	Working Path 4	Working Path 5
1 — 2	1→2	1→3→2	1→4→2	1→3→4→2	1→4→3→2
1 — 3	1→3	1→2→3	1→4→3	1→2→4→3	1→4→2→3
1 — 4	1→4	1→2→4	1→3→4	1→2→3→4	1→3→2→4
2 — 3	2→3	2→1→3	2→4→3	2→1→4→3	2→4→1→3
2 — 4	2→4	2→1→4	2→3→4	2→1→3→4	2→3→1→4
3 — 4	3→4	3→1→4	3→2→4	3→1→2→4	3→2→1→4

**Table 3.1** All possible combination of working routes for a 4-node network.

	Design by C program	Design by MDA	Design by GA
Topology formed			
Total topology cost	500	500	500
Routing and Working Capacity Planning			
Total working capacity cost	80	80	80
Rerouting and Spare Capacity Planning			
Total Spare Capacity cost	108	112	108
Total network cost	688	692	688

**Table 3.2** Result and comparison of Experiment 3.3.2.

From this result, it can be seen that both MDA and GA work successfully and accurately in designing a small survivable network. MDA does not perform as well as GA because MDA only uses the shortest path for working capacity planning, so the result may not be as good. We perform two more experiments

below to show that our designing principle is right and MDA and GA are excellent in solving the design problem of survivable network with a reasonable size.

### **3.3.3 Experiment about Principle of Survivable Network Design**

As we have mentioned, in designing a survivable network, the topology design, the working capacity planning and the spare capacity planning should be jointly considered together. However, up to the best of our knowledge, there is no published work for such an approach. At the same time, the problem of joint optimization is so big and complex that it is very difficult to handle. Here we perform an experiment to show that the principle we developed before is appropriate for designing survivable networks under joint optimization and we show that the design can be handled with a low complexity for a reasonable size network.

First, we assume that the geographical information, the cost parameters  $\alpha$  and  $\beta$ , as well as the traffic demand matrix for a 10-node network are given. All these information is shown in Figure 3.2 to Figure 3.5. The lowest cost survivable network is designed under the conditions of one working path for routing the normal traffic demand and single fault-tolerance. We consider only link restoration as path restoration can also be solved by the same approach but its higher complexity is not necessary for demonstrating the concept.

After we have obtained a design of the network with the optimized topology and working capacity, we can perform SCAPE for this optimized network to get the

final network. We use this approach to jointly consider the routing, working capacity planning and the spare capacity planning. Finally, we show a comparison to demonstrate that this point of view is appropriate in designing a survivable network. The results and the running time are shown in Table 3.3.

0	2	3	2	5	2	4	3	3	2
2	0	2	4	2	1	1	2	2	6
3	2	0	2	2	3	2	2	4	2
2	4	2	0	2	3	2	3	2	5
5	2	2	2	0	1	2	4	4	2
2	1	3	3	1	0	2	3	2	1
4	1	2	2	2	2	0	2	4	3
3	2	2	3	4	3	2	0	2	2
3	2	4	2	4	2	4	2	0	5
2	6	2	5	2	1	2	2	5	0

**Figure 3.2** Distance Matrix  
for Experiment 3.3.3

0	4	4	4	4	4	4	4	4	4
4	0	4	4	4	4	4	4	4	4
4	4	0	4	4	4	4	4	4	4
4	4	4	0	4	4	4	4	4	4
4	4	4	4	0	4	4	4	4	4
4	4	4	4	4	0	4	4	4	4
4	4	4	4	4	4	0	4	4	4
4	4	4	4	4	4	4	0	4	4
4	4	4	4	4	4	4	4	0	4
4	4	4	4	4	4	4	4	4	0

