1-1-2012

A Nonlinear Model For The Optimal Capacity Placement In Self-Healing ATM Networks Based On Link Restoration Strategy

ABM B. Alam
Ryerson University

A NONLINEAR MODEL FOR THE OPTIMAL CAPACITY PLACEMENT IN SELF-HEALING ATM NETWORKS BASED ON LINK RESTORATION STRATEGY

by

A B M Bodrul Alam

B.Sc., Jahangirnagar University, Savar, Dhaka, Bangladesh, 2005

A Thesis

presented to Ryerson University

in partial fulfillment of the

requirements for the degree of

Master of Science

in the Program of

Computer Science

Toronto, Ontario, Canada, 2012

Author's Declaration

Abstract

A Nonlinear Model for the Optimal Capacity Placement in Self-Healing ATM Networks Based On Link Restoration Strategy

Master of Science

Computer Science

Ryerson University

Network Survivability is a critical issue in telecommunications network due to increasing dependence of the society on communication systems. Fast restoration from a network failure is an important challenge that deserves attention. This thesis addresses an optimal link capacity design problem for survivable asynchronous transfer mode (ATM) network based on the link restoration strategy. Given a projected traffic demands and the network topology, capacity and flow assignment are jointly optimized to yield the optimal capacity placement. The problem is formulated as large-scale nonlinear programming and is solved using a specific type of Lagrange method (so called Separable Augmented Lagrangian Algorithm or SALA for short). Several networks with diverse topological characteristics are used in the experiments to validate our proposed novel model, using capacity installation cost, routing cost, total network cost, used capacity and required CPU time, as performance metrics. Link restoration strategy is compared against global reconfiguration strategy using these performance metrics.

Acknowledgement

It is my honour to express my gratitude to people who made this challenge possible for me. First of all, I would like to thank my supervisor Dr. Isaac Woungang, for giving me the opportunity to work under him. He offered me such a challenging topic for my thesis and with his strong support and guidance; I am finally able to achieve my goals. He did not only guide me in the course of this thesis, but also provided me an opportunity to be benefited from his vast knowledge. He was always present to help me out and guide me whenever I needed his guidance.

I would also like to thank my co-supervisor, Dr. Oumar Mandione Guèye, for sharing his enormous knowledge about mathematical optimization, modified Newton method, SALA and for giving his time and guidance to make my journey as a graduate student a success.

I would like to acknowledge my gratitude to Department of Computer Science, Ryerson University, for financial support and work experience as a graduate student.

Lastly, and most importantly, I would like to thank my wife Faria Khandaker. Without her support, courage and adorableness I would not be here today. I dedicate this thesis to her.

Dedication

To my wife

Faria Khandaker

I would not be here without her strong belief in me.

Contents

List of Tables

List of Figures

List of Abbreviations

CFA	Capacity and Flow assignment
ATM	Asynchronous Transfer Mode
SALA	Separable Augmented Lagrangian algorithm
QoS	Quality of service
GMPLS	Generalized multi-protocol label switching
TP	Transmission Path
VP	Virtual Path
VC	Virtual Circuit
VPI	Virtual Path Identifier
VCI	Virtual Circuit Identifier

Chapter 1

Introduction

1.1 Context of Our Study

The Asynchronous Transfer Mode (ATM) is a major technology that supports the Broadband Integrated Services Digital Network (BISDN) technology. ATM networks allow transferring different kinds of services such as voice, video, data; with different bandwidth requirements and different quality of services (QoS). Because of the importance of ATM networks, the importance of the survivable ATM networks design cannot be neglected. A survivable ATM network can be defined as a network which can restore all failed traffics by itself when a failure happens to it.

Since ATM networks play an important role in modern life, issues of their survivability [1] have gained much attention in recent years. The design of survivable ATM networks is a specific aspect of a more general problem, referred to as the Capacity and flow assignment (CFA) problem [2]. The CFA problem has been intensively investigated in the literature of network survivability [3]. It consists in finding the optimal capacity placement and traffic flows of a network with the target of achieving minimum cost even if a failure occurs in the network, given the traffic demand and network topology.

1.2 Motivation

Currently, in overlay transport networks (IP/ATM/SONET/DWDM), each control plane is managed by its own layer. In recent years, the Generalized Multi-Protocol Label Switching

(GMPLS) has emerged as a new unified control plane for the above transport layers. The already installed ATM core self management features can be reused directly or with some adaptation as a particular implementation of GMPLS. As ATM has become popular, a lot of research works has been done in survivability of networks.

Among such research works, related to CFA problem, most of them are formulated as a linear multicommodity flow problem. According to best of our knowledge, only the work presented in [4], formulated the problem as a nonlinear programming problem, with near optimal solution. This feature motivated us to continue in the same direction and propose some novel alternative and innovative approach. Moreover, most of the previous works considered that a cost function of the self-healing ATM network is based only on the capacity installation cost, but in our approach, we consider both the capacity installation cost and transit delay cost.

1.3 Objective of the Thesis

This thesis proposes a novel mathematical model for the CFA problem under link restoration strategy for a survivable ATM network when a link failure occurs in the network. We only treat the case when one link fails at a time. Basically, we propose a link restoration strategy which aims to find the optimal capacity and traffic flow for a survivable ATM network when a link failure occurs in the network provided that the network topology and the traffic demands of the network are known in advance. We formulate the CFA problem as a nonlinear multicommodity flow problem, which we solve by using a Lagrangian technique called the Separable Augmented Lagrange Algorithm (SALA).

The nonlinearity comes from the objective function of the proposed mathematical model, where both the routing cost and capacity installation cost are considered. The design objective of our proposed strategy is to optimize the capacity and flow allocation while minimizing the network cost and achieving all traffic constraints and survivability requirements. Unlike other relevant similar works [5-6], we consider both the routing cost and capacity installation cost in our objective function.

1.4 Thesis Scope

In our thesis, we consider a preplanned restoration scheme in the design of survivable ATM networks under a single link failure scenario. The design objective is to provide the network survivability cost effectively. Our proposed link restoration mathematical model considers all possible commodities (all source-destination pairs) of the network. Both symmetric and asymmetric traffic demands are considered and it is assumed that traffic demands per node pair as well as the network topology are given. It is also assumed that 100% restoration is the target, where restoration can occur at the virtual path (VP) or the virtual channel (VC) layers and no capacity modularization is considered. In practicality, the users of the network are unconcerned by the restoration mechanism. Typically, the notion of reliability is closely tied to the QoS that the users receive under normal operation of the network or when a failure occurs (failure state).

In this thesis, the reliability (i.e. survivability) objective is considered at the application layer in terms of QoS delivered to the user. Due to the fact that failure can occur at various layers, we take the view that our network design should follow an integrated procedure which takes into account all layers and their failure mechanisms. For the actual implementation of our network, we consider only the link restoration as restoration mechanism for a given link failure scenario, assuming that all the other layers are made aware of this procedure, so that no other restoration procedure is started by them.

1.5 Thesis Contribution

The objectives of this thesis are as follows:

- Proposing a novel nonlinear model for the CFA problem under link restoration strategy.
- Proposing a Separable Augmented Lagrangian Algorithm (SALA) to solve the CFA problem.
- Comparing the global reconfiguration strategy versus link restoration strategy in different scenarios with diverse network topologies.

1.6 Thesis Organization

In Chapter 2, some basics of ATM networks, along with the restoration strategies that are considered in our context are discussed. Some mathematical foundations of our techniques are also presented.

In Chapter 3, our optimization model for link restoration strategy is described along with the proposed solution approach.

In Chapter 4, we describe our results based on several performance metrics.

In Chapter 5, we conclude our thesis and highlight some future works.

Chapter 2

Background and Related Work

In this chapter, we first introduce some definitions related to Asynchronous Transfer Mode (ATM) technologies. We also discuss the restoration strategies that are used in the design of self healing networks. Finally representative previous research works are described.

2.1 ATM Networks

ATM is designed to transmit voice, video and data, all in the same network. Different types of traffic have different tolerance for packet loss and end-to-end delay. Packets containing voice are to be delivered on time and some loss of data may not deteriorate the quality of the voice. On the other hand, while transmitting a data file, the loss of data cannot be tolerated, but there is no stringent requirement that it should be delivered as fast as possible. The main features of ATM networks are discussed as follows.

2.1.1 Characteristics of ATM

The ATM Packet is known as a cell, and it has a fixed size of 53 bytes as shown in Fig. 2.1. It consists of a payload of 48 bytes and a header of 5 bytes.

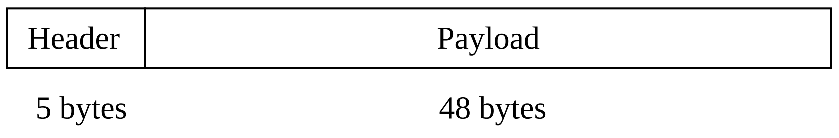


Figure 2.1: ATM Cell

ATM uses asynchronous time division multiplexing. ATM multiplexers fill a slot with a cell from any input channels, and the slot remains empty if there is no cell to send from any of the channels. Figure 2.2 illustrates this fact.

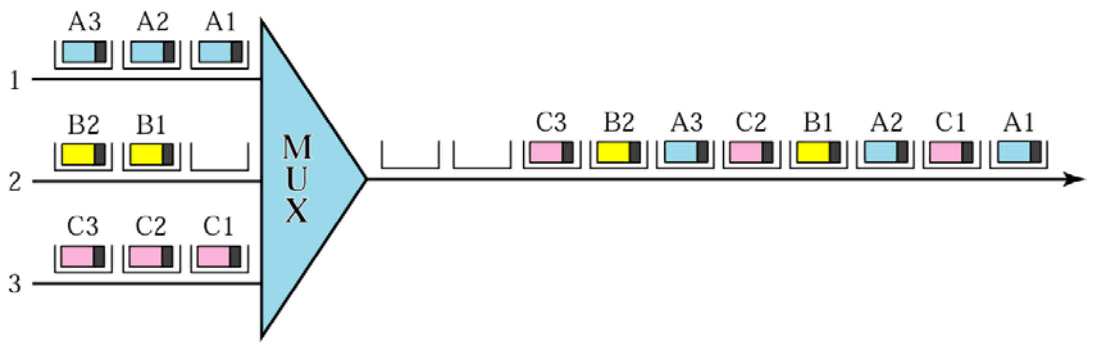


Figure 2.2: ATM Multiplexing [7]

Here, there are 3 input channels that are to be multiplexed. At the first time slot, there is no cell from channel 2. Therefore, the multiplexer fills the slot with a cell from channel 3. When there is no cell from any of the channel, the slot is empty.

In ATM, cells belonging to a virtual connection follow the same path. The bandwidth is assigned based on the requirement and available capacity at the time of connection. ATM uses Virtual Path Identifier (VPI) and Virtual Circuit Identifier (VCI) fields of a virtual connection to forward cells. At each intermediate node, VCI/VPI values are swapped and new values are filled. This process is referred to as cell relaying and cell switching. The virtual circuits in ATM are classified into two categories: Virtual Channel connection (VCC) and Virtual Path connection (VPC) as illustrated in Fig. 2.3.

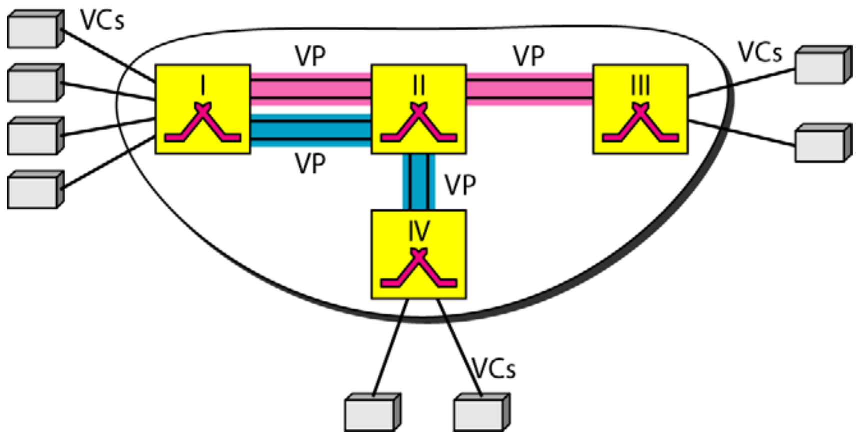


Figure 2.3: Virtual Paths (VPs) and Virtual Channels (VCs) [7]

In Figure 2.3, 8 endpoints are communicating using 8 VCs. The first 2 VCs uses the same path from switch I to switch III, so we can bundle this 2 VCs together to form 1 VP. The other 2 VCs share the same path from switch I to switch IV, these 2 VCs can be grouped together.

The ability to guarantee QoS is one of the biggest strengths of ATM technology. ATM refers to limiting the cell loss, the transit delay and the delay variation induced by the ATM network.

2.1.2 ATM Network Architecture

ATM network interfaces are of different types as shown in Figure 2.4. Prominent among these is the User-Network interface (UNI). There are 2 variations of the UNI: public and private. Network Node Interface (NNI) provides an interface between 2 ATM switches. The Data Exchange Interface (DXI) defines a standardized method to interface between frame based devices.

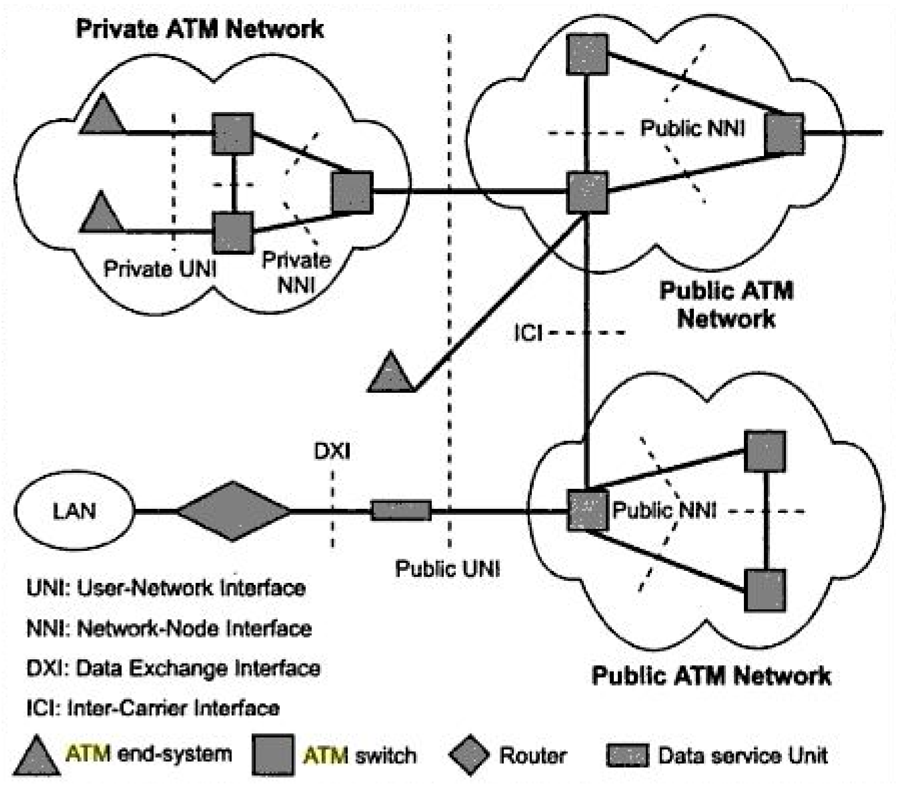


Figure 2.4: Architecture of an ATM Network [8]

The ATM reference model is defined in ITU-T recommendation [ITU-T I.321]. Figure 2.5 depicts the model, which is made of user plane, control plane and management plane.

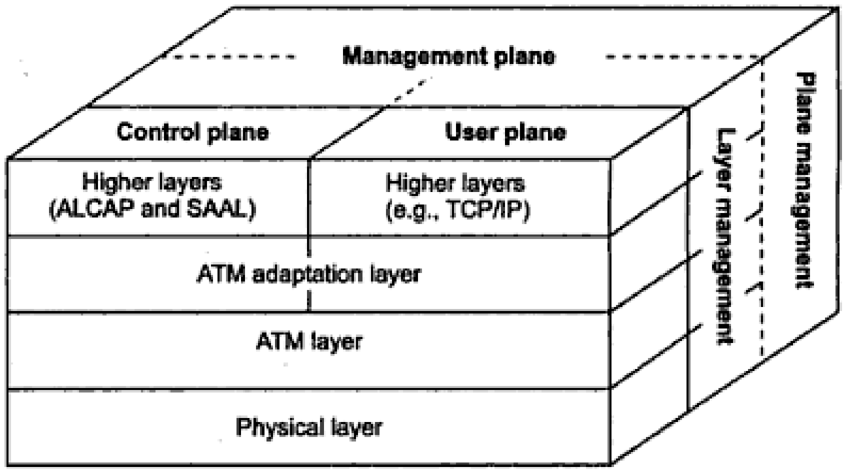


Figure 2.5: ATM Reference Model [8]

The user plane is for transferring the user's information, while the control plane performs the connection control functions. The management plane is used for maintenance. Each plane is divided into a number of layers but only the lower three layers are specified by the ITU-T. The first one is the physical layer. The functions of this layer are bit transfer, bit alignment, line coding, cell rate decoupling, cell header processing, scrambling/ descrambling and frame generation and recovery.

The next layer is the ATM layer. The main functions of this layer are cell multiplexing and de-multiplexing, cell VPI/VCI translation, cell header generation, generic flow control, congestion notification and connection assignment and removal.

The third layer is the ATM Adaptation layer. Depending upon the type of application, this layer performs a wide variety of functions. AAL is divided into segmentation and reassembly (SAR) sub-layer and convergence sub-layer (CS). These entities help in achieving routing in ATM.

Routing is the act of forwarding packets from source to destination. In ATM, routing is considered as signalling. Signalling is used to establish switched virtual connections. The purpose of signalling is to route the signalling messages from source to destination. To provide the means to route signalling messages, ATM forum has defined the Private Network-to-Network Interface (PNNI) protocol.

2.1.3 Current Transport Control Plane

Up to recently, the vision of designing a network with an integrated view of all the reliability mechanisms embedded was still a challenge from an implementation view point. This situation has changed since the Internet Engineering Task Force (IETF) has introduced the Generalized Multiprotocol Level Switching (GMPLS) technology. In the current overlay network (IP-over-ATM-over-SONET), each layer has its own control plane which can imitate restoration independently of what happens in other layers. With the advent of GMPLS, a new unified control plane and signalling functions is introduced, which is used for the design of multi layers transport networks including next generation networks.

In this context, the ATM core implementation framework can be reused in GMPLS, provided that some adjustments are made on it. In this respect, ATM technology is still a viable solution to most applications.

2.2 Network Restoration Strategy

Network restoration (also called survivability) is the ability of a network to function properly even in case of a failure. The designing of a restorable network [9] is an optimization problem that is optimizing the capacity and flow assignment maintaining several constraints.

Generally, the restoration schemes can be classified into two categories: reactive restoration scheme and preplanned restoration scheme as shown in Figure 2.6.

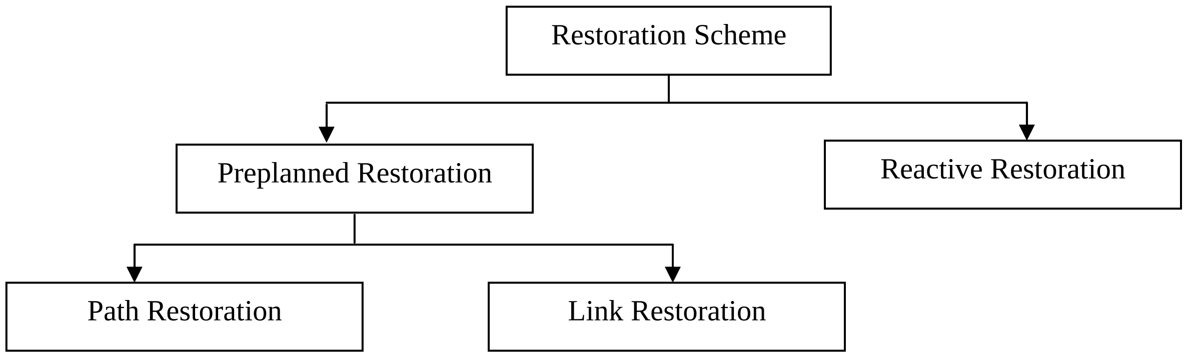


Figure 2.6: Classification of Restoration Scheme

- **Reactive Restoration**

Reactive restoration scheme starts to search for alternate routes after the failure occurs by broadcasting the restoration messages and then reroutes the affected traffic. Reactive restoration scheme is more suitable for networks with rapid change of traffic patterns. This restoration scheme requires little knowledge about the current status of the network.

