End-to-end Shared Restoration in Multi-domain Networks

By

Zhiying Gao

A thesis presented to Lakehead University

in partial fulfillment of the requirement for the degree of

Master of Science in Control Engineering

Thunder Bay, Ontario, Canada, 2008

Your file Votre référence
ISBN: 978-0-494-43421-5
Our file Notre référence
ISBN: 978-0-494-43421-5

Content

List of Figures

List of Tables

Acknowledgements

First of all, I gratefully acknowledge my supervisor, Dr. Hassan Naser, for the great support and opportunity he provided throughout my study and research at Lakehead University. His office door was always open to me whenever I needed his advice on the research or other problems. The accomplishment of my thesis would have remained a distant dream without him. I have learnt a great deal from him and will always cherish all that I have learnt from him. I would like to thank my professors, Dr. Zhiwei Mao, Dr. Abdelhamid Tayebi and Dr. Xiaoping Liu, for inspiring and enriching my growth as a student and a researcher.

I would also like to thank other members of our research team for their advices and their willingness to share their thoughts with me. I am very grateful to all my friends in Canada for being friendly, caring, supportive and helpful in many ways.

I am extremely grateful to my family for everything they have done for me. They have been a long lasting source of energy during this research and always been the inspiration and motivation in my life.

Abstract

Emerging multi-service data applications require high-bandwidth high-quality connectivity across multiple network domains, each of which is generally controlled by an independent service provider. These applications necessitate the need for highly intelligent survivable routing mechanisms to compute end-to-end paths and to perform functions of protection and bandwidth management across multiple domains. On the other hand, current protection and restoration mechanisms focus on the network survivability inside a single domain network. Powerful dynamic protection and restoration algorithms have been developed for single-domain networks. The majority of these algorithms are based on the exchange of detailed link-state information among the nodes, which makes them less attractive to networks with multiple domains where link-state information needs to be abstracted within each domain for efficiency and scalability reasons.

To address this problem, we present two network information abstraction models designed to aggregate link-state information within each domain and only to advertise the aggregated information to other domains. The first abstraction model is referred to as *virtual path* abstraction model, with which every domain is abstracted as a set of border-nodes interconnected by virtual paths. The multi-domain network is then topologically aggregated to become a single-domain network, called virtual path network, which consists of border-nodes interconnected internally by virtual paths and externally by inter-domain links. The second abstraction model is referred to as *virtual node* model, with which

every domain is modeled as a virtual node with a certain internal minimum capacity that can be advertised to other domains. The multi-domain network is then topologically aggregated to become a single-domain network, called virtual node network, consisting of virtual nodes interconnected by inter-domain links.

We have designed and developed three distributed end-to-end shared restoration schemes based on the information abstraction models presented above. These three schemes are referred to as Link Disjointed Virtual Path (LDVP) restoration, Domain Disjointed Virtual Path (DDVP) restoration, and Link Disjointed Virtual Node (LDVN) restoration. The LDVP and LDVN schemes are designed to provide link diversity between the primary and backup paths of each demand, whereas the DDVP scheme is designed to compute a pair of domain-disjointed paths for the demand.

We show that the proposed schemes are more scalable than the existing restoration schemes because they require less amount of link-state information to be advertised between the domains. This will reduce the routing message overhead and make the proposed schemes to be scalable to large multi-domain networks.

We also evaluate the performance of the proposed schemes in terms of capacity usage and restoration time through simulation experiments on two multi-domain networks; one is based on the NSF (National Science Foundation) network, and the other is based on the European Optical Network. The simulation results show that the proposed schemes save the backup bandwidth significantly because of the sharing of backup resources among failure-disjointed connections. The simulation results also show that the restoration time achieved by the proposed restoration schemes (over the multi-domain network) is around or less than 60 ms, which is within the range accepted in today's networks.