**Figure 3.3**  $\beta_{ij}$  Matrix  
for Experiment 3.3.3

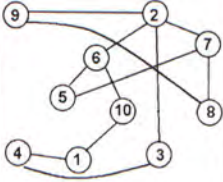
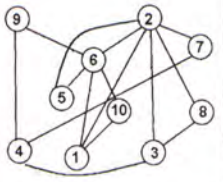
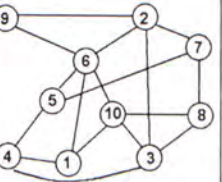
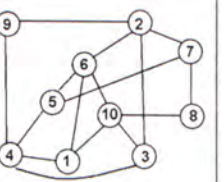
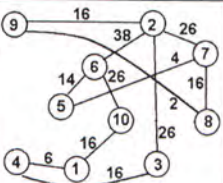
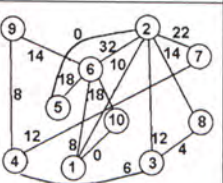
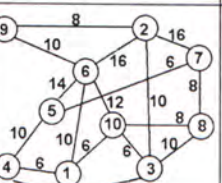
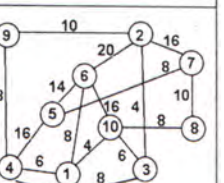
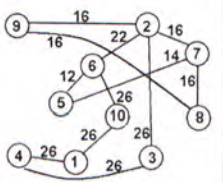
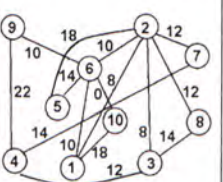
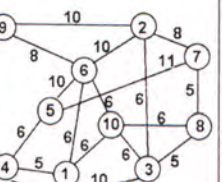
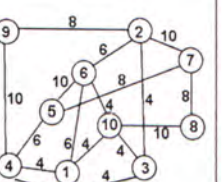
0	50	50	50	50	50	50	50	50	50
50	0	50	50	50	50	50	50	50	50
50	50	0	50	50	50	50	50	50	50
50	50	50	0	50	50	50	50	50	50
50	50	50	50	0	50	50	50	50	50
50	50	50	50	50	0	50	50	50	50
50	50	50	50	50	50	0	50	50	50
50	50	50	50	50	50	50	0	50	50
50	50	50	50	50	50	50	50	0	50
50	50	50	50	50	50	50	50	50	0

**Figure 3.4**  $\alpha_{ij}$  Matrix  
for Experiment 3.3.3

0	2	2	2	2	2	2	2	2	2
2	0	2	2	2	2	2	2	2	2
2	2	0	2	2	2	2	2	2	2
2	2	2	0	2	2	2	2	2	2
2	2	2	2	0	2	2	2	2	2
2	2	2	2	2	0	2	2	2	2
2	2	2	2	2	2	0	2	2	2
2	2	2	2	2	2	2	0	2	2
2	2	2	2	2	2	2	2	0	2
2	2	2	2	2	2	2	2	2	0

**Figure 3.5** Traffic Matrix  
for Experiment 3.3.3



	Without joint optimization		With joint optimization	
	MDA	GA	MDA	GA
Topology formed				
Total topology cost	1000	1300	1400	1400
Working Capacity Planning				
Total working capacity cost	1232	1064	1016	1032
Spare Capacity Planning				
Total Spare Capacity cost	1632	1312	808	728
Running time (ms)	691	61338	1011	243269
Total network cost	3864	3676	3224	3160

**Table 3.3** Comparison of network costs with or without joint optimization.

From the results shown in Table 3.3, we observe that no matter which method (MDA or GA) was used, when jointly considering the spare capacity planning during the design process, either method can result in a solution that is 14% to

16% better than the ones obtained by separately considering the topology, working capacity planning and spare capacity planning. Based on this, we arrive at the following insight. If one only considers the topology design and working capacity planning, the design will try to restrict the topology and working capacity planning to as low a cost as possible. This will limit the possible choices of routes for restoration and cause the spare capacity planning to concentrate on only a few links that may result in higher spare cost. From Table 3.3, both cases cause almost a double increase in spare capacity cost which result in a more expensive survivable network overall. Hence, it is necessary to jointly optimize the topology design, working capacity planning and spare capacity planning altogether during the design process. From the table, even for a 10-node network, the running time for MDA is only 1 second and 4 minutes for GA. It shows that both MDA and GA can achieve the joint optimization design objective in a reasonable time.