- **Preplanned Restoration**

In preplanned restoration scheme [5], all the restoration paths for all the commodities with sufficient bandwidth are pre-computed. In case of a failure, a node responsible for restoring the affected traffic activates the restoration route that is precomputed and reroutes the affected traffic. Preplanned restoration is used in the network environment where traffic patterns do not change frequently.

The preplanned restoration scheme can further be classified into path and link restoration. Path restoration can further be divided into two categories global and failure-oriented reconfiguration. In global reconfiguration, all the traffic (affected and unaffected) of all the commodities are rerouted whereas in failure-oriented reconfiguration only the affected commodities are rerouted when a failure occurs.

In this thesis, we are concerned with link restoration strategy as shown in Figure 2.7. Link restoration is a local restoration scheme where the source node of the failed link and the original destination make a decision to reroute the affected traffic. So it is obvious that this scheme is less flexible compare to path restoration.

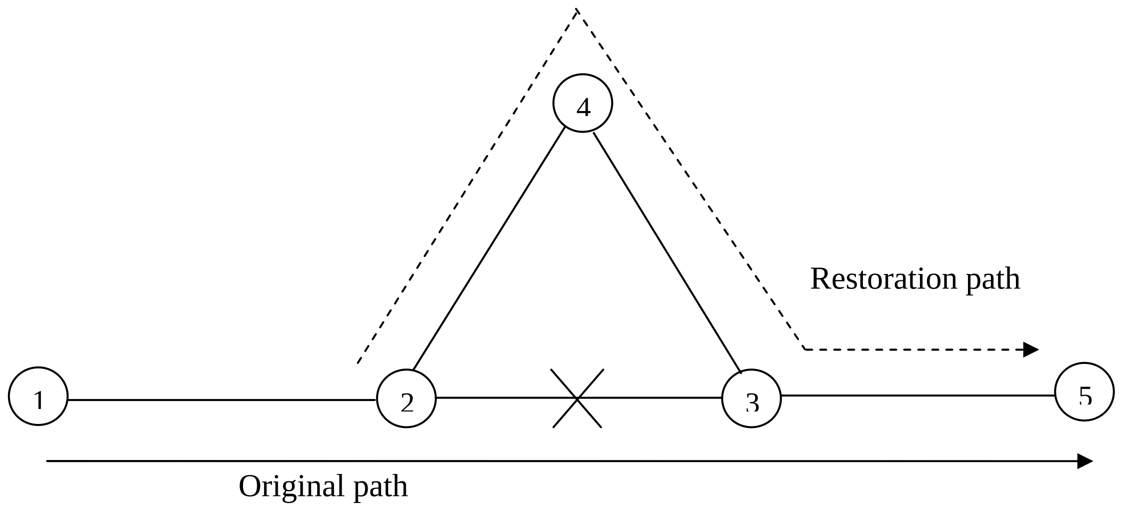


Figure 2.7: Link restoration

In figure 2.7, node 1 and node 5 are the source and destination of a network. Now the original path is 1-2-3-5. But the link between node 2 and node 3 has failed. As we are considering the link restoration strategy the source node of the failed link (i.e. node 2) and the original destination (i.e. node 5) decide how to reroute the affected traffic. Therefore, the flow will be rerouted through the path 2-4-3-5.

2.3 Mathematical Optimization Techniques

Mathematical optimization [10] is a field where the aim is to find a minimum or maximum for a given function. The general form of an optimization problem (P_1) can be as follows:

$$\min_{x} f(x) \tag{2.1}$$

$$s.t. \;\; g(x) = 0$$

where $f(x)$ is the objective function, $g(x)$ describes the constraints and x is a vector in $\mathbb{R}^n$ (i.e. a vector with n real components) generally referred to a decision variable. The aim is to find a value of x for which the objective function is minimum and all the constraints in $g(x)$ are satisfied.

A variable x verifying the constraints (i.e. for which $g(x)=0$) is called feasible solution. The set of feasible solutions is called a feasible set.

When f is linear, the problem (P_1) is called linear optimization problem. Moreover, if the variable of the objective function and the constraint is integer, (P_1) is called linear integer problem. If some of the variables are integer and some are continuous, then that problem is called Mixed Linear Integer Programming problem.

If f is nonlinear, (P_1) is called a nonlinear optimization problem. If the variables are integers then it is called nonlinear integer problem which is much more difficult to solve compared to the linear integer problem. The problem may also be mixed integer non linear optimization problem, which is even more difficult to solve.

2.3.1 Unconstrained Nonlinear Optimization Problem

Without the constraints $g(x)=0$ in Equation (2.1), the problem (P_1) is called unconstrained optimization problem. There are several efficient methods to solve unconstrained nonlinear problems [11].

Let's consider the following unconstrained problem

(P_1): $\min_x f(x)$ where $x \in \mathbb{R}$ and $f(x)$ is twice differentiable.

The algorithm for solving (P_1) using gradient methods is as follows:

Algorithm for Unconstrained Nonlinear Optimization Problem

```
Step 1: Initialization
Choose $\varepsilon > 0$ (very small) and an initial value of $x_0$. Then set $t = 0$. $\varepsilon > 0$
Step 2: If $\|\nabla f(x_t)\| < \varepsilon$ and $\nabla^2 f(x_t) > 0$ (where $\|x\|$ denotes the norm of x) stop.
Else go to step 3
Step 3: Compute $x_{t+1} = x_t + \alpha_t d_t$
Set $t = t + 1$ and go back to step 2.
```

The following is an explanation of the above algorithm:

- d_t is called a descent direction because it decreases the value of the objective function. Several descent directions are proposed and each of them corresponds to a specific method with different convergence rate.
- When $d_t = -\nabla f(x_t)$, the method is called a steepest descent method.
- When $d_t = -\left[\nabla^2 f(x_t)\right]^{-1} \nabla f(x_t)$, the method is called Newton method. Sometimes $\nabla^2 f(x_t)$ is not positive defined or does not exist. In that case, d_t is not a descent direction and $\nabla^2 f(x_t)$ should be modified by replacing it with a definite positive matrix. This involves using a modified Newton method rather than computing the inverse of $\nabla^2 f(x_t)$, because doing so is time consuming.
- α_t is called the step size in the descent direction. In this thesis we use the Armijo method [12] to find a suitable α_t.

- The convergence analysis of this type of algorithms [13] has generated sequences x_t at a stationary point, but the speed of convergence is not the same for all of them.

2.3.2 Constrained Nonlinear Optimization Problem

If f and g are differentiable in the problem (P_1) stated in Equation (2.1) , the presence of the constraints makes the problem more difficult to solve. Typically, Lagrangian methods combined with penalty methods are used to solve such problems.

2.3.2.1 Penalty Method

A Penalty method consists of modifying the objective function by adding the constraints. Several penalty methods have been proposed in the literature [12]. Here, we present one of them called the exact penalty. It consists of transforming the problem (P_1) into the following equivalent problem:

$\min\limits_x f(x) + \frac{\lambda_t}{2}\|g(x)\|^2$, where $x \in \mathbb{R}$, λ_t is sequence converging to infinity when $t \to \infty$. λ_t is called a penalty parameter and when $\lambda_t \to \infty$, this enforces the current solution to be feasible for problem (P_1). The constrained problem $(P_1)$is transformed into an unconstrained problem and any unconstrained optimization method can be used to solve it. Basically, the solution approach works as follows:

Penalty Methods

Step 1: Initialization

Choose $\lambda_0 > 0$ *and set* $t = 0$.

Step 2: Solve $\min_x f(x) + \frac{\lambda_t}{2}\|g(x)\|^2$ *with an unconstrained method and go to step 3*

Step 3: If solution is optimal stop; otherwise go to step 4.

Step 4: Update λ_t *and go back to step 2.*

λ_t *is chosen such that* $\lambda_t \to \infty$ when $t \to \infty$. λ_t *can be updated in several ways. For instance,* $\lambda_{t+1} = \beta\lambda_t$ *where* $\beta > 1$.

2.3.2.2 Lagrangian Method

The Lagrangian method proposes another way to modify the objective function of problem (P). In this case, the problem (P_1) is replaced by the following problem:

$$\min_x f(x) + u_t g(x)$$

where $x \in \mathbb{R}$ and u_t is a sequence of parameters called Lagrange multipliers or dual variables. $f(x) + u_t g(x)$ is called the Lagrange function. This method works as follows:

Lagrangian Algorithm

```
Step 1: Initialization
Choose $u_0$ and set $t = 0$.
Step 2: Solve $\min_x f(x_t) + u_t g(x_t)$ with an unconstrained method and go to step 3
Step 3: If solution is optimal, stop. Otherwise go to step 4.
Step 4: Update $u_t$ and go back to step 2.
```

2.3.2.3 Augmented Lagrangian Method

The Augmented Lagrangian method (also called multipliers method) is a hybrid method between Lagrangian method and Penalty method. We present here the case using the exact penalty term. It should be noted that we use this method in the context of this thesis. In this method, the problem (P_1)in Equation (2.1) is transformed as follows:

$$\min_x f(x) + u_t(g(x))\frac{\lambda_t}{2}\|g(x)\|^2$$

where u_t are Lagrange multipliers and λ_t are penalty parameters; but here, λ_t is not required to go to infinity. This method works as follows:

```
Augmented Lagrangian Algorithm

Step 1:  Initialization
Choose $\lambda_0 > 0$ and $u_0$, then set $t = 0$.
Step 2: Solve $\min_x f(x) + u_t(g(x))\frac{\lambda_t}{2}\|g(x)\|^2$ with an unconstrained method and go to step 3
Step 3:  If solution is optimal, stop; otherwise go to step 4.
Step 4:  Update $u_t$ and $\lambda_t$ as follows:
$\lambda_{t+1} = \beta\lambda_t$; $u_{t+1} = u_t + \lambda_t g(x_t)$ and $t = t+1$. Then go back to step 2.
```

The stopping criterion is based on the Karush-Khun-Tucker(KKT) optimality conditions[14]. In this thesis a novel special case of the Augmented Lagrangian method (SALA) is proposed to solve our proposed CFA problem based on link restoration strategy.

2.4 Related Work

The design of survivable ATM networks is an instance of a classical problem named as Capacity and Flow Assignment (CFA) problem. This problem was intensively investigated by many researchers in the recent years. Typically, the CFA problem is to determine the optimal capacity and flow of all the links of a network while keeping the network cost minimum when the traffic demand and bandwidth requirements are known. Many researchers have investigated the CFA problem as a linear multicommodity flow problem. Only one of them [4] has modeled the problem as a nonlinear multicommodity flow problem.

In [5], Xiong and Mason formulated the CFA problem arising in the design of survivable ATM networks as a linear programming problem. They solve the problem by using a standard method and a heuristic algorithm, with the objective to minimize the spare cost. They compared several restoration strategies quantitatively, in terms of spare cost. However, in their scheme, it is assumed that no capacity modularization is made to conform to physical transportation systems. In addition, their proposed formulation does not treat the case of non-bifurcated flow restoration.

In [15, 16] Murakami and Hyong discussed the same issue based on link and end-to-end path restoration. They proposed a joint optimization scheme to address the CFA problem. Simulation results were provided, sharing the advantages of end-to-end restoration compared to link restoration. However, in their scheme, the case of node failure scenario was not investigated.

In [17], Murakami and Kim proposed a large scale linear programming method to jointly optimize the capacity and flow assignment problem with a target of minimizing the cost. The authors tested their method on two sample networks for several traffic demands and found that a significant amount of cost saving is possible through their method. However, this method is only applicable to preplanned restoration schemes.

In [18], the issue of choosing backup virtual paths on the optimized spare capacity allocation was discussed. The authors compared four Spare Capacity Allocation (SCA) design schemes quantitatively in terms of spare capacity requirements. Moreover, a link based design approach was also introduced and the solution obtained was shown to be adequate in terms of grade of service (GoS) and quality of service (QoS) requirements.

In [19], Cheng-Shong Wu et al. addressed the CFA problem and proposed a linear programming formulation for both path and link restoration schemes to minimize the usage of spare capacity. After evaluating their method, they concluded that their proposed scheme was good efficient in spare capacity usage and had a fast restoration time. But this method is applicable only if the backup virtual paths are precomputed before the failure occurs.

In [4], Murakami and Hyong investigated the CFA problem and proposed a non-linear, non-smooth multicommodity flow formulation with linear constraints. They demonstrated that their algorithm using modified flow deviation method can converge to a near optimal solution by properly adjusting the optimization parameters. However, avoiding premature convergence is a challenge in their method.

In [20], Herzberg et al. presented an approach for spare capacity assignment which reduces the total weighted cost. Their approach takes into account that the restoration routes do not violate the predetermined hop limit value, and the CFA problem is formulated as a linear programming problem which includes two parts. The first part provides a lower bound solution on the spare cost and the second part rounds up the solution to a feasible one. The authors also compare the result of this approach with existing one. The authors apply their scheme in only small and moderate networks, for large networks they do not propose any formulation.

In [21], Dunn et al. investigated the CFA problem as well. They showed that the k-successive shortest link disjoint paths (KSP) are faster, easier and more robust in restoration for span failure and presented a comparative study of the effectiveness of KSP versus Max flow for restorable networks. However, KSP cannot provide an optimal solution for all types of networks. In addition, they do not consider the path restoration and node failure scenario.

In [22], Grover et al. developed a heuristic algorithm to solve the problem of near-optimal spare capacity assignment in mesh type networks for span failure. They applied the k-shortest link disjoint paths, assuming that rerouting capability can be done in the network. However, the full trade off between restorability and redundancy was not revealed in this approach. Their results showed that their approach is applicable for real world planning applications.

In [23], Nishimura et al. proposed a distributed control mechanism which is applicable for both link and path restoration. The method allows the shared use of spare channels for various failure scenarios. As a result, the efficient use of spare channels can be achieved. Using a network-flow technique, a linear programming based scheme was also proposed leading to a fast restoration. However, the complexity of their proposed algorithm was not investigated.

In the literature we found that a lot of research was done in this field, but most of them formulated the CFA problem as a linear multicommodity flow problem. Only a few [2] treated the CFA problem using a nonlinear model. Unlike the method in [2], in our proposed method, we consider both the routing cost and the capacity installation cost within our objective function. Our method finds a near optimal solution while achieving a fast restoration time.

Chapter 3

Optimization Model and Solution Approach

In this chapter, our novel mathematical model using the link restoration strategy is described first. Then the techniques used for solving our model are described in depth, followed by our proposed algorithm for the link restoration strategy.

3.1 Optimization Model for Link Restoration

Link restoration strategy is a local restoration scheme where the source node of the failed link and the original destination decide to reroute the affected traffic. For a better understanding, we consider the notations given in Table 3.1.

Table 3.1: Parameters used in our Link Restoration Model

Notations	Meaning
$N(A,B)$	A graph N where A is the number of nodes and B is the number of arcs
γ_a	Transit delay unit cost
K	Number of commodities
r_k	Traffic demand (bandwidth requirement) of commodity k
Q_l	Total number of backup paths when the arc l fails
z_a^l	Aggregate flow on the arc a when the arc l fails
π_{kp}^a	Equals to 1 if the p^{th} path of commodity k uses the arc a and 0 otherwise
C_a^l	Capacity on arc a when the arc l fails
$x_{kp}^{l_0}$	The flow commodity k using by its p^{th} path p when the arc l_0 is not failed
d_a	Unit cost of capacity installation
y_q^l	Aggregated flow rerouted on backup path q when the arc l fails
π_{kp}^l	Equals to 1 if p^{th} path of commodity k is affected when the arc l fails, otherwise 0
θ_{aq}^l	Equals to 1 if the q^{th} backup path uses the arc a when the arc l is failed, otherwise 0
$k_{kp}^{al} = 0$	If the p^{th} path of commodity K is affected when the arc l fails a is a downstream arc with respect to l, 1 otherwise

The traffic on the failed arc l is considered as one commodity originating from the upstream node of the link failure. With a link restoration strategy, the objective function can be written as follows:

$$\min \sum_{a=1}^{n} [\gamma_a \frac{z_a^l}{C_a^l - z_a^l} + d_a C_a^l \ ] \tag{3.1}$$

$s.t.$

$$z_a^l = \sum_{k=1}^{K} \sum_{p=1}^{N_k} k_{kp}^{al} \pi_{kp}^{a} x_{kp}^{l_0} + \sum_{q=1}^{Q_l} \theta_{aq}^{l} y_q^l, \forall a = 1,.....,n; a \neq l \tag{3.2}$$

$$\sum_{p=1}^{N_k} x_{kp}^{l_0} = r_k, \forall k = 1,...........,K, \tag{3.3}$$

$$\sum_{q=1}^{Q_l} y_q^l = \sum_{k=1}^{K} \sum_{p=1}^{N_k} \pi_{kp}^{l} x_{kp}^{l_0}, \tag{3.4}$$

$$0 \le z_a^l \le C_a^l, \forall a = 1,..........,n, \forall s \in S \tag{3.5}$$

$$x_{kp}^{l_0} \ge 0, \forall k = 1,......,K, \forall p = 1,..........,N_k \tag{3.6}$$

$$\mathrm{y}_\mathrm{q}^\mathrm{l} \ge 0, \forall \mathrm{q} = 1,.....,\mathrm{Q}_{\mathrm{l},} \forall \mathrm{l} \tag{3.7}$$

- The target of the objective function is to minimize the routing cost and capacity installation cost. The first term of the objective function is the routing cost given by the Kleinrock's average delay function [14] where γ_a is the transit delay unit cost of the arc a, C_a^l and z_a^l denote respectively the capacity and traffic on arc a, when the arc l has failed. The second term of the objective function is the cost of the installed capacities where d_a is a unit cost of capacity installation.

- In constraint (3.2), y_q^l is the traffic flow on backup path q, Q_l is the total number of backup paths when the arc l fails, $x_{kp}^{l_o}$ denotes the amount of flow commodity k using its p^{th} path when no failure occurs in the network, θ_{aq}^l equals to 1 if the q^{th} backup path uses the arc a when a failure occurs on arc l, and 0 otherwise, $k_{kp}^{al} = 0$ if the p^{th} working path of the commodity k is affected when the arc l fails and a is a downstream arc with respect to l, otherwise $k_{kp}^{al} = 1$.
- Constraint (3.2) assures that the traffic flow on an arc a when a failure happens on arc l is the summation of flows on the working paths and the backup paths using that arc a.
- Constraint (3.3) is the flow requirement constraint, which ensures that all bandwidth demands of all commodities using all paths are fulfilled, i.e. the bandwidth requirement is met.
- Constraint (3.4) expresses that all traffic is ensured to be rerouted on the backup paths when an arc l fails.
- Constraint (3.5) is the capacity constraint which ensures that the traffic flow on an arc a should be positive and less than the capacity of that arc a.
- Constraint (3.6) and (3.7) are positivity constraints, which ensures that the traffic flows of every commodity on all of its paths should be positive

3.2 Solution Approach

To solve the above model (3.1) to (3.7) we use the Separable Augmented Lagrangian Algorithm (SALA), which involves using Dijkstra algorithm, Modified Newton method and the

Armijo method. We describe all of the methods in this section. At first we describe some basics about SALA and then apply SALA in our link restoration model.

3.2.1 Separable Augmented Lagrangian Algorithm (SALA)

SALA was proposed and extensively studied in [12] by Hamdi. It was designed for separable nonlinear large scale programming. For a better understanding, we consider the following separable problem:

$$\min \sum_{j=1}^{n} f_j(x_j) \tag{3.8}$$

$s.t.$

$$\sum_{j=1}^{n} g_j(x_j) = 0 \tag{3.9}$$

$$x_j \in S_j, \quad \forall j = 1,\dots,n. \tag{3.10}$$

when function f_j is twice differentiable, x_j are n_j-vectors and S_j are closed and bounded subsets of $\mathbb{R}^{n_j}$. $g_j : \mathbb{R}^{n_j} \to \mathbb{R}^m$ are smooth functions and constraints (3.9) are coupling constraints. To exploit the separability of the objective function, SALA consists to introduce allocation vectors y_j to decouple the constraints (3.9) by setting $y_j = -g_j(x_j), \forall j = 1,\dots,n.$ We obtain the following equivalent problem:

$$\min \sum_{j=1}^{n} f_j(x_j) \tag{3.11}$$

$$s.t. \sum_{j=1}^{n} y_j = 0 \tag{3.12}$$

$$g_j(x_j) + y_j = 0, \quad \forall j = 1,\ldots,n \tag{3.13}$$

$$x_j \in S_j, \qquad \forall j = 1,\ldots,n \tag{3.14}$$

Applying the Augmented Lagrangian method [13], we obtain the following:

$$\min \sum_{j=1}^{n} \left[f_j(x_j) + u_j (g_j(x_j) + y_j) + \frac{\lambda}{2} \left\| g_j(x_j) + y_j \right\|^2 \right] \tag{3.15}$$

$$s.t. \quad \sum_{j=1}^{n} y_j = 0 \tag{3.16}$$

$$x_j \in S_j, \quad \forall j = 1,\ldots,n \tag{3.17}$$

In the above equations, u_j are Lagrange multipliers and $\lambda > 0$ is a penalty parameter. To solve problem (3.15)-(3.17), Hamdi et al. proposed in [12] to minimize alternatively with respect to x_j and y_j. Note that at iteration t, when the problem is solved with respect to one variable the other one is supposed to be known. At iteration t, we then obtain the following sub problems:

$$x_j^t = \arg\min_{x_j \in S_j} \left\{ f_j(x_j) + u_j^t \left(g_j(x_j) + y_j^{t-1} \right) + \frac{\lambda_t}{2} \left\| g_j(x_j) + y_j^{t-1} \right\|^2 \right\} \qquad \text{(a)}$$

$$y_j^t = \arg\min_{\sum_j y_j = 0} \left\{ u_j^t \left(g_j(x_j^t) + y_j \right) + \frac{\lambda_t}{2} \left\| g_j(x_j^t) + y_j^{t-1} \right\|^2 \right\} \qquad \text{(b)} \qquad (3.18)$$

$$u_j^{t+1} = u_j^t + \lambda_t \left(g_j(x_j^t) + y_j^t \right) \qquad \text{(c)}$$

The equation (3.18)-(c) is the Lagrange multipliers formula. [12] The sub-problems can be solved explicitly by using the Karush-Kuhn-Tucker conditions [13]. We then obtain,

$$y_j^t = -g_j\left(x_j^t\right) + \frac{1}{n}\sum_{j=1}^{n} g_j\left(x_j^t\right), \quad \forall j = 1,\dots,n \quad \text{and } u^{t+1} = u^t + \frac{\lambda_t}{n}\sum_{j=1}^{n} g_j\left(x_j^t\right).$$

From the above Equation it can be observed that the same dual vector is used for all sub-problems [12]. The Separable Augmented Lagrangian Algorithm can then be presented as follows:

```
*Separable Augmented Lagrangian Algorithm (SALA)*
1. Choose $u^1$, $\lambda > 0$, $\varepsilon > 0$, $\beta > 0$, $y_j^0$ such that $\sum y_j^0 = 0$ and $t = 1$.
2. $\forall j = 1,\dots,n$ Compute
   $x_j^t = \arg\min_{x_j \in S_j} \left\{ f_j(x_j) + u_j^t\left(g_j(x_j) + y_j^{t-1}\right) + \frac{\lambda_t}{2}\left\| g_j(x_j) + y_j^{t-1} \right\|^2 \right\}$
3. Compute
   $\delta^t = \sum_{j=1}^{n} g_j\left(x_j^t\right)$
   If $\left\|\delta^t\right\| < \varepsilon$ stop. Else go to step 4.
4. Update formula
   $y_j^t = -g_j\left(x_j^t\right) + \frac{1}{n}\delta^t$
   $u^{t+1} = u^t + \frac{\lambda_t}{n}\delta^t$
   $\lambda^{t+1} = \beta\lambda^t$
   $t \leftarrow t+1$ and go back to step 2.
```

3.2.2 SALA Applied on Link Restoration Strategy

Now, let's apply SALA to the link restoration model (3.1) to (3.7). For this purpose, let u be the dual vector defined by the following $u = (u_1, u_2, ..., u_n, U_1, U_2, ..., U_K, w) \in \Re^{n+K+1}$. If (z^*, x^*, y^*, u^*) is an optimal solution, the KKT conditions [14] involve the following equations:

$$\phi_a(z_a^{l*}, C_a^{l*}) = u_a^* \text{ if } z_a^{l*} > 0 \tag{3.19}$$

$$\phi_a(z_a^{l*}, C_a^{l*}) > u_a^* \text{ if } z_a^{l*} = 0 \tag{3.20}$$

$$\sum_a k_{kp}^{al} \pi_{kp}^a u_a^* = U_k^* + \pi_{kp}^l \omega^* \text{ if } x_{kp}^{l_0 *} > 0 \tag{3.21}$$

$$\sum_a k_{kp}^{al} \pi_{kp}^a u_a^* > U_k^* + \pi_{kp}^l \omega^* \text{ if } x_{kp}^{l_0 *} = 0 \tag{3.22}$$

$$\sum_a \theta_{kq}^a u_a^* = -\omega^* \text{ if } y_q^{l*} > 0 \tag{3.23}$$

$$\sum_a \theta_{kq}^a u_a^* > -\omega^* \text{ if } y_q^{l*} = 0 \tag{3.24}$$

Now the optimal paths are shortest paths where the arc costs are given by this equation: $\phi_a(x_a^l, C_a^l) = \dfrac{\gamma_a}{C_a^l - x_a^l} + d_a$. The total cost is considered as the summation of the transit delay cost and capacity installation cost of an arc a. Finally, a feasible solution is optimal when the following conditions are satisfied:

$$\zeta_a = \phi_a(z_a^{lt}, C_a^l) - u_a^t \quad \text{if } z_a^l > 0 \tag{3.25}$$

$$\zeta_a = \min\{\phi_a(z_a^{lt}, C_a^l) - u_a^t; 0\} \text{ if } z_a^l = 0 \tag{3.26}$$

$$\zeta_{kp} = (\textstyle\sum_a k_{kp}^{al} \pi_{kp}^{a} u_a) - U_k - \pi_{kp}^{l} \omega \text{ if } x_{kp}^{l_0} > 0 \tag{3.27}$$

$$\zeta_{kp} = \min\{(\textstyle\sum_a k_{kp}^{al} \pi_{kp}^{a} u_a) - U_k - \pi_{kp}^{l} \omega; 0\} \text{ if } x_{kp}^{l_0} = 0 \tag{3.28}$$

$$\psi_q = (\textstyle\sum_a \theta_{aq}^{l} u_a) + \omega \text{ if } y_q^{l} > 0 \tag{3.29}$$

$$\psi_q = \min\{(\textstyle\sum_a \theta_{aq}^{l} u_a) + \omega; 0\} \text{ if } y_q^{l} = 0 \tag{3.30}$$

Indeed, a feasible solution is optimal if for all a, we have $\zeta_a = 0$, for all k and p, $\zeta_{kp} = 0$ and for all q, $\psi_q = 0$.

$$m_a^t = 1 + \textstyle\sum_{k=1}^{K} \left[\sum_{p=1}^{N_k} k_{kp}^{al} \pi_{kp}^{a} + \sum_{q=1}^{Q_l} \theta_{aq}^{l} \right] \tag{3.31}$$

$$\Lambda_a^t = z_a^{lt} - \textstyle\sum_{k=1}^{K} \left[\sum_{p=1}^{N_k} k_{kp}^{al} \pi_{kp}^{a} x_{kp}^{l_0 t} + \sum_{q=1}^{M_k} \theta_{aq}^{l} y_q^{lt} \right] \tag{3.32}$$

$$Z_k^t = r_k - \textstyle\sum_{p=1}^{N_k} x_{kp}^{l_0 t} \tag{3.33}$$

$$\Delta_l^t = \textstyle\sum_{k=1}^{K} \sum_{p=1}^{N_k} \pi_{kp}^{l} x_{kp}^{l_0 t} - \sum_{q=1}^{M_k} y_q^{lt} \tag{3.34}$$

$$n_l^t = \textstyle\sum_{k=1}^{K} \sum_{p=1}^{N_k} \pi_{kp}^{l} + Q_l \tag{3.35}$$

In Equation (3.31), $(m_a^t - 1)$ represents the total number of paths that are both primary and backup paths through the arc a. In Equation (3.35), n_l^t represents the summation of the total use of the failed arc and the number of backup paths. Equation (3.32) is the aggregated flow constraint, which ensures that the total flow of an arc must be equal to the summation of the flow of primary and backup path uses of that arc. In equation (3.33), Z_k^t is the flow requirement constraint which means, the total flow of all the paths for a commodity must be equal to its bandwidth requirement. Equation (3.34) ensures that the total flow on the failed arc for all the commodities and all paths must be equal to the flow of the backup paths. A solution to the above problem is feasible if $\Lambda_a^t = 0$, $Z_k^t = 0$ and $\Delta_l^t = 0$.

Before describing the link restoration algorithm, it is important to shed some lights on the other important methods that we have used in our proposed algorithm.

3.2.3 Dijkstra Algorithm

The Dijkstra algorithm is used to find the shortest path from a given source node to a given destination node in the network. In our proposed algorithm, as we need to explore the paths for each commodity of a network, we use Dijkstra to find the shortest path for each commodity. In fact we find a new shortest path that is not already explored and use that path if its cost is less than that of the previous explored path. The complexity of this algorithm is $O(|N|^2)$ where N is the number of nodes in the network.

3.2.4 Modified Newton Method

The modified Newton method is an iterative descent method to find a solution to an optimization problem. Typically, this method is a combination of the original Newton method with the steepest descent method [13]. The modified Newton method has several advantages such as ensuring a global convergence as well as fast convergence. Due to its fast converging nature, we have adopted this method in our SALA-based algorithms. The pseudo code of our modified Newton method is as follows:

Algorithm for Modified Newton Method

```
Step 1: Initialize all the parameters as follows:
```

$\varepsilon > 0, \mu_a^0, 0 \le x_a^0 < C_a^0, \alpha \in [0;1], i = 0$

*Step 2: **C**ompute the step size in the modified Newton direction through Armijo method.*

$\alpha_i = Armijo(x_a^i, C_a^i, \mu_a^i, d_M^i)$

Step 3: Calculate the capacity and flow for the next iteration.

$\left(C_a^{i+1}, x_a^{i+1}\right) = \left(C_a^i, x_a^i\right) + \alpha_i d_M^i$

Step 4: If the new estimated capacity and flow can fulfill the stopping condition, we return them as solution. Otherwise we go to step 2 and repeat the process as follows:

$If \left\|\nabla L\left(x_a^{i+1}, C_a^{i+1}, \mu_a^i\right)\right\| < \varepsilon \quad stop$

Else set $\mu_a^{i+1} = \mu_a^i + \alpha\left(x_a^{i+1} - C_a^{i+1}\right), i = i + 1,$ *and then go back to step2.*

Armijo Method

For an optimization problem, after selecting the descent method and the descent direction, another method is needed to search an appropriate step size in the descent direction. The modified Newton method uses the Armijo method for determining the step size of the descent direction. The appropriate choice of the step size ensures that the sequence of the values generated by the descent method converges with some rate of convergence.

The steps of the Armijo method are as follows:

Armijo Method

Step 1: Initialize the values of all the necessary parameters as follows:

$Choose\, \beta \in \left[\frac{1}{10}; \frac{1}{2}\right], (x_a^0, C_a^0), \sigma \in \left[10^{-5}; 10^{-1}\right], m = 0$ *and t= 0*

Step 2: Compute the capacity and flow for the next iteration

$$\left(C_a^{t+1}, x_a^{t+1}\right) = \left(C_a^t, x_a^t\right) + \sigma\beta^m d_M^t$$

Step 3: Compute the step size as follows:

If $L(C_a^t, x_a^t) - L(C_a^{t+1}, x_a^{t+1}) \geq -\sigma\beta^m \nabla L(C_a^t, x_a^t, \mu_a^t) d_M^t$ *Stop and return* β^m

Else $m := m+1$ *and go back to step 2.*

3.3 Proposed Algorithm for Link Restoration Model

The detail algorithm of our proposed link restoration strategy is described as follows:

Proposed Algorithm for Link Restoration Model

---

1. *Initialization*

   Choose $\varepsilon > 0$, u_a^l, U_k^l, $\omega, \lambda_0 > 0$, $\beta > 1$, $z_a^{l_0} = 0$, $x_{k1}^{s_0} = 0$, $y_{k1}^{s0} = 0$, $\Lambda_a^0 = 0$, $m_a^1 = 1$,

   $t = 1$

2. *For all arcs a = 1,....., n, Calculate the optimal flow and capacity of each arc as follows*

   $$(z_a^{lt}, C_a^t) = \arg_{0 \le z_a^{lt} < C_a} \min\left\{ \frac{\gamma_a z_a^{lt}}{C_a^t - z_a^l} + d_a C_a^t + u_a^t z_a^{lt} + \frac{\lambda_t}{2}\left[(z_a^l)^2 - 2 z_a^l (z_a^{l(t-1)} - \frac{\Lambda_a^{t-1}}{m_a^{t-1}}) \right] \right\}$$

   This is done by using the Modified Newton method.

3. *End (For)*

4. *For all k=1,...., K*

   $$\zeta_1 = \min_{\{1 \le p \le N_k / \pi_{kp}^l = 1\}} \sum_{a=1}^{n} k_{kp}^{al} \pi_{kp}^a \phi_a (z_a^l, C_a^l)$$

   Determine ω_1 *the length of the shortest path between O_k and D_k and* $\phi_a(z_a^l, C_a^l)$ *is the cost of an arc.*

   Here, we first calculate the cost of all the explored path for a commodity and pick the path that has minimum cost. We then calculate the cost of the path from all available paths and ω_1 *takes the minimum one among them.*

5. *If* $\omega_1 < \zeta_1$, *then* $N_k = N_k + 1$ *(Introduction of a new working path); i.e. if any new path costs less than the previous explored path cost then we explore a new path.*

6. End (If)

7. While $p \le N_k$ *compute*

$$x_{kp}^{l_0} = \max\left\{0; x_{kp}^{t-1} + \frac{\sum_a k_{kp}^{al}\pi_{kp}^{a}u_a - U_k + \omega\pi_{kp}^{l}}{\lambda_t(1+\pi_{kp}^{l} + \sum_a k_{kp}^{al}\pi_{kp}^{a})} + \frac{\sum_a k_{kp}^{al}\pi_{kp}^{a}\frac{\Lambda_a^{t-1}}{m_a} + \pi_{kp}^{l}\frac{\Lambda_l}{n_l} + \frac{Z_k}{N_k}}{1+\pi_{kp}^{l} + \sum_a k_{kp}^{al}\pi_{kp}^{a}}\right\}$$

This equation represents the flow of all the explored paths.

8. If $x_{kp}^{l_0} = 0,\quad N_k = N_k - 1$ *(Cancellation of a non active working path). In other words, if any explored path does not have any flow we discard that path and the total path number of that specific commodity is reduced by 1.*

9. End (If)

10. Else If $\pi_{kp}^{l} = 1$ *then compute,*

$$\zeta_2 = \min_{1\le q\le Q_l}\sum_{a=1}^{n}\theta_{aq}^{l}\phi(z_a^{l}, C_a^{l})$$

Determine ω_2 *the length of the shortest path between O*$_k$ *and D*$_k$ *where the arc costs are given by* $\phi_a(z_a^{l}, C_a^{l})$.

11. If $\omega_2 < \zeta_2$ *then* $Q_l = Q_l + 1$ *(Introduction of a new backup path); i.e. if any new backup path cost is less than the cost of the previous explored path then a new backup path is explored.*

12. End (If)

13. While $q \le Q_l$ *Compute the flows of all backup paths as follows:*

$$y_q^{lt} = \max\left\{0; y_q^{l(t-1)} + \frac{\sum_a u_a\theta_{aq}^{l} - \omega^{t}}{\lambda_t(1+\sum_a \theta_{aq}^{l})} + \frac{\sum_a \theta_{aq}^{l}\frac{\Lambda_a^{t-1}}{m_a^{t}} - \frac{\Delta_l}{n_l}}{1+\sum_a \theta_{aq}^{l}}\right\}$$

14. If $y_q^{l} = 0$ *then* $Q_l = Q_l - 1$ *(Cancellation of a non active backup path). In other words, if any explored backup path does not have any flow, we discard that path and the total backup path number of that specific commodity is reduced by 1.*

15. End (If)

```
16.  End (While q)
17.  End (Else of step 10)
18.  End (While p)
19. End (For k)
    This k is the closing of step 4. So we repeat all the steps from step 4 to step 18 for all the
    commodities.
20. Compute the values of the feasibility and optimality conditions.
    $F = \max\left\{\max_a \left|\Lambda^t_a\right|, \max_k \left|Z^t_k\right|, \left|\Delta^t_l\right|\right\}$
    $Opt = \max\left\{\max_a \left|\zeta^t_a\right|, \max_{kp} \left|\zeta^t_{kp}\right|, \max_q \left|\psi^t_q\right|\right\}$
21. If $\max\{F, Opt\} < \varepsilon$ then Stop.
22. End (If)
23. Else go to step 24
    Otherwise update some values and repeat the process until the threshold is achieved.
24.  Update formula
    $u^{t+1}_a = u^t_a + \frac{\lambda_t}{m^t_a}\Lambda^t_a$
    $U^{t+1}_k = U^t_k + \frac{\lambda_t}{N_k} Z^t_k$
    $\omega^{t+1} = \omega^t + \frac{\lambda_t}{n^t_l}\Delta^t_l$
    $m^{t+1}_a = 1 + \sum_{k=1}^{K}\left[\sum_{p=1}^{N_k} k^{al}_{kp}\pi^a_{kp} + \sum_{q=1}^{Q_l}\theta^l_{aq}\right]$
    $n^{t+1}_l = \sum_{k=1}^{K}\sum_{p=1}^{N_k}\pi^l_{kp} + Q_l$
    $\lambda_{t+1} = \beta\lambda_t$
    Set $t \leftarrow t+1$ and go back to step 2
```

3.4 Algorithm for Global Reconfiguration Strategy

For comparing our results against the global reconfiguration strategy, we implemented the following algorithm for the global reconfiguration strategy (using the SALA method).

Algorithm for Global Reconfiguration Strategy

Step1: Initialize all the necessary parameter values as follows:

$\varepsilon > 0,\ u_1^1,...,u_n^1,\ \lambda_0,\ \beta,\ x_1^0,...x_n^0,\ C_1^0,...,C_n^0\ x_{11}^{s_0},..,x_{k1}^{s_0},\ \Delta_1^0,..,\Delta_n^0\ \ n_1^1,...,n_n^1,\ t$ *Where* u_1^1 *is the Lagrange multipliers,* λ_0 *is the penalty parameter,* $x_{11}^{s_0},...,x_{k1}^{s_0}$ *are the initial flows of all the working paths of all the commodities,* $x_1^0,...,x_n^0$ *and* $C_1^0,...,C_n^0$ *are the initial flows and capacities of the all arcs,* t *iteration number,* $n_1^1 - 1$ *is the initial total number of paths through the arcs etc.*

Step 2: Calculate the flow and capacity of each arc using the modified Newton method and the Armijo method as follows:

For all arcs a = 1, 2....., n

$$(C_a^{st}, x_a^{st}) = \arg_{0 \le x_a^s < C_a} \min\left\{ \frac{\gamma_a x_a^s}{C_a^s - x_a^s} + d_a C_a^s + u_a^t x_a^s + \frac{\lambda_t}{2}\left[(x_a^s)^2 - 2x_a^s (x_a^{s(t-1)} - \frac{\delta_a^{t-1}}{n_a^t}) \right] \right\}$$

Step 3: End (For) of step 2.

Step 4: Calculate the cost of all the explored paths of a commodity and pick the path which has the minimum cost based on our cost function as follows:

For all k=1,..., K

$$\theta = \min_{p=1,...,N_k} \left\{ \sum_{a=1}^{n} \pi_{kp}^a \phi_a (x_a^{st}, C_a^{st}) \right\}$$

Determine η *, the cost of a new previously unexplored shortest path where the arc costs are equal to* $\phi_a(x_a^{st}, C_a^{st})$.

Step 5: If $\eta < \theta$ *, then introduce a new working path for the commodity.*

$N_k = N_k + 1.$

Step 6: End (If) of step 5.