### **3.3.4 Experiment about Performance of MDA and GA**

After showing the design principle and capability of MDA and GA in solving the survivable design problem, we now perform another experiment to evaluate the performance of MDA and GA and get some insight on the problem of survivable network design.

We use the same 10-node network and the geographical information, as well as the traffic demand matrix given in the last experiment. We vary the cost parameters  $\alpha$  and  $\beta$  to compare the performance of MDA and GA, as well as the effect of the cost parameters during the design process. The result of this

experiment is shown in Table 3.4 and the average results are shown in Table 3.5.

$\alpha$	$\beta$	Link Restoration		Path Restoration	
		MDA	GA	MDA	GA
4	4	1812	1392	1360	1296
4	8	2912	2524	2472	2368
4	20	6944	5856	5836	5520
4	40	13664	11364	11376	10820
4	200	67424	55896	55696	52852
8	4	2344	1616	1576	1508
20	4	3056	2184	2088	1988
40	4	3496	2936	2712	2644
200	4	7064	7288	6648	6928

**Table 3.4** Comparison of MDA and GA with respect to various cost parameters

	Link Restoration		Path Restoration	
	Average cost	Average running time (ms)	Average cost	Average running time (ms)
MDA	12079	1056	9973	1846
GA	10117	276805	9547	602319

**Table 3.5** The average result and running time for all the cases in Table 3.4.

From the results shown in Table 3.4 and 3.5, we observe the following:

1. GA performs better than MDA in almost all cases. The main reason is that MDA only considers the shortest path for routing the normal traffic while GA find the combination of all the possible working paths for the working capacity planning phase. Therefore, it has a larger search space and can find a better solution. However, as the search space increases in size, the

- convergence time to a better solution will also increase rapidly. This is why we need a large number of population size and number of generation to achieve a high quality solution, that make average running time for GA is much larger than that of MDA. That is a trade off.
2. From Table 3.4, when  $\alpha = 200$  and  $\beta = 4$ , MDA can outperform GA for both link restoration and path restoration. It gives us the insight that MDA is a better tool for designing optimal cost survivable network when the installation cost is much larger than the bandwidth cost. As MDA iteratively chooses the most expensive link to delete until the network cannot satisfy the two-connectedness constraint, it means that it concentrates on optimizing the topology design first. Under this situation, MDA can perform better than GA which always tries to jointly optimize the topology, the working and spare capacity planning and does not concentrate on achieving the optimal cost for a specific topology.
  3. For link restoration, GA can outperform MDA by 16% on average. For path restoration, GA just outperform MDA on average by 4%. While at the same time the running time of GA is about 300 times than that of MDA, much larger than that of MDA with order of  $O(N^2)$ , as estimated in the previous section. This is because GA has a better route diversity which is more suitable for the joint optimization to obtain the least cost survivable network. However, as the path restoration scheme tries to diversify the use of network resources to achieve a more economical spare capacity planning, this scheme then compensates for the more focus effort of MDA and can make it a better algorithm. Therefore, MDA together with path restoration is a cost-effective and efficient tool for designing survivable network because of its low

complexity.

From this experiment, we can say that GA is a better tool for achieving the least cost survivable network in general due to its strong jointly optimization capability.

At the same time, MDA is also a very efficient tool for designing survivable networks under the path restoration scheme or when the installation cost is the determining factor due to its lower complexity and better performance under these situations.

Although the experiment is done with constant  $\alpha_{ij}$  and constant  $\beta_{ij}$ , both GA and MDA can tackle problems with  $\alpha_{ij}$  and  $\beta_{ij}$  change from link to link.

### **3.4 Conclusions**

We have shown that a better optimization can be achieved when the topology design, working capacity planning and spare capacity planning can be optimized simultaneously during the design process. However, the joint optimization problem is a very big and complex problem so two heuristics have been developed for solving this problem. The first one is the Modified Drop Algorithm (MDA), and the second one is the Genetic Algorithm (GA).

Both GA and MDA perform very well but GA generally outperforms MDA by a small amount with the trade off that the run time is much longer than that of MDA.