Step 7: Compute the flows of all paths of the commodity using the following formula:

While $p \le N_k$ *compute*

$$x_{kp}^{st} = \max\left\{0; x_{kp}^{st-1} + \frac{(\sum_{a=1}^{n} \pi_{kp}^{a} u_a^t) - U_k^t}{\lambda_t (1 + \sum_{a=1}^{n} \pi_{kp}^{a})} + \frac{\sum_{a=1}^{n}\left(\pi_{kp}^{a} \frac{\delta_a^{t-1}}{n_a^t}\right) - \frac{\delta_k^{t-1}}{N_k}}{1 + \sum_{a=1}^{n} \pi_{kp}^{a}}\right\}$$

Step 8: If $x_{kp}^{st} = 0 \quad N_k = N_k - 1$ *(Cancellation of a non active working path).*

Step 9: End (If) of step 8.

Step 10: End (While) of step 7.

Step 11: End (For) of step 4.

Step 12: Calculate the values of the parameters δ^t *and* σ^t *for the optimality and feasibility conditions of our strategy according to the following equations:*

$$\delta^t = \max\left\{\max_a \left|\delta_a^t\right|, \max_k \left|\delta_k^t\right|\right\} \text{ and } \sigma^t = \max\left\{\max_a \left|\sigma_a^t\right|, \max_{kp} \left|\sigma_{kp}^t\right|\right\}$$

Step 13: If $\max\{\delta^t, \sigma^t\} < \varepsilon$ *Stop i.e. our algorithm is stopped as our feasibility and optimality conditions are fulfilled and we have achieved the optimal flows and capacities of the arcs.*

Step 14: End (If) of step 13.

Step 15: Else go to step 16.

Step 16: Update the parameter values of our algorithm using the following equations:

$$u_a^{t+1} = u_a^t + \frac{\lambda_t}{m_a^t} \delta_a^t$$

$$U_k^{t+1} = U_k^t + \frac{\lambda_t}{N_k} \delta_k^t$$

$$n_a^{t+1} = 1 + \sum_{k=1}^{K} \sum_{p=1}^{N_k} \pi_{kp}^{a}$$

$$\lambda_{t+1} = \beta \lambda_t$$

```
$t \leftarrow t+1$ and go back to step 2.
```

The algorithm for global reconfiguration is used only to compare the performance of our proposed SALA-based link restoration strategy.

Chapter 4

Numerical Results

In this chapter, we describe the performance of our proposed algorithm under various network topologies, traffic demands, performance metrics and scenarios. There are several input parameters for our proposed algorithm. The main ones are described below.

4.1 Performance Evaluation Parameters

4.1.1 Network Topologies

We consider a network as a graph N(*A, B)* where *A* is the number of nodes and B is the number of arcs. For our experiments, three different scenarios are considered (which are described next). For scenario 1, we consider the following network topologies N(15,132), N(15,142), N(15,152) and N(15,162). For Scenario 2, we consider 5 different networks : N(3,6), N(5,20), N(8,56), N(12,126) and N(15,162). For Scenario 3, we use the network N(15, 152). The graphical representations of these networks are shown in the Appendix section.

4.1.2 Traffic Demands

Traffic demand (bandwidth requirement) is the required traffic to be carried through the network for a given commodity (node pair). It can be symmetric (same bandwidth) or asymmetric (different bandwidth per direction). For symmetric traffic demands, we set a fixed value (5 bits/second) for all commodities. For asymmetric traffic, we generate the traffic demands for each commodity as follows:

Code for Generating Asymmetric Traffic Demands

```
double randdouble( double min,  double max)
        {
         double result;
         if (min>max)
                 result = max + (rand()/((( double)RAND_MAX + 1) / min));
        else result = min + (rand()/((( double)RAND_MAX + 1) / max));
        return result;
        }
```

For each of the commodities in the network, the above code generates a random number within the range min and max, where min and max are predefined values.

4.1.3 Scenarios

The following scenarios are considered:

- **Scenario 1:**

We vary the network connectivity, i.e. the number of the arcs of the networks is changed and the impact of this variation is studied using both symmetric and asymmetric traffic demands, for the considered performance metrics.

- **Scenario 2:**

We vary the network size, that is the numbers of the nodes, and we study the impact of this variation using both symmetric and asymmetric traffic demands, for the considered performance metrics.

- **Scenario 3:**

We keep the network size fixed and vary the traffic demand and see the impact of this variation for all the considered performance metrics.

4.1.4 Performance Metrics

We use five performance metrics to evaluate our numerical results.

- **Capacity Installation Cost**

The first performance metric is the capacity installation cost. As our target of this restoration strategy is to minimize the network cost, one of the costs is the capacity installation cost. We calculate this cost simply by multiplying the capacity and the unit cost for all the arcs of the network and then take the summation of all those.

- **Routing Cost**

Routing cost is our second performance metric . In our algorithm, we consider the routing cost and try to minimize it. To calculate the routing cost, we use the Kleinrock's average delay function [14].

- **Total Network Cost**

We consider the total network cost of a network as the summation of the capacity installation cost and the routing cost.

- **Total Required Capacity**

The main purpose of this research is to optimize the capacity and flow of the network in order to minimize the network cost. Therefore, capacity is the very important metric to be considered.

- **CPU Time**

Time is an important factor for restoration. We calculate the required CPU time to get the solution and consider this as a performance metric. Basically, it depends on the processor that is used. In this thesis, we use an Intel 1.86 GHz processor with 1 GB memory to run our proposed link restoration algorithm.

The parameters used for our algorithms are captured in Table 4.1.

Table 4.1: Parameters used for our proposed algorithm

Restoration Scheme	Link Restoration
Failure Scenario	Single link failure
Maximum paths per node pair	10
Maximum number of hops for a path	10
Capacity Modularization	No capacity modularization is made to conform to physical transmission system
Link Orientation	Bidirectional link (a link is made by 2 unidirectional arcs)
CPU Configuration	Processor : Intel Xenon 1.86 GHz Memory: 1 GB
Initial Parameters value for the SALA based Link restoration algorithm	$\gamma_a = 0.001$(transit delay unit cost), $d_a = 0.001$ (unit capacity installation cost), $u_a = 0.01374$ (Lagrange multiplier), $\beta = 2.7$, $U_k = 0.001659$ (Lagrange multiplier), $w = 0.006$, $\lambda = 0.01092$ (penalty parameter)
Performance Metric	Capacity Installation cost, Routing cost, Total Network Cost, Total Required Capacity, CPU Time

Traffic Demand	Symmetric and Asymmetric
Networks	*N*(3,6), *N*(5, 20), *N*(8, 56), *N*(12, 126),*N*(15, 132), *N*(15, 142), *N(*15, 152), *N*(15, 162)
Routing strategy	All possible paths

4.2 Performance Comparison of Link and Global Reconfiguration Strategies

In this section, the performance analysis and comparison of the link restoration and global reconfiguration strategies under the above mentioned scenarios are described.

4.2.1 Results for Scenario 1

4.2.1.1 Capacity Installation Cost

For comparing the performance metric capacity installation cost between the link and global reconfiguration strategies, the following networks N(15,132), N(15,142), N(15,152), N(15,162) are considered.

The results obtained in terms of capacity installation cost using symmetric traffic demands are captured in Table 4.2 and depicted in Figure 4.1.

Table 4.2: Capacity Installation Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

Networks	N(15,132)	N(15,142)	N(15,152)	N(15,162)
Link	4.399	4.239	4.075	3.995
Global	1.841	1.774	1.702	1.623

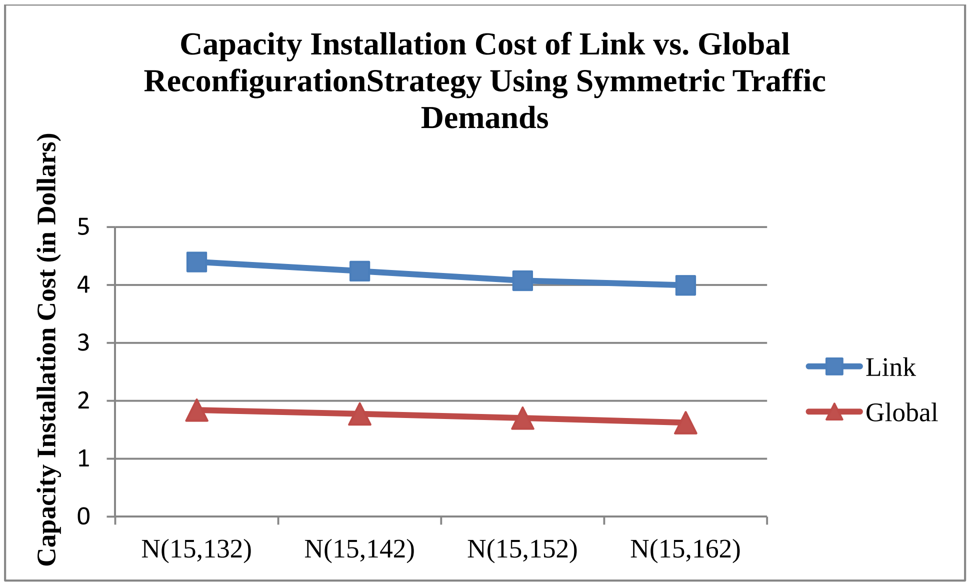


Figure 4.1: Capacity Installation Cost (in Dollars) for Link vs. Global Restoration Strategy using symmetric traffic demands

Figure 4.1 shows that as the number of arcs increases in a network, the capacity installation cost decreases. Figure 4.1 reveals the fact that in networks having large number of arcs, have better options for selecting candidate paths for each commodity than in networks having fewer arcs when a failure occurs in the network, and consequently networks with large number of arcs require less capacity installation cost.

From Figure 4.1, it can also be observed that the global reconfiguration strategy requires less capacity than the link restoration strategy. This is understandable because link restoration is less flexible in selecting restoration paths when failure occurs in the network and thus cannot better share the capacity of the network.

Like symmetric traffic demands, we have also compared the restoration strategies with respect to capacity installation cost under asymmetric traffic demands. Table 4.3 summarizes the results and Figure 4.2 depicts them.

Table 4.3: Capacity Installation Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

Networks	N(15,132)	N(15,142)	N(15,152)	N(15,162)
Link	1.745	1.685	1.624	1.596
Global	0.751	0.725	0.696	0.668

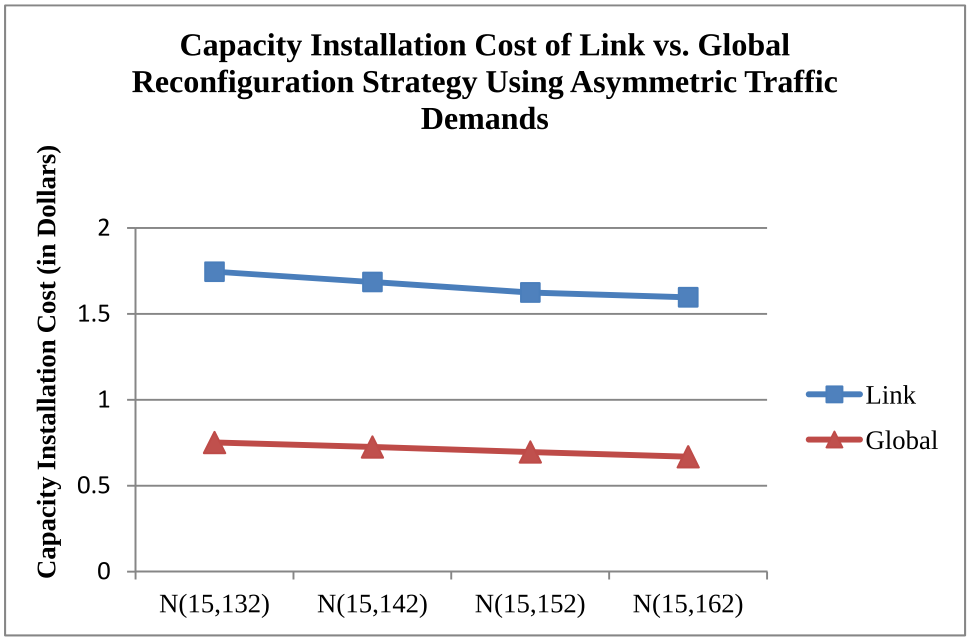


Figure 4.2: Capacity Installation Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

In asymmetric traffic demands, the performance metric capacity installation cost behaves in the same way as it does in symmetric traffic demands. Global reconfiguration strategy performs better than link restoration strategy because it has more flexibility in its optimization procedure.

4.2.1.2 Total Used Capacity

The results obtained when evaluating the performance metric total used capacity for the restoration strategies are summarized in Table 4.4 and depicted in Figure 4.3.

Table 4.4: Total Used Capacity (in Bits) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

Networks	N(15,132)	N(15,142)	N(15,152)	N(15,162)
Link	4399.600	4239.892	4075.670	3995.660
Global	1841.420	1774.161	1702.800	1623.620

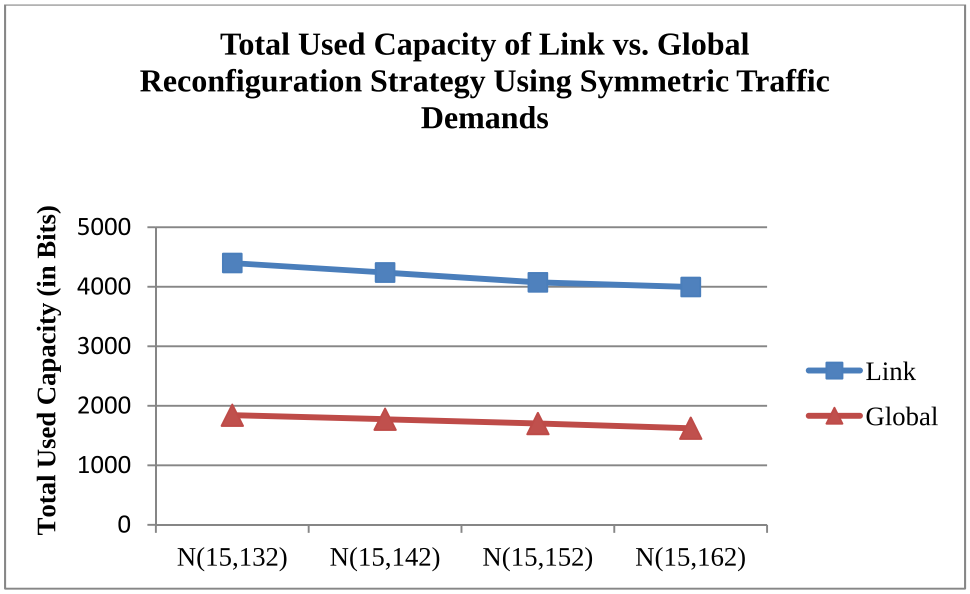


Figure 4.3: Total Used Capacity (in Bits) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

It's obvious that, the more flexible the optimization procedure is, the less capacity is required to optimize the network when a failure occurs. Figure 4.3 reveals this phenomenon for global reconfiguration and link restoration strategies. As the global reconfiguration strategy is more flexible for selecting candidate paths for the commodities, it can better share the network capacity and as a result, it requires less capacity than the link restoration strategy while designing the self-healing ATM network.

We repeat the same experiment using asymmetric traffic demands. The results for the total used capacity are captured in Table 4.5 and depicted in Figure 4.4.

Table 4.5: Total Used Capacity (in Bits) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

Networks	N(15,132)	N(15,142)	N(15,152)	N(15,162)
Link	1745.890	1685.890	1624.880	1596.920
Global	751.606	725.361	696.066	668.390

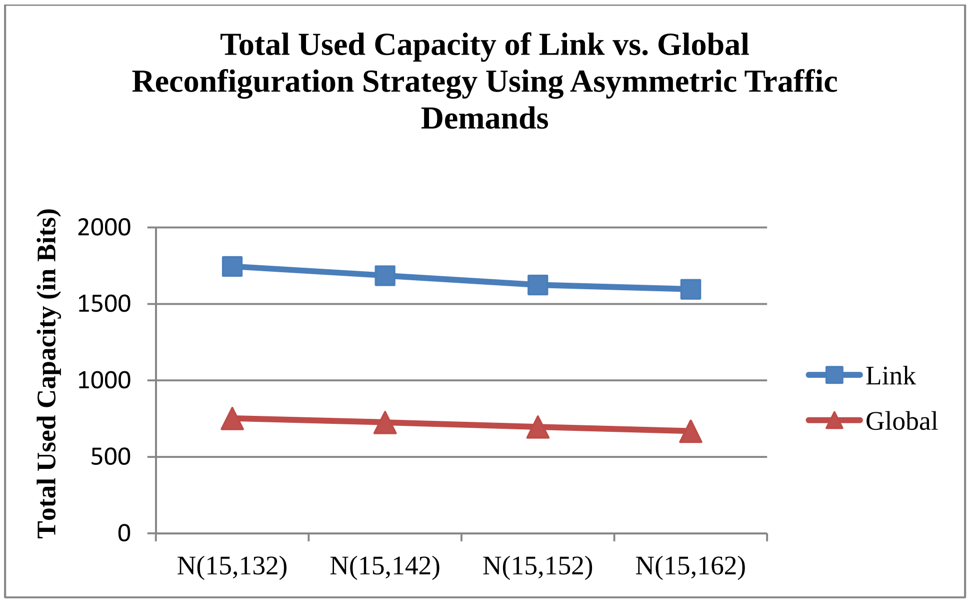


Figure 4.4: Total Used Capacity (in Bits) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

In Figure 4.4, it can be observed that for asymmetric traffic demands, the performance metric total used capacity behaves in the same way as it does in symmetric traffic demands.

4.2.1.3 Routing Cost

We have evaluated the metric routing cost for comparing the restoration strategies and results are captured in Table 4.6 and depicted in Figure 4.5.

Table 4.6: Routing Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

Networks	N(15,132)	N(15,142)	N(15,152)	N(15,162)
Link	0.069	0.073	0.078	0.083
Global	0.643	0.695	0.749	0.783

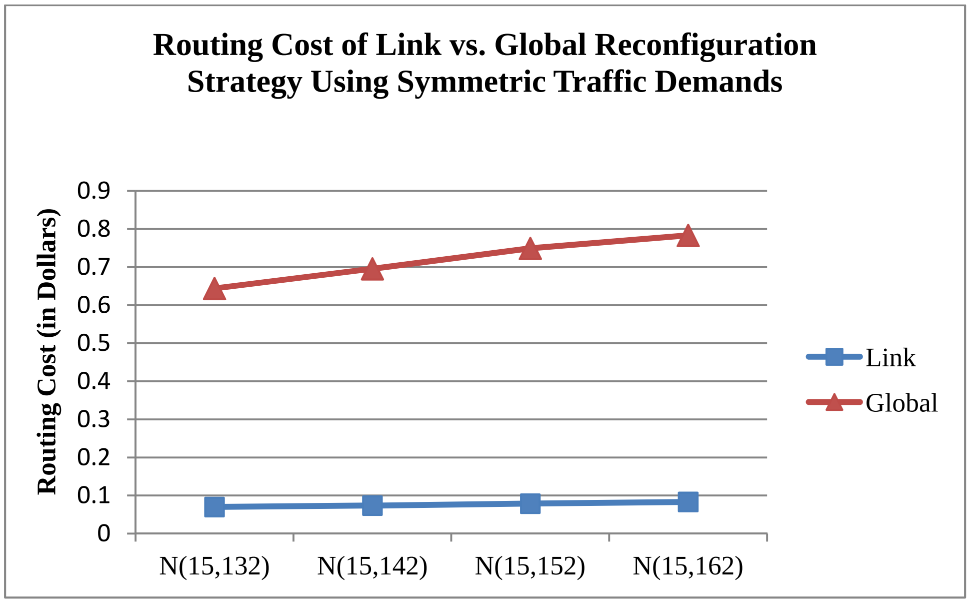


Figure 4.5: Routing Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

Figure 4.5 depicts the fact that with the increase in the number of arcs in a network, better options for choosing the candidate routes for optimizing the network cost exist, but on the other hand routing cost is increased as more arcs are used for routing the traffic flows.

From Figure 4.5, it can also be observed that the link restoration strategy requires less routing cost than global reconfiguration strategy. This can be justified by the fact that the global reconfiguration strategy reroutes both the affected and unaffected traffic in case of link failure while the link restoration reroutes only the affected traffic flows keeping the unaffected traffics unchanged. As a result, global reconfiguration strategy can better share the network capacity, but at the expense of more traffic routing than in the case of link restoration strategy.

We now repeat the same experiment but using asymmetric traffic demands. The results are captured in Table 4.7 and Figure 4.6.

Table 4.7: Routing Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

Networks	N(15,132)	N(15,142)	N(15,152)	N(15,162)
Link	0.068	0.072	0.076	0.083
Global	0.499	0.526	0.551	0.567

From Table 4.7 it can be observed that, routing cost of global reconfiguration in asymmetric traffic is much higher than that of the link restoration strategy, which is exactly the same as in symmetric traffic demands.

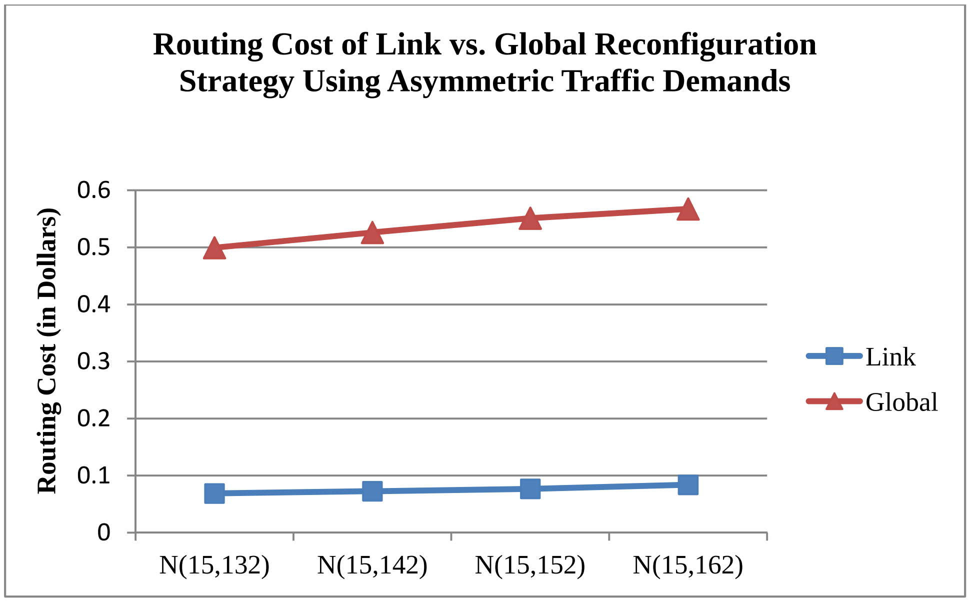


Figure 4.6: Routing Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

In Figure 4.6, it is clearly observed that the same trend as in the case of symmetric traffic demands prevails.

4.2.1.4 Total Network Cost

The total network cost for link and global reconfiguration strategies under symmetric traffic demands are captured in Table 4.8 and depict in Figure 4.7.

Table 4.8: Total Network Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

Networks	N(15,132)	N(15,142)	N(15,152)	N(15,162)
Link	4.469	4.313	4.154	4.078
Global	2.485	2.469	2.452	2.406

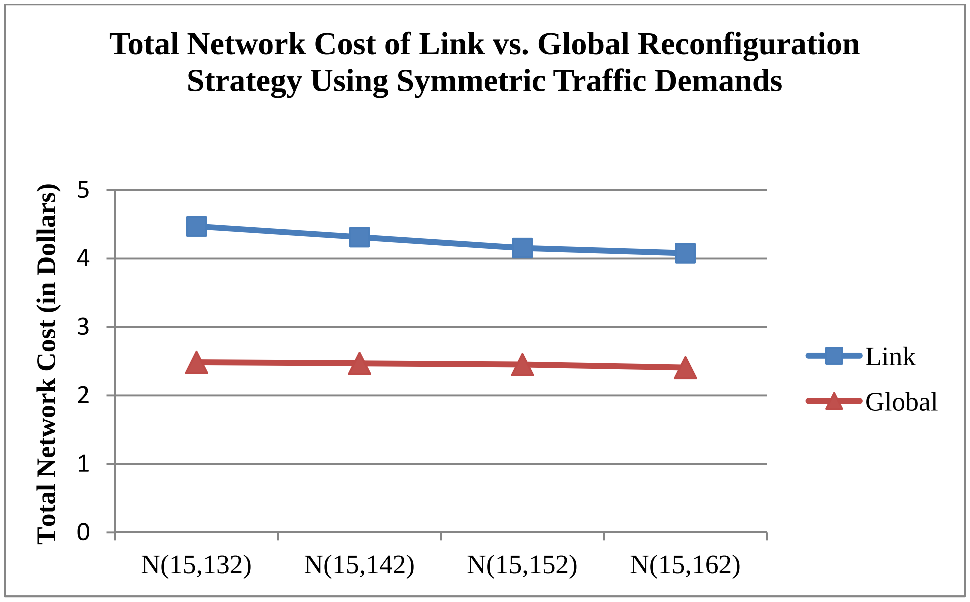


Figure 4.7: Total Network Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

Figure 4.7 shows that when the number of arcs of a network is gradually increased, the total network cost is gradually decreased for all the restoration strategies. This is justified by the fact that capacity installation cost has larger value than the routing cost in a specific network. So, the trend of total network cost depends on the trend of capacity installation cost. We already have seen that, with the increase of the number of arcs, capacity installation cost decreases and as a result the total network cost is also decreased for both restoration strategies.

In Figure 4.7, it can also be observed that the global reconfiguration strategy requires less network cost to optimize the network than the link restoration strategy. This can be justified by the fact that the optimization procedure for global reconfiguration is more flexible in selecting the candidate routes for the commodities, and hence the capacity of the network is shared better, thus less network cost is needed for optimizing the network.

Like symmetric traffic demands, we have also compared the restoration strategies with respect to total network cost under asymmetric traffic demands. Table 4.9 summarizes the results and Figure 4.8 depicts the results.

Table 4.9: Total Network Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

Networks	N(15,132)	N(15,142)	N(15,152)	N(15,162)
Link	1.814	1.758	1.701	1.680
Global	1.251	1.250	1.247	1.235

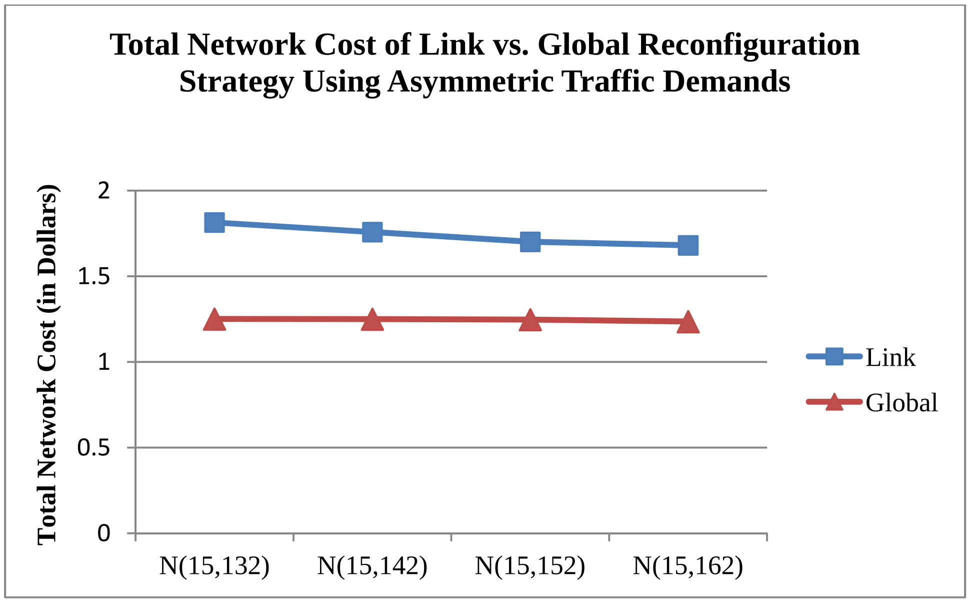


Figure 4.8: Total Network Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

Figure 4.8 reveals that, in case of asymmetric traffic demands, total network costs in both restoration strategies are also dominated by capacity installation costs, consequently the total network costs are decreased when the number of arcs in a network is increased.

Likewise symmetric traffic demands, the global reconfiguration strategy requires less network cost to optimize the network than the link restoration strategy.

4.2.1.5 CPU Time

With respect to CPU time, we have compared the link and global reconfiguration strategy. The results are captured in Table 4.10 and depicted in Figure 4.9.

Table 4.10: CPU Time (in Seconds) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

Networks	N(15,132)	N(15,142)	N(15,152)	N(15,162)
Link	0.750	0.810	0.830	1.040
Global	0.220	0.220	0.260	0.360

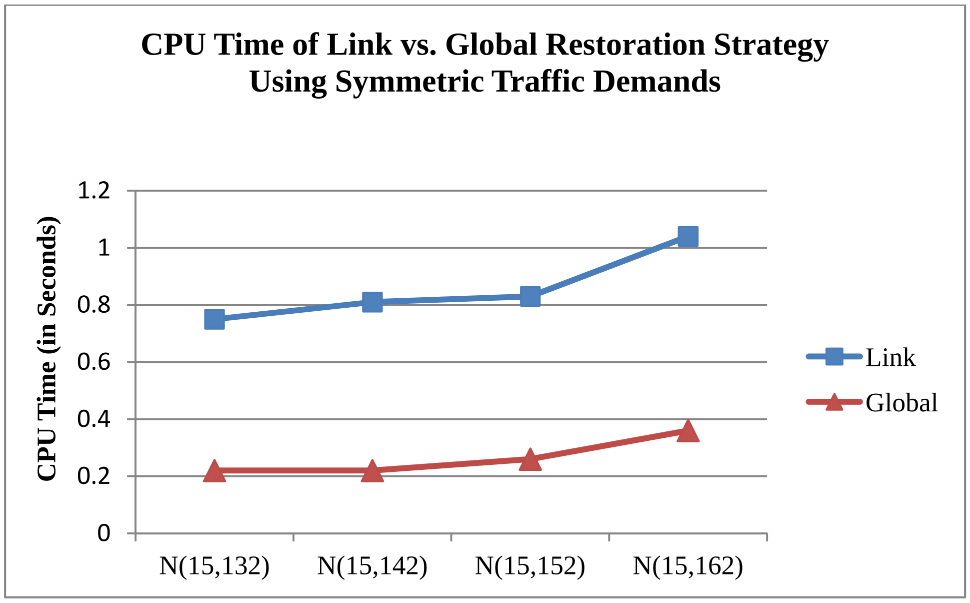


Figure 4.9: CPU Time (in Seconds) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

Figure 4.9 shows that the proposed restoration strategies are quite fast as the link restoration strategy can execute only within 1.04 seconds for restoring the affected traffics of a large dense network and global reconfiguration strategy requires less time than the link restoration.

We repeat the same experiment using asymmetric traffic demands. The results are captured in Table 4.11 and depict in Figure 4.10.

Table 4.11: CPU Time (in Seconds) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

Networks	N(15,132)	N(15,142)	N(15,152)	N(15,162)
Link	0.750	0.810	0.830	1.040
Global	0.220	0.220	0.260	0.360

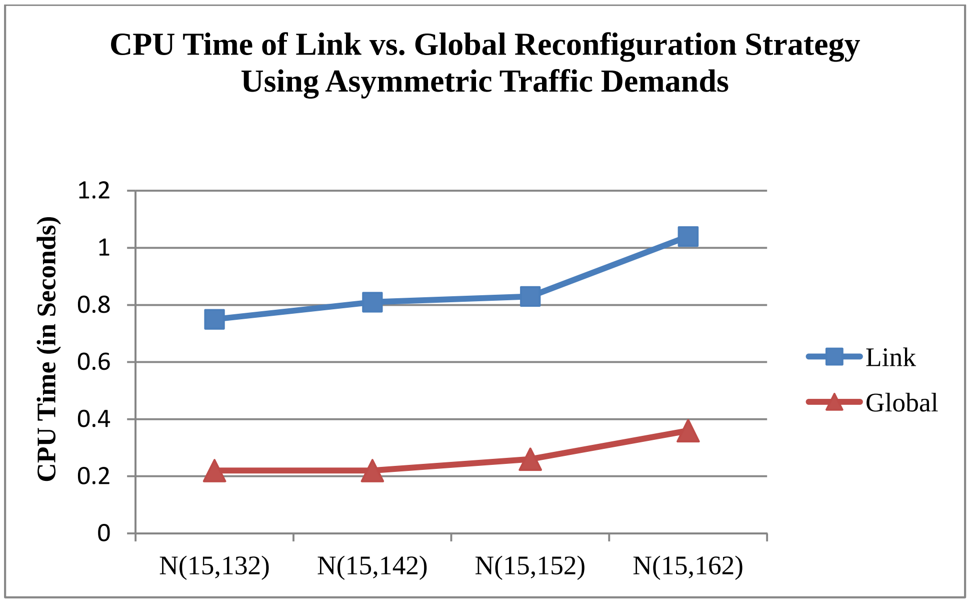


Figure 4.10: CPU Time (in Seconds) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

Figure 4.10 shows that even for asymmetric traffic, our proposed restoration strategy is quite fast. Here also, global reconfiguration strategy requires less time than the link restoration.

4.2.2 Results for Scenario 2

In scenario 2, the performance comparison between the link restoration and global reconfiguration strategy has been done in 5 different sizes of networks: N(3, 6), N(5, 20), N(8, 56), N(12, 126), N(15, 162), using both symmetric and asymmetric traffic demands.

4.2.2.1 Capacity Installation Cost

The results for comparing the two restoration strategies in terms of capacity installation cost using symmetric traffic demands are summarized in Table 4.12 and depicted in Figure 4.11.

Table 4.12: Capacity Installation Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

Networks	N (3,6)	N(5,20)	N(8,56)	N(12,126)	N(15,162)
Link	0.145	0.359	0.895	2.136	3.995
Global	0.050	0.134	0.350	0.851	1.623

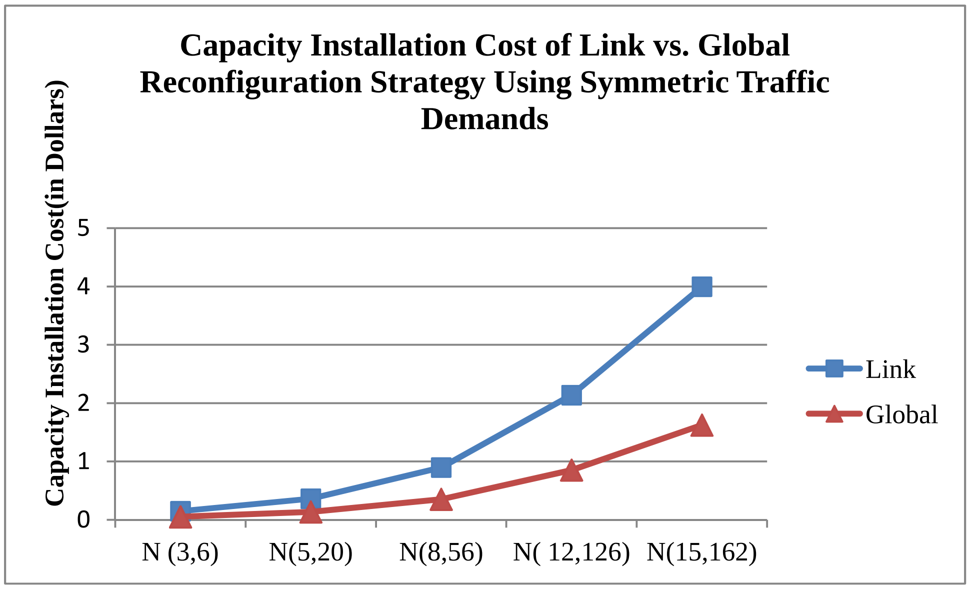


Figure 4.11: Capacity Installation Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

Figure 4.11 shows that as the size of the networks increases, so does the capacity installation cost. This is due to the fact that when the network size is increased, the number of commodities and their traffic demands are also increased. To fulfill the increased traffic demands, more capacity is needed for the network, which in turn increases the capacity installation cost.

In Figure 4.11, it is also observed that the link restoration strategy requires more capacity installation cost than the global restoration strategy. This is justified by the fact that the optimization procedure of link restoration is less flexible in selecting the backup paths for a commodity when a single link failure occurs in the network. Moreover, the difference between the global reconfiguration and link restoration strategy increases in respect of capacity installation cost when the network size increases.

We now repeat the experiment using asymmetric traffic demands. The results are captured in Table 4.13 and Figure 4.12. Clearly, the same observations as in the case of symmetric traffic demands prevail.

Table 4.13: Capacity Installation Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

Networks	N (3,6)	N(5,20)	N(8,56)	N(12,126)	N(15,162)
Link	0.051	0.132	0.349	0.841	1.576
Global	0.002	0.057	0.154	0.369	0.668

In Table 4.13, it is also observed that the link restoration strategy requires more capacity installation cost than the global restoration strategy using asymmetric traffic demands which is exactly same as using symmetric traffic demands.

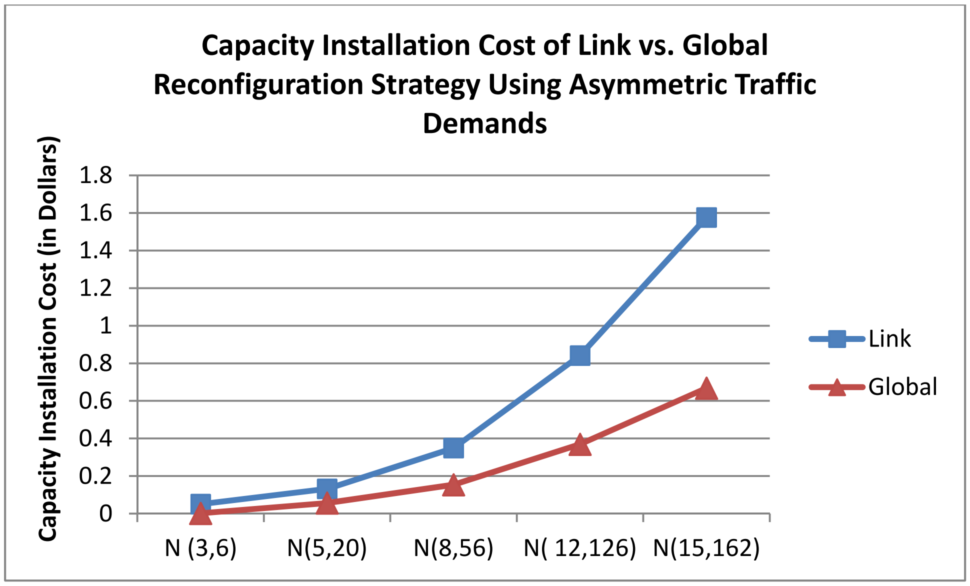


Figure 4.12: Capacity Installation Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

4.2.2.2 Total Used Capacity

The results obtained when evaluating the performance metric total used capacity for the restoration strategies are summarized in Table 4.14 and depicted in Figure 4.13.

Table 4.14: Total Used Capacity (in Bits) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

Networks	N (3,6)	N(5,20)	N(8,56)	N(12,126)	N(15,162)
Link	145.650	359.650	895.690	2136.820	3995.660
Global	50.124	134.162	350.260	851.671	1623.616

In Table 4.14, it is clearly seen that link restoration strategy requires more capacity than the global reconfiguration strategy. In Figure 4.13, the expected results are obtained for link restoration strategy because with the increase in network size, the number of commodities increases, and thus the required capacity also increases. In addition, the optimization procedure

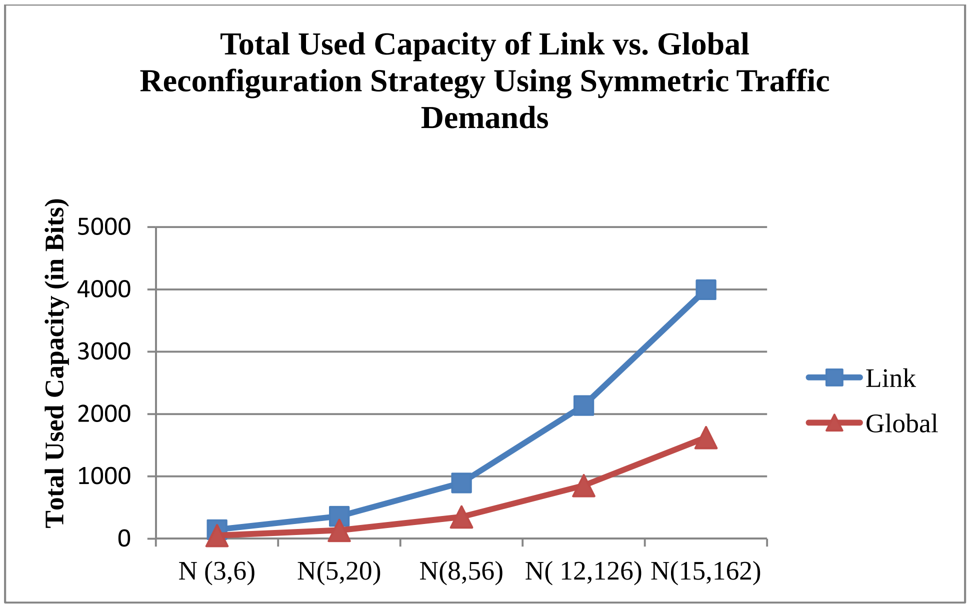


Figure 4.13: Total Used Capacity (in Bits) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

for link restoration is less flexible in selecting the candidate routes compared to the case of global reconfiguration. In Figure 4.13, it can also be observed that the difference between link and global reconfiguration is quite small in small size networks compared to large size networks.