We can also see that MDA performs very good or even better when compared with GA under the path restoration scheme or when the determining factor is the link installation cost.

In summary, both GA and MDA are efficient and useful tools for designing the least cost survivable network.

## Chapter 4 Conclusions and Future Work

It can be foreseen that all-optical networks will become the foundation of the future telecommunication infrastructure due to the rapid advancement of the optical technology and rapid cost reduction of the corresponding optical components. The main issue that needs to be considered is the network architecture. Another important issue is the survivability. This motivates us to develop a framework for the design of all-optical survivable network.

In this thesis, we first investigate survivability issue. The spare capacity planning problem has been shown to be NP-complete. Many heuristics have been proposed to solve the problem and the most common one up to now is integer programming. For this method, we have shown that the complexity is very large for large size networks as the number of variables and constraints scales with the exponential form of the number of nodes and edges of the network. Other drawbacks are that it cannot provide any insight of how to choose the restoration path set for planning, and that there is no effective method for handling nonlinear objective functions or constraints for integer programming.

We have developed an efficient, generic and robust heuristic called SCAPE for solving the spare capacity planning problem in a way that can provide the insight for choosing real-time path set and it can be expected to solve the nonlinear problems due to its adaptive property. From the experimental result, it has been shown that SCAPE works well in both link restoration and path restoration. This means that SCAPE is a useful, efficient and generic tool for the spare capacity

planning problem.

After tackling the survivability issue, we have also investigated the big complex network design problem for arbitrary network topology. Intuitively, the most economical design should jointly consider the topology, the working capacity planning and spare capacity planning altogether. However, the existing approach for designing such kind of survivable network is to tackle only the topology design, or at most handle the topology and working capacity planning together. This is understandable as each of these three sub-problems is NP-complete. It can be imagined that the complexity of this design problem is so big that it is very difficult to jointly optimize them.

With the help of a low complexity heuristic SCAPE, we can handle the joint optimization problem easily and we develop two heuristics MDA and GA based on SCAPE for solving the joint optimization problem. From the experimental results, it can be shown that the design principle for the joint optimization of topology, working capacity planning and spare capacity planning produces a better result and is the right way to go. Both GA and MDA are shown to be efficient and useful tools for designing the least cost survivable networks. GA is especially good for producing the best cost-optimized design of survivable network with a slightly higher complexity while MDA is a very good, low complexity design tool in combination with the path restoration scheme and where the installation cost is the determining factor.

As the Wavelength Division Multiplexing (WDM) technique seems to be the



most probable technique for the transmission of data in the future all-optical network, the present work needs to be further developed to handle the wavelength continuity problem in both spare capacity planning and network design. This poses a great challenge.

Another important issue is the internetworking problem. When internetworking with IP networks, the design objective will need to include the QoS variables, and this will further increase the complexity of the design problem. These two problems will be investigated in the future.

Finally, what we have handled up to now is limited to one single link failure with symmetric traffic. If asymmetric traffic is introduced and multiple failures are considered, the problem will involve more constraints and become more complex, it would be interesting to see how the present algorithms can be modified to tackle these problems in the future.

In conclusion, this thesis has provided a framework for tackling the future all-optical network architecture design and we hope that the tools SCAPE, MDA and GA developed can provide a strong foundation for the future development of fault-tolerant all-optical networks.

## Appendix A The Interference Heuristic for the path restoration scheme

We have adopted the interference heuristic in SCAPE for backtracking purpose with path restoration scheme. In this appendix, the interference heuristic for the path restoration scheme will be introduced. This interference heuristic is proposed by Iraschko.