We repeat the same experiment using asymmetric traffic demands and the results are captured in Table 4.15 and depicted in Figure 4.14.

Table 4.15: Total Used Capacity (in Bits) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

Networks	N (3,6)	N(5,20)	N(8,56)	N(12,126)	N(15,162)
Link	51	132	349	841	1576
Global	2	57	154	369	668

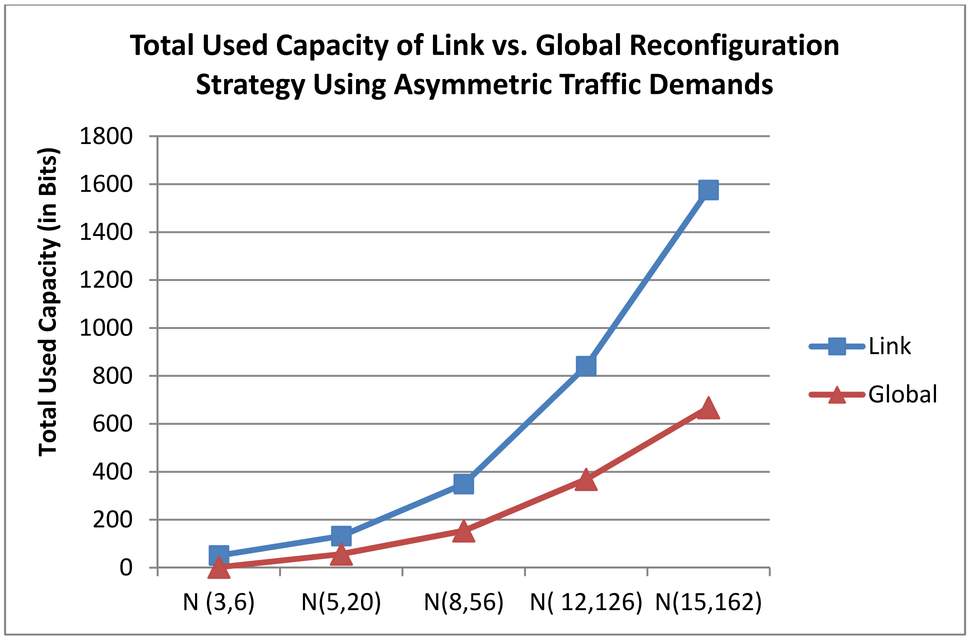


Figure 4.14: Total Used Capacity (in Bits) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

In Figure 4.14, it can be observed that in terms of total used capacity, the global reconfiguration strategy performs better than the link restoration strategy for asymmetric traffic demands.

4.2.2.3 Routing Cost

We have evaluated the metric routing cost for comparing the restoration strategies and the comparison results are captured in Table 4.16 and depicted in Figure 4.15.

Table 4.16: Routing Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

Networks	N (3,6)	N(5,20)	N(8,56)	N(12,126)	N(15,162)
Link	0.003	0.012	0.029	0.065	0.084
Global	0.015	0.085	0.265	0.602	0.783

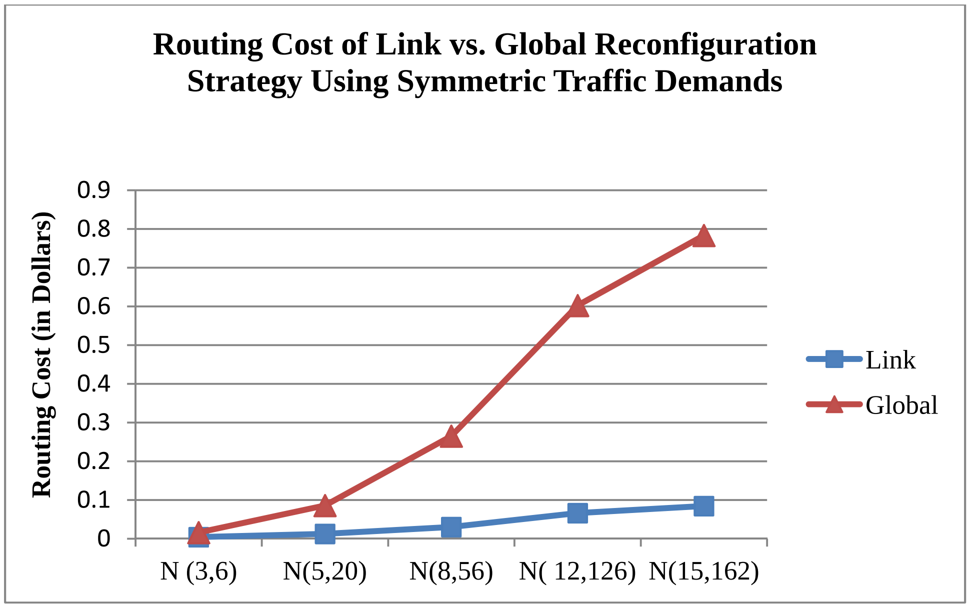


Figure 4.15: Routing Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

Figure 4.15 reveals that the routing cost is increasing along with the increment in network size for both restoration strategies. It can also be observed that the link restoration strategy in terms of routing cost, we can see that link restoration strategy performs better than the global reconfiguration strategy. This is due to the fact that the link restoration strategy reroutes only the affected traffic flows when a link failure occurs whereas in the global reconfiguration strategy all the traffic flows (affected and unaffected) are rerouted throughout the network. It is also observed that the difference between global and link restoration strategy in terms of routing cost is more pronounced in large size networks and less pronounced in small size networks. We also evaluate the routing cost with asymmetric traffic demands. The results are captured in Table 4.17 and depict in Figure 4.16.

Table 4.17: Routing Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

Networks	N (3,6)	N(5,20)	N(8,56)	N(12,126)	N(15,162)
Link	0.003	0.010	0.028	0.062	0.078
Global	0.014	0.055	0.160	0.382	0.567

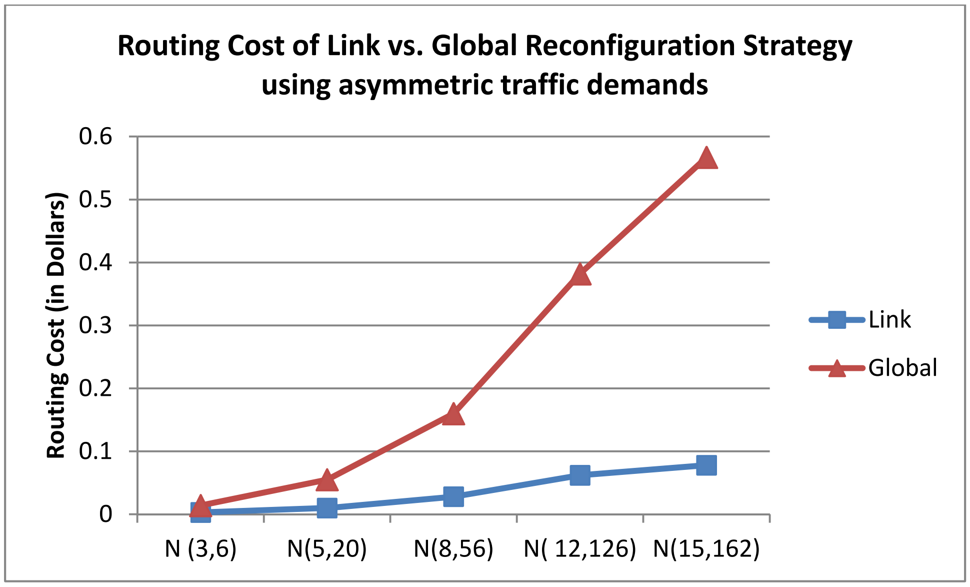


Figure 4.16: Routing Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

Likewise symmetric traffic demands, link restoration strategy has less routing cost than the global reconfiguration strategy and this difference increases with the increment of network size for asymmetric traffic demands.

4.2.2.4 Total Network Cost

We have evaluated the comparison for link and global reconfiguration strategy for the performance metric, total network cost. The results are captured in table 4.18 and depicted in figure 4.17.

Table 4.18: Total Network Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

Networks	N (3,6)	N(5,20)	N(8,56)	N(12,126)	N(15,162)
Link	0.149	0.371	0.925	2.202	4.079
Global	0.065	0.219	0.615	1.453	2.406

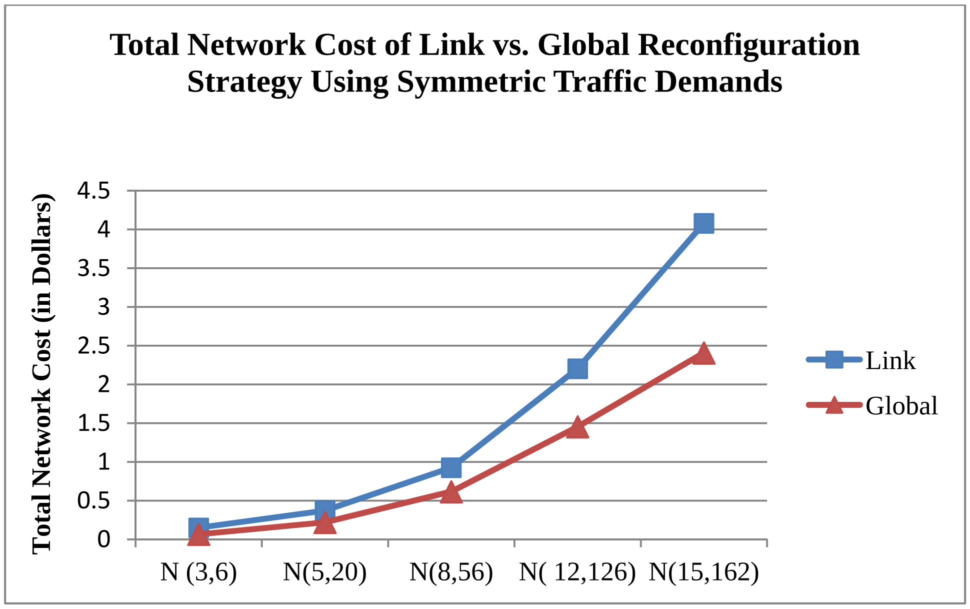


Figure 4.17: Total Network Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

Figure 4.17 reveals that if the network size is increased, the total network cost of the network is also increased for both restoration strategies. This is quite understandable since link restoration generates more capacity installation cost than global reconfiguration.

We now repeat the same experiment using asymmetric traffic demands. The results that are captured in Table 4.19 and Figure 4.18 clearly show the same observations as in the case of symmetric traffic demands.

Table 4.19: Total Network Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

Networks	N (3,6)	N(5,20)	N(8,56)	N(12,126)	N(15,162)
Link	0.054	0.142	0.377	0.904	1.654
Global	0.035	0.113	0.315	0.751	1.230

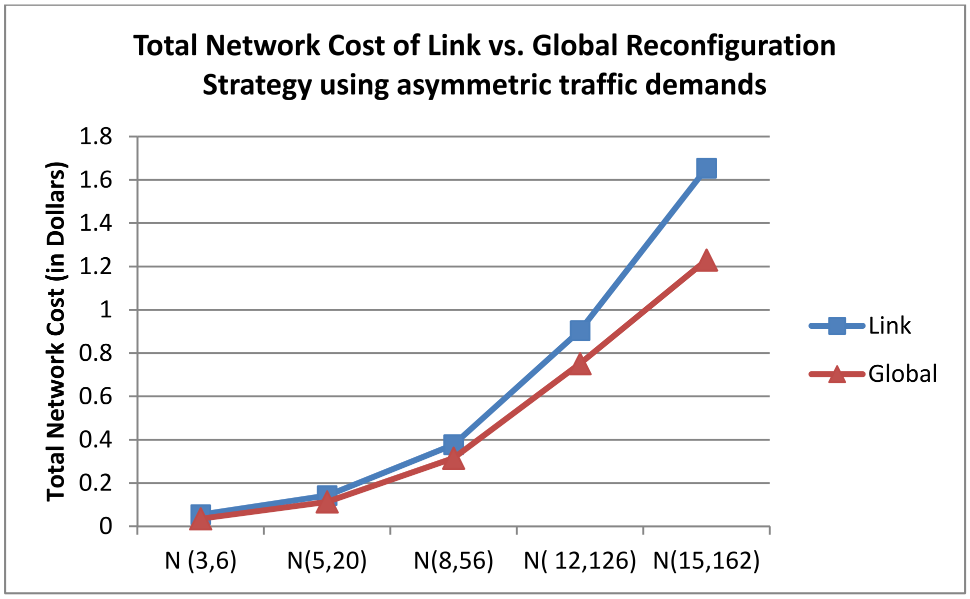


Figure 4.18: Total Network Cost (in Dollars) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

4.2.2.5 CPU Time

With respect to CPU time, we have compared the link and global reconfiguration strategy and the results are captured in Table 4.20 and depicted in Figure 4.19.

Table 4.20: CPU Time (in Seconds) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

Networks	N (3,6)	N(5,20)	N(8,56)	N(12,126)	N(15,162)
Link	0.010	0.040	0.170	0.500	1.040
Global	0.001	0.010	0.020	0.110	0.360

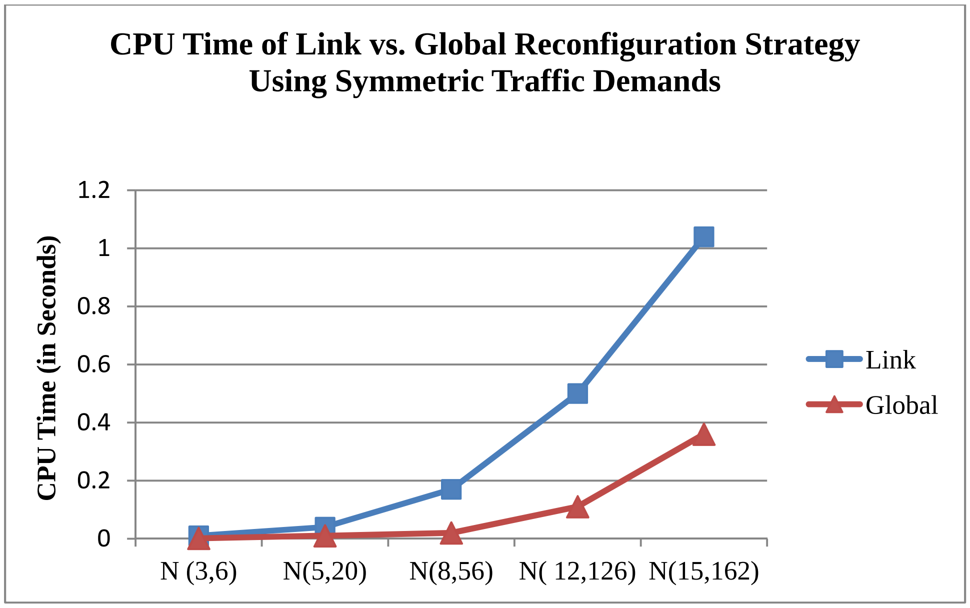


Figure 4.19: CPU Time (in Seconds) for Link vs. Global Reconfiguration Strategy using symmetric traffic demands

Figure 4.19 reveals that our proposed link restoration strategy is quite fast as the convergence time is 1.04 seconds only for restoring the affected traffics of the large size network N(15,162). In addition, link restoration requires more time than global reconfiguration strategy.

We also evaluate CPU time using asymmetric traffic demands and the results are captured in Table 4.21 and depicted in Figure 4.20.

Table 4.21: CPU Time (in Seconds) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

Networks	N (3,6)	N(5,20)	N(8,56)	N(12,126)	N(15,162)
Link	0.010	0.020	0.100	0.350	0.350
Global	0.001	0.005	0.020	0.110	0.190

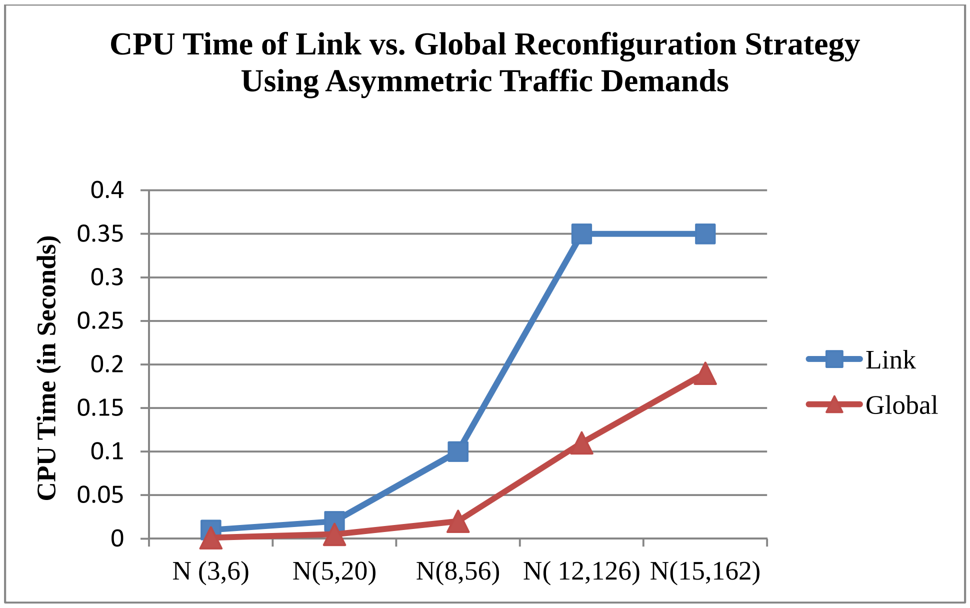


Figure 4.20: CPU Time (in Seconds) for Link vs. Global Reconfiguration Strategy using asymmetric traffic demands

Figure 4.20 shows the required time for executing our proposed algorithm using asymmetric traffic demands. Here also, we have found that the execution time for the large size network N(15,162) is only 0.36 seconds for the link restoration algorithm. Global reconfiguration needs less time to execute than link restoration.

4.2.3 Results for Scenario 3

In this section, the performance comparison between the link and global restoration strategies is done in terms of the same performance metrics discussed earlier, where the traffic demands of the commodities are varied in the network N(15,152).

4.2.3.1 Capacity Installation Cost

We have varied the traffic demands of the commodities here. The results are captured in the Table 4.22 and depicted in Figure 4.21.

Table 4.22: Capacity Installation Cost (in Dollars) for Link vs. Global Reconfiguration Strategy for different traffic demands

Traffic Demands	0.1	0.2	1	2	5	10
Link	0.171	0.213	0.812	1.632	4.065	8.130
Global	0.038	0.076	0.358	0.696	1.702	3.206

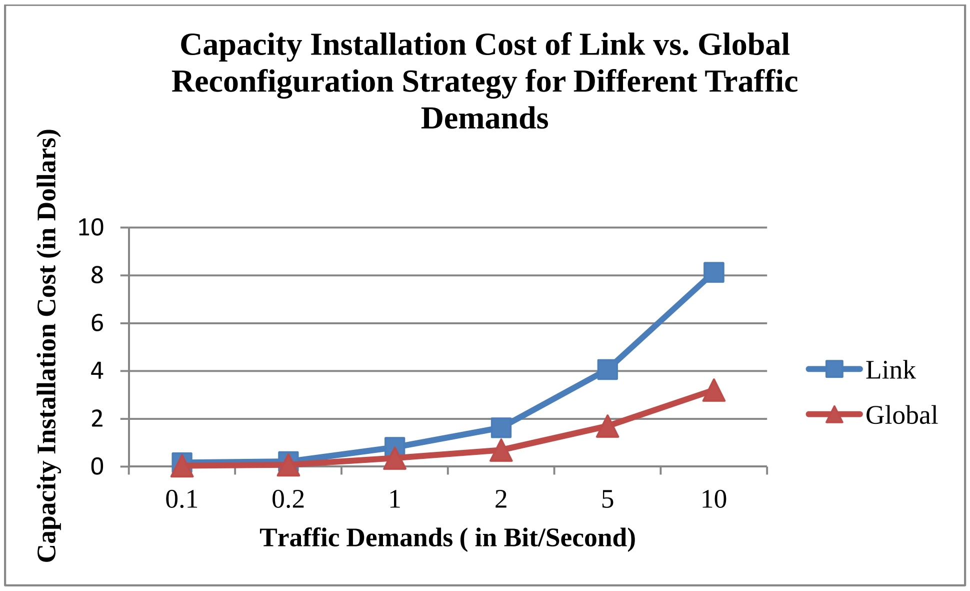


Figure 4.21: Capacity Installation Cost (in Dollars) for Link vs. Global Reconfiguration Strategy for different traffic demands

Figure 4.21 shows that as the traffic demands increase, the capacity installation cost also increases for both restoration strategies. It is also observed that global reconfiguration strategy requires less capacity installation cost compared to the link restoration scheme. This can be due to the fact that the optimization process in the link restoration strategy is less flexible in selecting the routes, thus cannot share the capacity of the network in a better way, and thus results in more capacity installation cost than in the case of global reconfiguration strategy. It is also observed that the difference in capacity installation cost between link restoration and global reconfiguration strategies increases along with the increment in bandwidth requirements of the commodities.

4.2.3.2 Total Used Capacity

The results obtained when evaluating the performance metric total used capacity for the restoration strategies are summarized in Table 4.14 and depicted in Figure 4.13.

Table 4.23: Total Used Capacity (in Bits) for Link vs. Global Reconfiguration Strategy for different traffic demands

Traffic Demands	0.1	0.2	1	2	5	10
Link	171.694	213.603	812.560	1632.970	4065.670	8130.760
Global	38.932	76.279	358.318	696.066	1702.800	3206.830

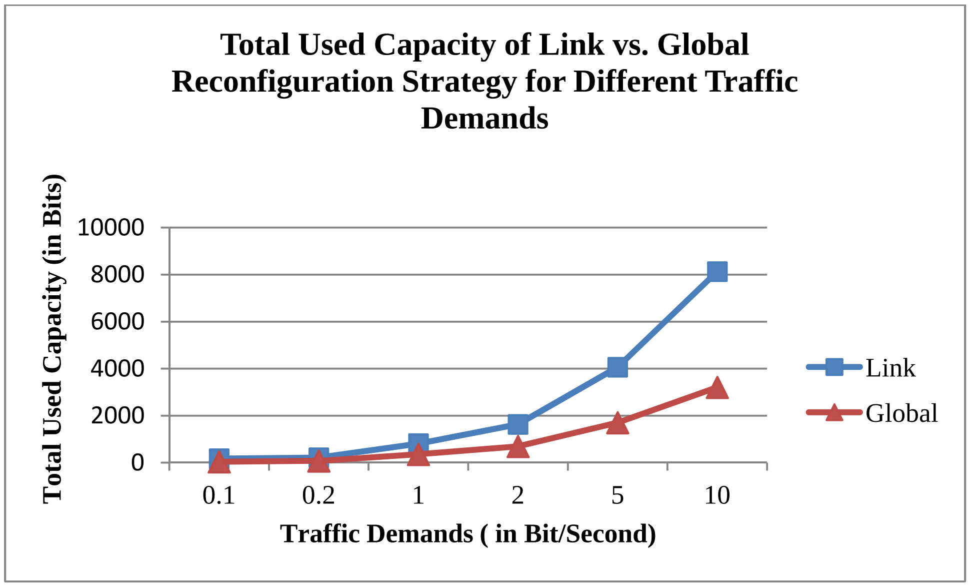


Figure 4.22: Total Used Capacity (in Bits) for Link vs. Global Reconfiguration Strategy for different traffic demands

Figure 4.22 also reveals that the global reconfiguration strategy requires less capacity to optimize the network than the link restoration strategy. This is due to the fact that the global reconfiguration is flexible for better sharing the capacity of the network compared to the link restoration strategy. Moreover, the difference between global reconfiguration and link restoration with respect to the required capacity increases when the traffic demands of the commodities increase.

4.2.3.3 Routing Cost

We have evaluated the metric routing cost for comparing the restoration strategies and the comparison results are captured in Table 4.24 and depicted in Figure 4.23.

Table 4.24: Routing Cost (in Dollars) for Link vs. Global Reconfiguration Strategy for different traffic demands

Traffic Demands	0.1	0.2	1	2	5	10
Link	0.028491	0.046749	0.07769	0.07969	0.08269	0.08469
Global	0.11	0.16	0.4537	0.55117	0.74926	0.93117

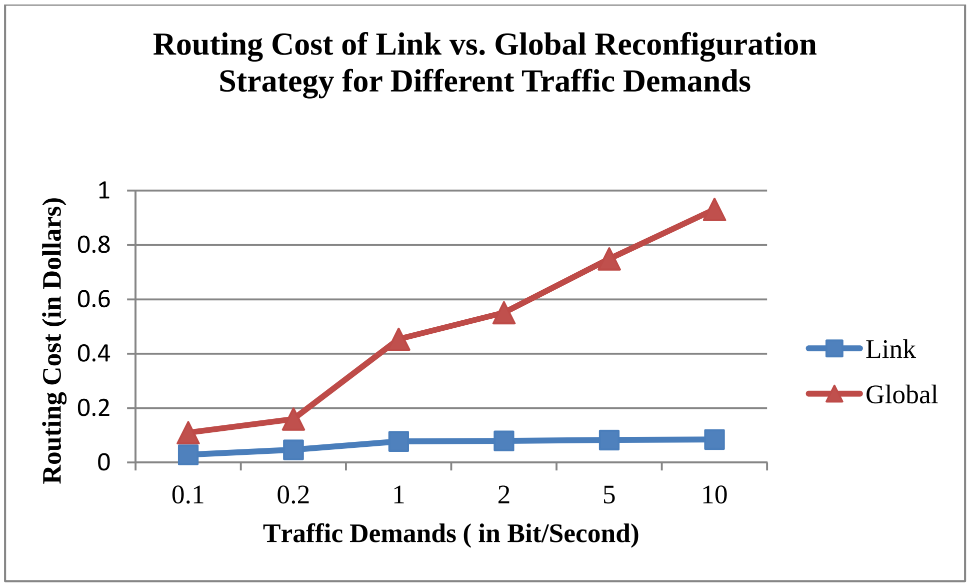


Figure 4.23: Routing Cost (in Dollars) for Link vs. Global Reconfiguration Strategy for different traffic demands

Figure 4.23 shows that the link restoration strategy requires less routing cost than the global reconfiguration strategy. This is essentially due to the fact that the link restoration strategy routes the affected traffics only whereas the global reconfiguration strategy reroutes all unaffected and affected traffics for failure recovery.

4.2.3.4 Total Network Cost

We have evaluated the metric total network cost for comparing the restoration strategies and the comparison results are captured in Table 4.25 and depicted in Figure 4.24.

Table 4.25: Total Network Cost (in Dollars) for Link vs. Global Reconfiguration Strategy for different traffic demands

Traffic Demands	0.1	0.2	1	2	5	10
Link	0.200	0.260	0.890	1.712	4.148	8.215
Global	0.148	0.236	0.812	1.247	2.452	4.138

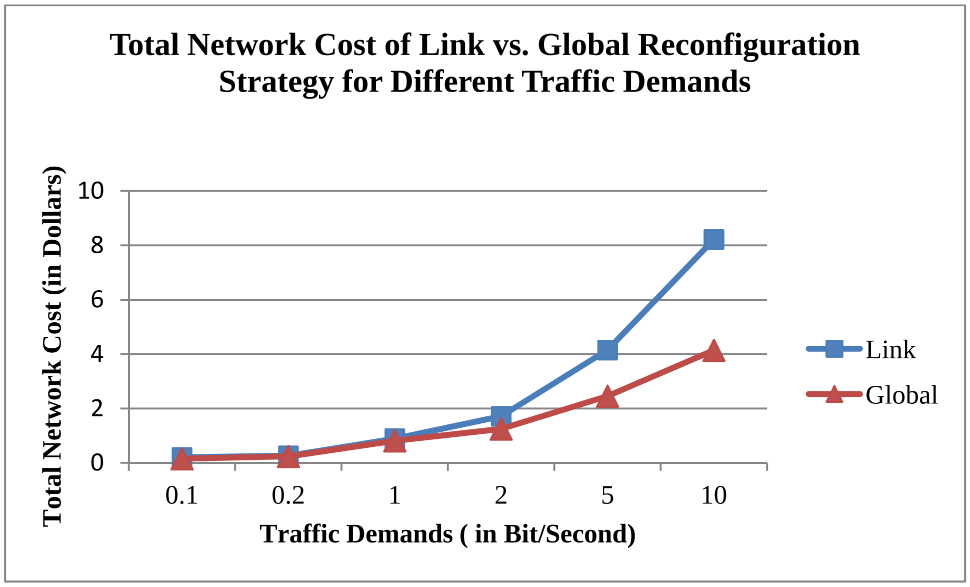


Figure 4.24: Total Network Cost (in Dollars) for Link vs. Global Reconfiguration Strategy for different traffic demands

Figure 4.24 reveals that the link restoration strategy requires more network cost than the global reconfiguration strategy. This is attributed to the fact that the global reconfiguration scheme generates less capacity installation cost compared to the link restoration strategy.

4.2.3.5 CPU Time

With respect to CPU time, we have compared the link and global reconfiguration strategy and the results are captured in Table 4.26 and depicted in Figure 4.25.

Table 4.26: CPU Time (in Seconds) for Link vs. Global Reconfiguration Strategy for different traffic demands

Traffic Demands	0.1	0.2	1	2	5	10
Link	0.110	0.180	0.820	0.830	0.830	0.840
Global	0.080	0.120	0.200	0.260	0.260	0.260

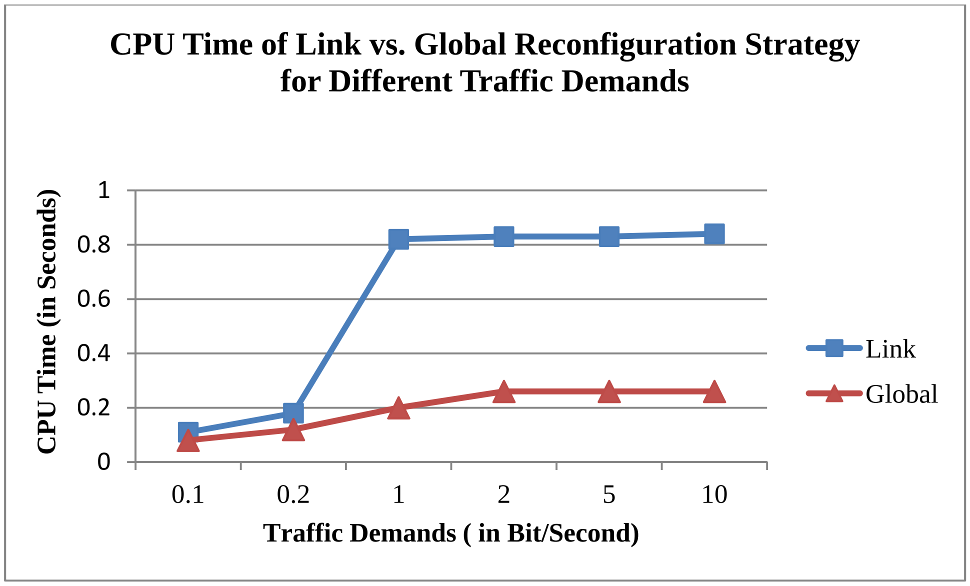


Figure 4.25: CPU Time (in Seconds) for Link vs. Global Reconfiguration Strategy for Different Traffic Demands

Figure 4.25 shows that our proposed link restoration strategy can converge very fast for large size networks i.e. only 0.84 seconds when the traffic demand is 10 bits per second per commodity. The global reconfiguration strategy takes less time to converge compared to the link restoration strategy.

Chapter 5

Conclusion

In this thesis, we have addressed the CFA problem in self-healing ATM network under link restoration strategy. The problem has been modeled as a multicommodity nonlinear model and solved using the Separable Augmented Lagrangian Algorithm (SALA) assuming that the network topology and projected traffic demands are given.

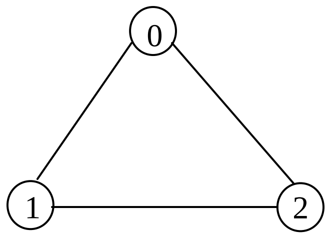
The findings of this thesis can be summarized as follows:

- The global reconfiguration strategy performs better than the link restoration strategy in terms of the total network cost, total used capacity and CPU time. But in terms of routing cost, link restoration strategy always performs better than the global reconfiguration strategy.
- The difference between the global reconfiguration and link restoration strategies is more pronounced in networks having fewer arcs in terms of total used capacity and capacity installation cost. But in terms of routing cost, the difference between these two strategies is more pronounced in a network having large number of arcs.
- The difference between the global reconfiguration and link restoration strategies is quite small for a small bandwidth requirement, whereas the difference becomes evident for a large bandwidth requirement.

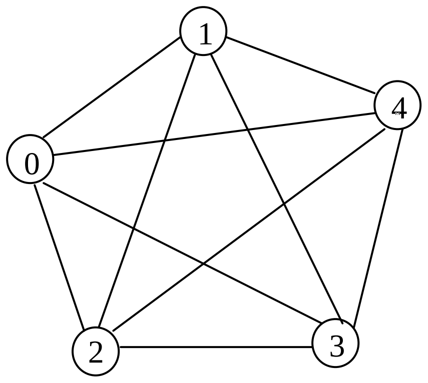
- The capacity installation cost, total required capacity and the total network cost gradually decrease for both of the restoration strategies when the number of arcs of a network gradually increases, using symmetric and asymmetric traffic demands.
- All the five performance metrics of the two restoration strategies are increased when the bandwidth requirements of the commodities are increased.

As a future work, node failures could be investigated as well as applying our solution approach to other types of networks such as Generalized Multi-Protocol Label Switching (GMPLS) networks.