Given a survivable network of the form  $G(\mathbf{N}, \mathbf{E}, \mathbf{s})$  where  $\mathbf{N}$  is the set of nodes,  $\mathbf{E}$  is the set of edges, and  $\mathbf{s}$  is the vector of spare capacities where  $s_j$  represents the spare capacity of edge (or link)  $j$ . The path restoration routing problem is defined in the following mathematical model:

$$\text{Max} \left[ \sum_{r=1}^{D_i} \sum_{p=1}^{P_i^r} f_i^{r,p} \right] \quad \forall i = 1, \dots, E;$$

$$\text{subject to } \sum_{p=1}^{P_i^r} f_i^{r,p} = X_i^r, \quad \forall r = 1, \dots, D_i, \quad \forall i = 1, \dots, E;$$

$$\sum_{r=1}^{D_i} \sum_{p=1}^{P_i^r} \delta_{i,j}^{r,p} \cdot f_i^{r,p} \leq s_j, \quad \forall j = 1, \dots, E, \quad \forall i = 1, \dots, E;$$

$$f_i^{r,p} \geq 0, \text{ integer}, \quad \forall i = 1, \dots, E, \quad \forall r = 1, \dots, D_i, \quad \forall p = 1, \dots, P_i^r;$$

where

$D_i$  the number of affected OD pair after link  $i$  is failed;

$P_i^r$  the total number of eligible restoration routes for affected OD pair  $r$  if link  $i$  is failed;

$f_i^{r,p}$  flow assigned to the  $p^{th}$  restoration route for the affected OD pair  $r$  if link  $i$  is failed;

$X_i^r$  the number of traffic demand needed to be rerouted for the affected OD pair  $r$  if link  $i$  is failed;

$\delta_{i,j}^{r,p}$  1 if link  $j$  is used by the  $p^{th}$  restoration route for the affected OD pair  $r$  if link  $i$  is failed, else 0.

From this model, it can be seen that the path restoration routing problem is the integer multicommodity maximum flow (MCMF) problem. This is a complex combinatorial optimization problem. Iraschko proposed an Interference Heuristic for solving this problem with  $O(N^3 \log N)$ . Most of the material in this appendix is adopted from [10].

The principle of this heuristic is simple. Given a network with spare capacity assignment for each link, the restoration is maximized when the particular restoration path chosen causes the minimum interference to the other restoration paths. If a particular restoration path chosen would cause many other restoration paths not useable under the spare capacity planned network, it means that this path could become more costly than others and have the least effect in maximizing the use of the spare capacity in the network. Hence the less interference a path caused, the higher priority the path should be chosen for path restoration.

Based on this principle, the Interference Heuristic is constructed for solving the MCMF problem. The pseudo code is as follows:

```

for (failed link  $i \in E$ ) do
  { release the working capacity portions of all failed paths under link  $i$  failure.
    while (restoration paths are feasible for any unrestored affected OD pair  $r$  under
      link  $i$  failure) do
      { for (every unrestored affected OD pair  $r$  under link  $i$  failure) do
        { find the set of k-isolated-shortest restoration paths for affected OD
          pair  $r$  ;
          }
        for (every unrestored affected OD pair  $r$  under link  $i$  failure) do
          { for (every path  $p$  in the k-isolated-shortest restoration path set for
            affected OD pair  $r$ ) do
              {Compute interference number  $I_i^{r,p}$  of path  $p$  caused.
              }
            }
          }
        find the path  $p_m$  with minimum  $I_i^{r,p}$  and implement.
        remove the spare capacity on all the links used for the path  $p_m$ .
      }
    }
  }

```

More details can be found in [10].