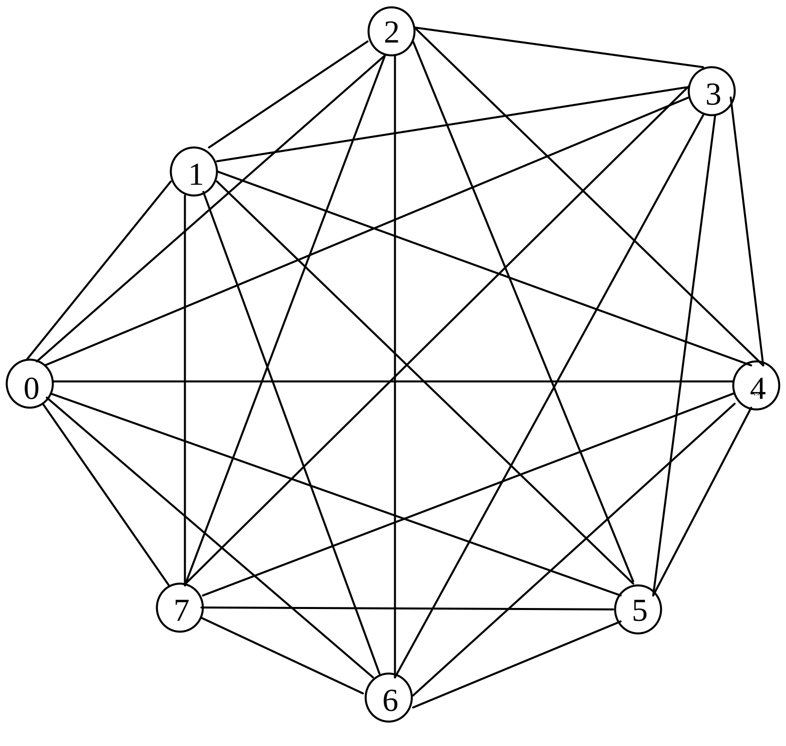
Appendix A



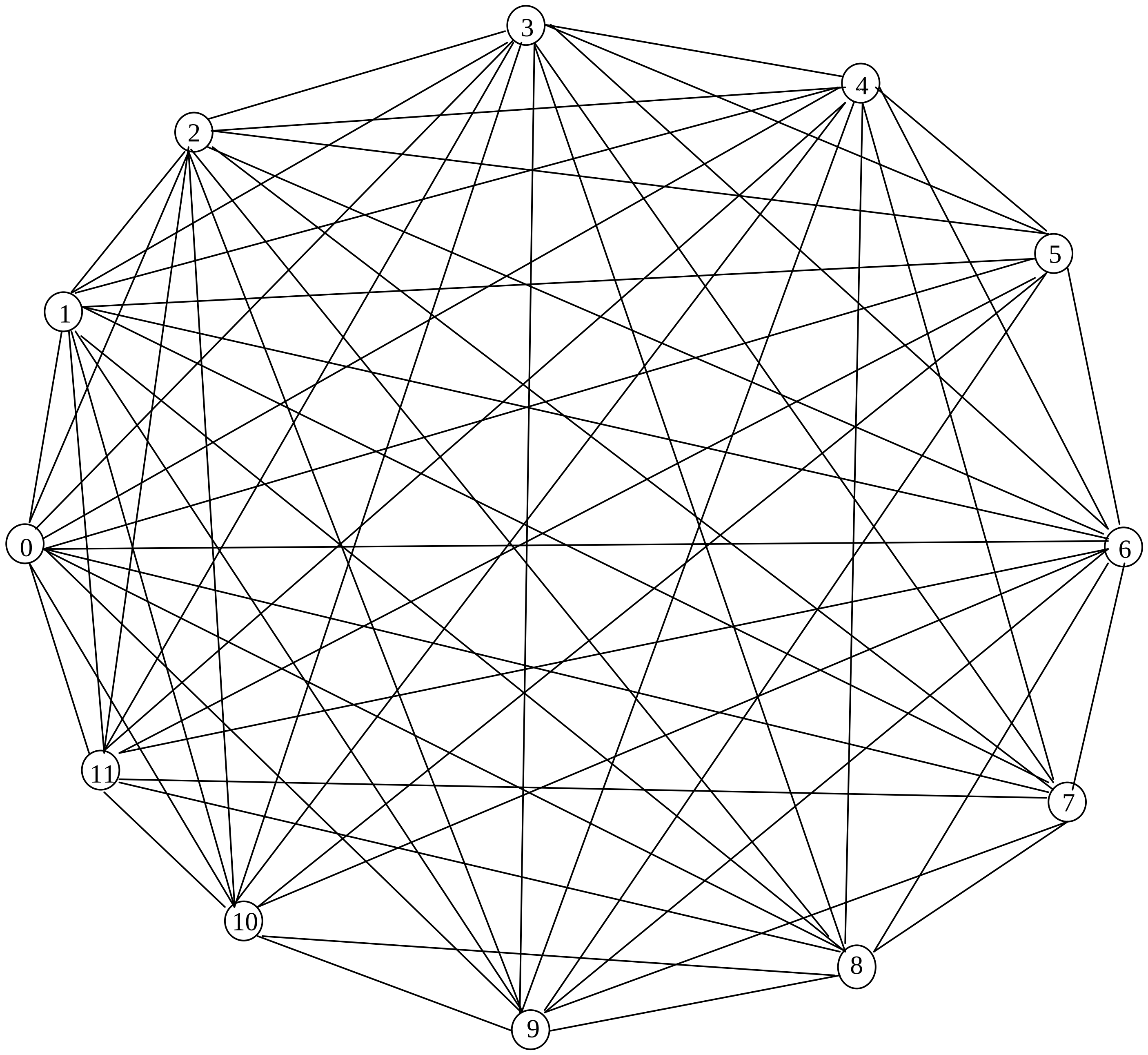
N(3, 6) Network with 3 Nodes and 6 arcs



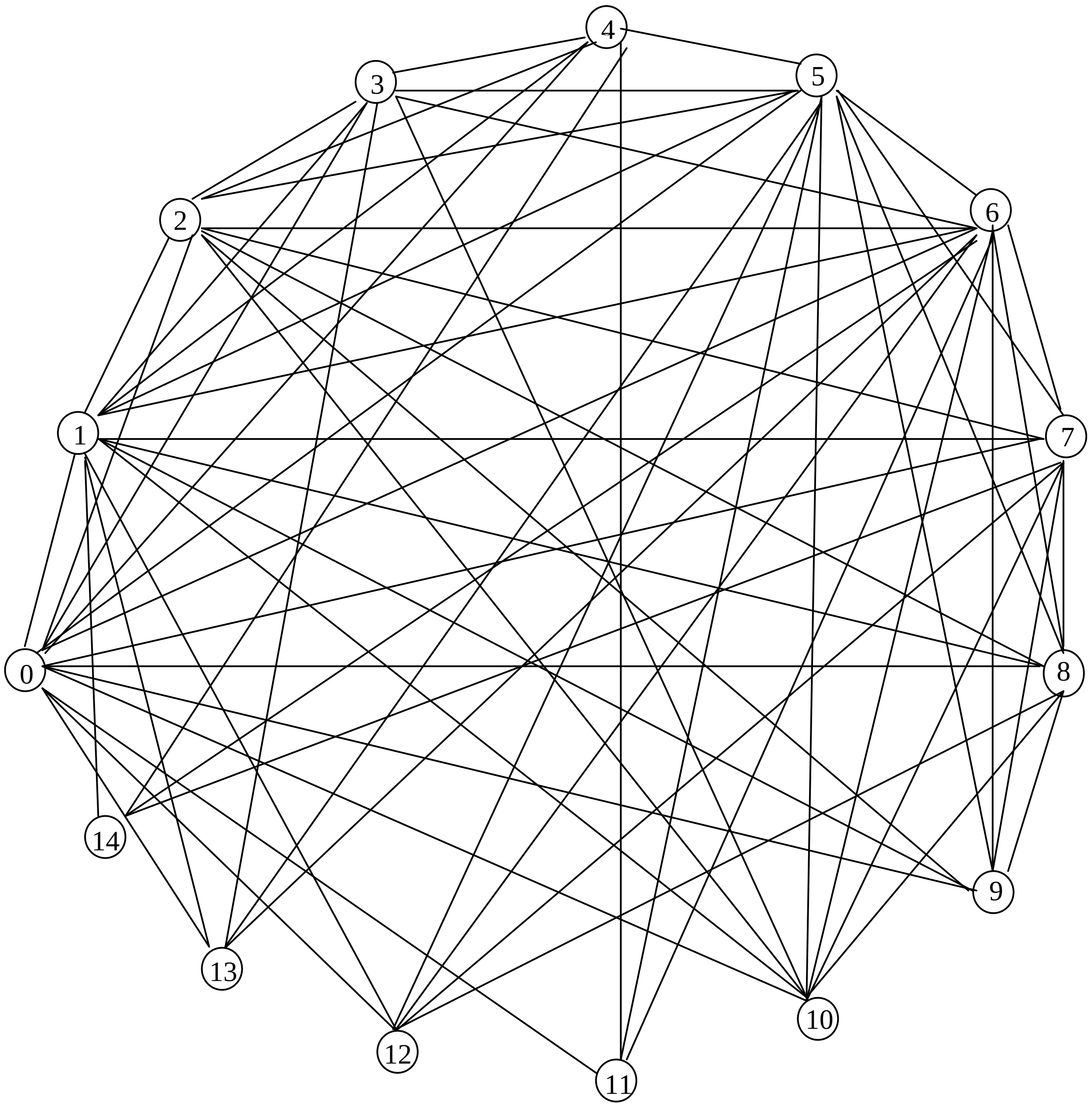
N (5, 20) Network with 5 Nodes and 20 arcs



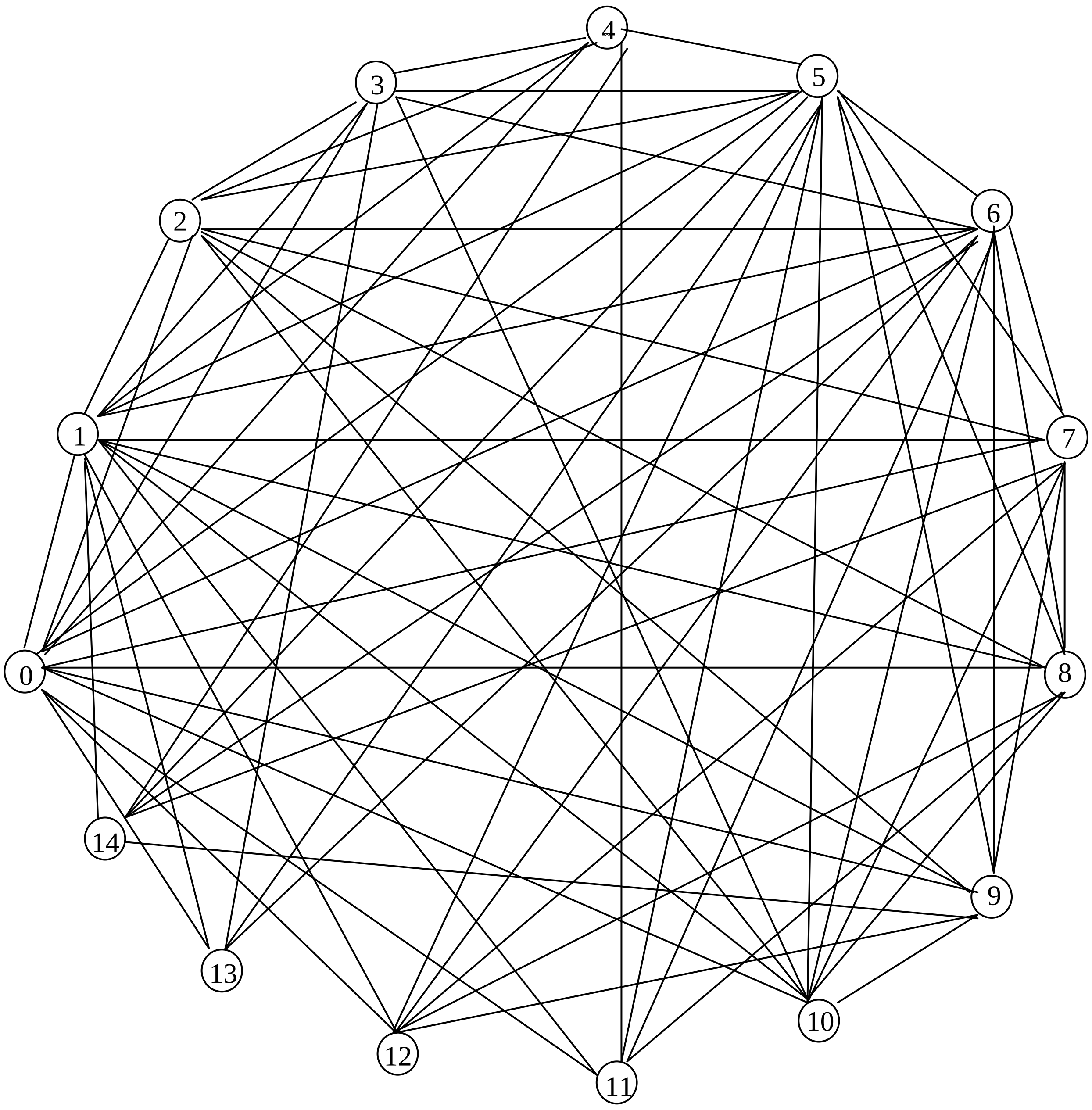
N(8, 56) Network with 8 Nodes and 56 arcs



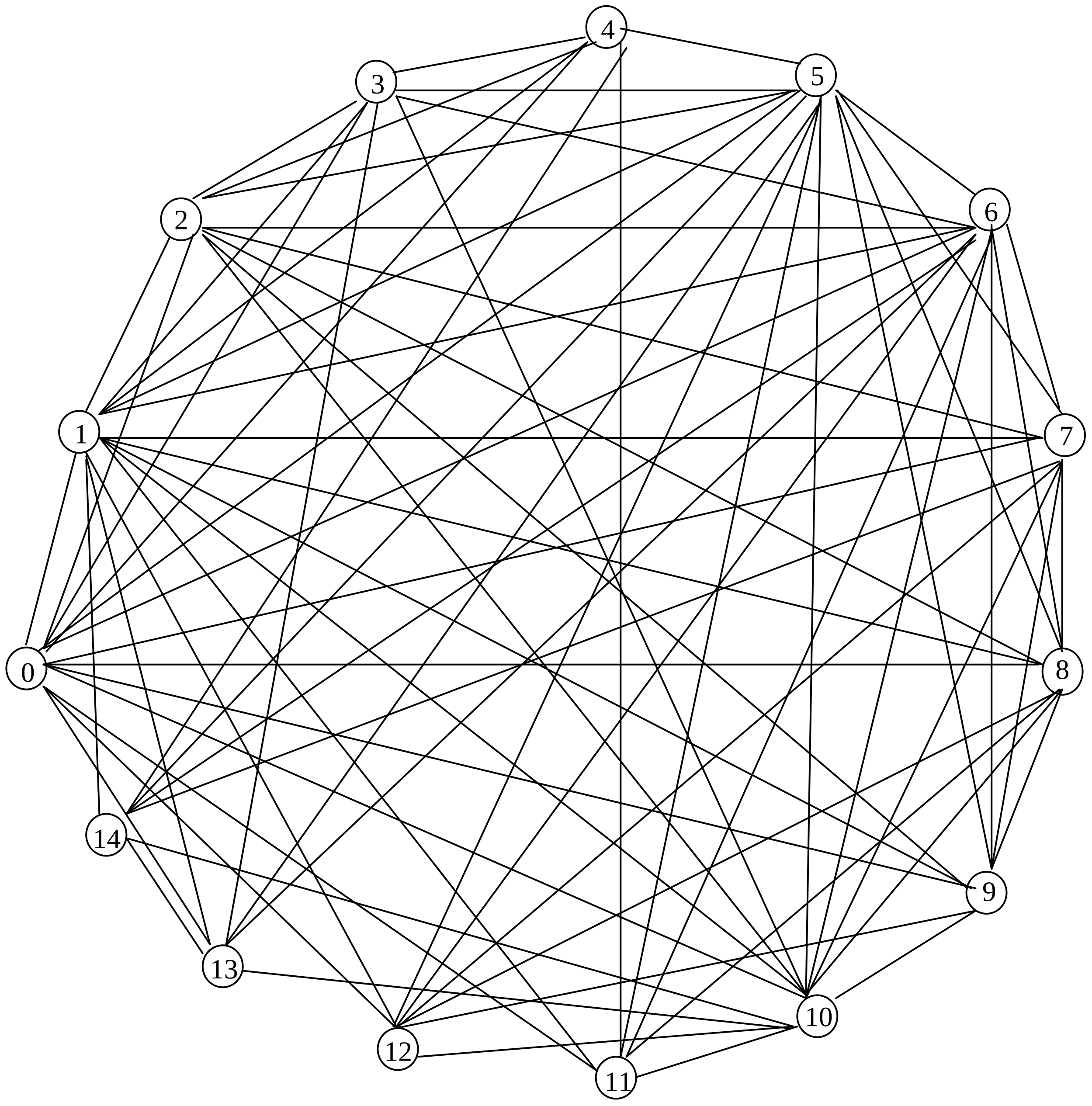
N (12,126) Network with 12 Nodes and 126 arcs



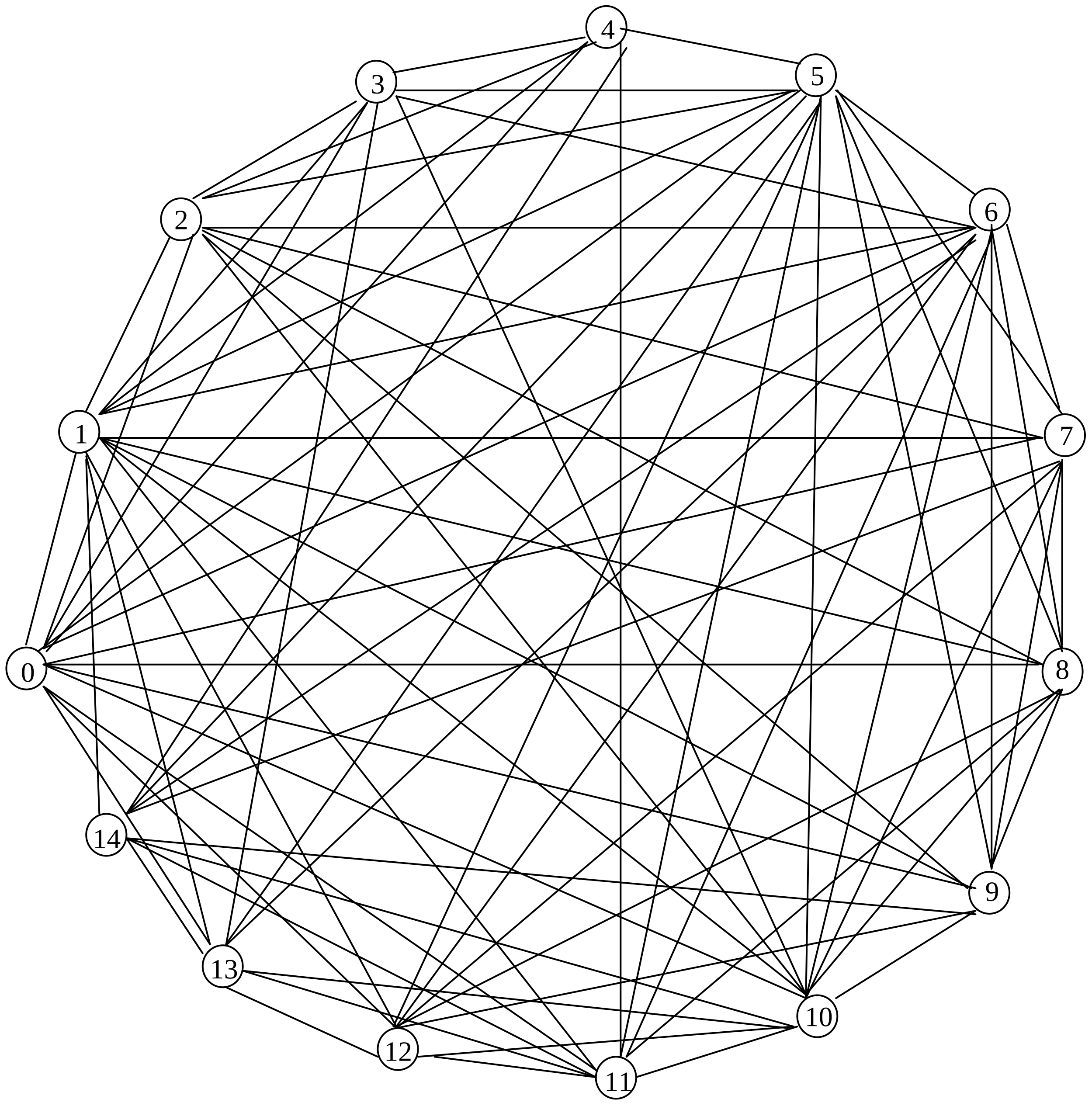
N (15,132) Network with 15 Nodes and 132 arcs



N (15,142) Network with 15 Nodes and 142 arcs



N (15,152) Network with 15 Nodes and 152 arcs



N (15,162) Network with 15 Nodes and 162 arcs

References

[1] D. Rajan and A. Atamturk, “Survivable network design: routing of flows and slack” in *Telecommunications Network Design and management, Sixth INFORMS Telecommunications Conference*, pp. 65-81, Florida, USA, March 2003.

[2] I. Woungang, G. Ma, M. K. Denko, S. Misra, H. C. Chao and M. S. Obaidat, “Survivable ATM mesh networks: Techniques and performance evaluation”, *Journal of Systems and Software*, vol. 83, Issue 3, pp. 457-466, 2010.

[3] B. Jxger and D. Tipper, “On fault recovery priority in ATM networks,” in *Proceedings of the IEEE International Communications Conference (ICC’98)*, vol. 3, pp.1410 – 1414, Atlanta, GA, June, 1998.

[4] K. Murakami, S. Hyong, “Near-Optimal Virtual Path Routing for Survivable ATM Networks”, in *Proceedings of International Conference on Computer Communications,* pp.208-215, Toronto, Ontario, Canada, June, 1994.

[5] Y. Xiong and L.G. Masson, “Restoration Strategies and Spare Capacity Requirements in Self-Healing ATM Networks”, *IEEE/ACM Transactions on Networking*, vol. 7, No. 1, pp. 98-110, February 1999.

[6] C. Y. Lee and S.J. Koh, “A Design of Self-Healing ATM Networks Based on Backup Virtual Paths”, *Computers and Operations Research*, vol. 25, Issues 7-8, pp. 595-609, July 1998.

[7] B. A. Forouzan, *Data Communications and Networking*. 4th ed. New York: McGraw-Hill, 2007, pp. 523-535, ISBN 978-0-07-296775-3.

[8] Sumit Kasera, *ATM Networks Concepts and Protocols.*1st ed. India: McGraw-Hill, 2006, pp. 40-50, ISBN 978-0-07147732-1.

[9] W. P. Wang, D. Tipper, B. Jxger, and D. Medhi, " Faulty recovery routing in wide area packet networks," in *Proceedings of 15th International Teletraffic Congress*, Washington, DC, June 1997.

[10] C. T. Kelly, *Iterative Methods for Optimization*, in: *Frontiers in Applied Mathematics*, vol.18, Philadelphia, PA: SIAM, 1999, ISBN 978-0898714333.

[11] P. E. Frandsen, K. Jonasson, H.B. Nielsen, and O. Tingleff, *Unconstrained Optimization, Informatics and Mathematical Modelling*, 3rd ed. Lyngby, Denmark: Technical University of Denmark, 2004, ch.5, sec.5.1, pp.45-46.

[12] A. Hamdi, P. Mahey and J.-P. Dussault, " A New Decomposition method in non convex Programming", *Recent Advances in Optimization*, P. Gritzmann, R. Horst, E. Sachs and R. Tichatschke eds., Lecture Notes in Economics and Mathematical Systems, vol. 452, pp. 90-104, 1997.

[13] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed. Belmont, Mass.: Athena Scientific, 1999, ISBN 1-886529-00-0.

[14] L. Kleinrock, *Queuing systems*, vol. 1, New York: Wiley Interscience, 1975, ISBN 978-0471491101.

[15] K. Murakami and S. Hyong, "Optimal Capacity and Flow Assignment for Self-Healing ATM Networks Based on Line and End-to-End Restoration", *IEEE/ACM Transaction on Networking*, vol. 6, no. 2, pp. 207-221, April 1998.

[16] K. Murakami and S. Hyong, “Comparative Study on Restoration Schemes of Survivable ATM Networks”, *In Proceedings of International Conference on Computer Communications,* vol. 1, pp. 345 – 352, Kobe, Japan, April 1997.

[17] K. Murakami and S. Hyong, “Joint Optimization of Capacity and Flow Assignment for Self-Healing ATM Networks”, in *Proceedings of IEEE International Communications Conference*, pp. 216-220, Seattle, WA, 1995.

[18] I. Woungang, S. Misra, and M. S. Obaidat, “On the Problem of Capacity Allocation and Flow Assignment in Self-healing ATM networks”, *Journal of Computer Communications,* vol. 30, Issue 16, pp. 3169-3178, 2007.

[19] C-S Wu, S-W Lee, Y-T Hou and Y-S Chu, “A New Preplanned Self-healing Scheme for Multicast ATM Network”, In *Proceeding of International Conference in Communication and Technology (ICCT’96)*, pp. 888-891, Beijing, May 1996.

[20] 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, Dec. 1995.

[21] D. Dunn, W. Grover and M. McGregor, "Comparison of k-shortest paths and maximum flow routing for network facility restoration", *IEEE J. Select. Areas Commun*, vol. 12, pp. 88-89, 1994.

[22] W. D. Grover, T. D. Bilodeau and B. D. Venables, "Near optimal spare capacity planning in a mesh restorable network", in *Proc. IEEE Global Telecommunications Conference*, pp. 2007-2012, Phoenix, AZ, Dec. 1991.

[23] H. Sakauchi, Y. Nishimura and S. Hasegawa, "A self-healing network with an economic spare-channel assignment" in *Proc. IEEE Global Telecommunications Conference*, pp. 438-443, San Diego, CA, Dec. 1990.