## Bibliography

- [1] T. E. Stern and K. Bala, "Multiwavelength Optical Networks," Addison Wesley, 1999
- [2] W. D. Grover, "Case studies of survivable ring, mesh and mesh-arc hybrid networks," GLOBECOM '92, vol.1, pp. 633 -638, 1992.
- [3] B. Van Caenegem, W. Van Parys, F. De Turck, and P. M. Demeester, "Dimensioning of survivable WDM networks," IEEE Journal on Selected Areas in Communications, vol. 16, pp. 1146 –1157, 1998.
- [4] B. D. Venables, "Algorithms for the spare capacity design of mesh restorable networks," Master of Science Thesis, University of Alberta, Fall, 1992.
- [5] H. Sakauchi, Y. Nishimura, and S. Hasegawa, "A self-healing network with an economical spare-channel assignment," GLOBECOM '90, vol.1, pp. 438 -443, 1990.
- [6] W. D. Grover, T. D. Bilodeau, and B. D. Venables, "Near optimal spare capacity planning in a mesh restorable network" GLOBECOM '91, vol.3, pp. 2007 -2012, 1991.
- [7] M. Herzberg, "A decomposition approach to assign spare channels in self-healing networks," GLOBECOM '93, vol.3, pp. 1601 -1605, 1993.
- [8] B. D. Venables, W. D. Grover, and M. H. MacGregor, "Two strategies for spare capacity placement in mesh restorable networks," IEEE Global Conference on Communications, vol.1, pp. 267 -271, 1993.
- [9] M. Herzberg, S. J. Bye, and A. Utano, "The hop-limit approach for spare-capacity assignment in survivable networks," IEEE/ACM Transactions on Networking, vol.3, pp. 775 –784, 1995.
- [10] R. R. Iraschko, M. H. MacGregor, and W. D. Grover, "Optimal capacity placement for path restoration in STM or ATM mesh-survivable networks," IEEE/ACM Transactions on Networking, vol. 6, pp. 325 –336, 1998.

- [11] R. R. Iraschko, and W. D. Grover, "A highly efficient path-restoration protocol for management of optical network transport integrity," *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 779–794, 2000.
- [12] A. Al-Rumaih, D. Tipper, Y. Liu, and B. A. Norman, "Spare Capacity Planning for Survivable Mesh Networks," *NETWORKING*, pp. 957-968, 2000.
- [13] H. S. Wilf, "Algorithms and Complexity," Prentice-Hall, 1986.
- [14] E. Lawler, "Combinatorial Optimization: Networks and Matroids," Dover, 2001.
- [15] S. Leyffer, "Deterministic Methods for Mixed Integer Nonlinear Programming", Ph.D thesis, Department of Mathematics and Computer Science, University of Dundee, 1993.
- [16] M. Groetschel and C. L. Monma and M. Stoer, "Computational results with a cutting plane algorithm for designing communication networks with low connectivity constraints," *Oper. Res.*, vol. 40, pp. 309-330, 1992.
- [17] S. J. Koh and C. Y. Lee, "A tabu search for the survivable fiber optic communication network design," *Computers industrial Engineering*, vol. 28, pp. 689-700, 1995.
- [18] King-Tim Ko, Kit-Sang Tang, Cheung-Yau Chan, Kim-Fung Man, and Sam Kwong, "Using genetic algorithms to design mesh networks," *Computer*, vol. 30, pp. 56–61, 1997.
- [19] Wenxia Chen and Junli Zheng, "Capacity design of ATM rings with genetic algorithms," *IEEE Asia-Pacific Conference*, pp. 883–886, 2000.
- [20] R. S. Cahn, "Wide Area Network Design: Concepts and Tools for Optimization," Morgan Kaufmann, 1998.
- [21] D. Goldberg, "Genetic Algorithms in Search, Optimization and Machine Learning," Addison-Wesley, 1989.

- [22] M. Gen and R. Cheng, "Genetic Algorithms and Engineering Optimization," John Wiley & Sons, INC, 2000.
- [23] T. H. Wu and S.F. Habiby, "Strategies and technologies for planning a cost-effective survivable fiber network architecture using optical switches," Journal of Lightwave Technology, Vol 8, pp. 152 –159,1990.
- [24] O.J. Wasem,R.H. Cardwell and T.H. Wu," Software for designing survivable SONET networks using self-healing rings," ICC, Vol 1,pp 425-431,1992.

## **Publications:**

- [1] K. S. Ho and K. W. Cheung, "Capacity Planning for Fault-Tolerant All-Optical Network," to appear in Asia-Pacific Optical and Wireless Communications Conference and Exhibition, Shanghai, sponsored by SPIE and CIC, October 14-18, 2002.
- [2] K. S. Ho and K. W. Cheung, "Low Complexity Design of Fault-Tolerant All-Optical Network with Arbitrary Mesh Topology," submitted to the IEEE International Conference on Communications, 2003.





CUHK Libraries



003955615