Notations

r	Demand
b	The bandwidth that demand r requests
d	Domain
src	Source domain
des	Destination domain
L_d	The matrix recording the link-disjointed relationship between any two virtual paths in domain d
l_{ij}^{d}	The link-disjointed relationship between i and j in domain d
K	The global matrix that records the backup bandwidth reserved on virtual paths, inter-domain links, or virtual nodes
k_{ij}	The amount of backup bandwidth needed on j if i fails
U_d	The matrix recording the backup bandwidth reserved on links in domain d for the local traffic
u_{ij}^{d}	The amount of backup bandwidth needed on a link j if a link i fails in domain d
V_d	The matrix recording the backup bandwidth reserved on links of domain d for the transit traffic
v_{ij}^{d}	The amount of backup bandwidth needed on link j in domain d if an inter-domain link or virtual node i fails
B_j	The total amount of backup bandwidth needed on j
$B_n^{d,T}$	The total amount of backup bandwidth needed on link n in domain d for the transit traffic
$S_{VP}(n)$	The set of all virtual paths that cross link n
$B_n^{d,L}$	The total amount of backup bandwidth needed on link n in domain d to restore the local traffic
B_n^{d}	The total amount of backup bandwidth needed on link n in domain d
$W_p^{d}(n)$	The cost of choosing intra-domain link n in domain d to be on the primary path of demand r
A_n^{d}	The available capacity on link n in domain d
$W_p(j)$	The cost of choosing j to be on the primary path for demand r
A_j	The available capacity on j
C_d^{I}	The internal capacity of virtual node (domain) d

$I(d)$	The set of all inter-domain links terminating on d
$W_b^d(n)$	The cost to choose link n in domain d on the backup path
$S_p^d(r)$	The set of links in domain d that are on the primary path of demand r
$T_n^{d,L}$	The maximum amount of backup bandwidth required on link n in domain d if a link in $S_p^d(r)$ fails
$W_b(j)$	The cost to choose j to be on the backup path of demand r
$S_p(r)$	The set of all inter-domain links and virtual paths/ virtual nodes along the primary path of demand r
$T_n^{d,T}$	The maximum amount of backup bandwidth required on link n in domain d if any member in set $S_p(r)$ fails
$S_{p,n}(r)$	The set of virtual nodes along the primary path of demand r
$M_P(r)$	The set of all intermediate domains of the primary path of demand r
$V(d)$	The set of virtual paths in domain d
T_j	The maximum amount of backup bandwidth required on j if any member in $S_p(r)$ fails
P_m^d	The total primary bandwidth already reserved on m in domain d
P_i	The total primary bandwidth already reserved on i
$S_l(j)$	The set of all links along virtual path j
$S_b(r)$	The set of inter-domain links and virtual paths/ virtual nodes on the backup path of demand r

Table I Notations Used in the Proposed Schemes

Chapter 1 Introduction

1.1 Network Survivability Mechanisms

In an optical network, each physical link is composed of several logical channels and each channel is expected to operate at a rate of several gigabits per second, and therefore the failure of network elements (e.g., fiber links and cross-connects) may lead to the failure of several optical channels, and may cause large amount of data loss. Hense, it is crucial to provide some mechansims to protect optical transport networks from failures [1][2][3][4].

A network failure may occur at a node or on a link in the network. However, because current node architectures usually have built-in redundancy that greatly improves their reliability, failures on links become more of a concern than node failures [5]. According to the reports from real world [2][3][4], multiple-link failures are extremely rare and single-link failures are the predominant form of failures in optical networks because the occurrence frequency of a link failure is very low. The research presented in this thesis has been focused on recovering network traffic in a multi-domain network in case of a single-link failure.

1.1.1 Protection and Restoration

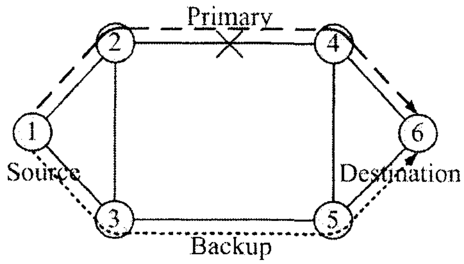
Survivability of a network can be defined as the network's capability to survive from network failures. Protection and restoration are two different schemes to ensure network survivability. Protection usually refers to the proactive survivability mechanisms. On the other hand, restoration refers to reactive survivability meachansims.

In protection schemes, backup resources are pre-computed and reserved for each connection at the conncection setup time. Two traditional classes of protection schemes are 1+1 and 1:1 protection. With 1+1 protection, two copies of the data are sent simultanously to the destination node; one copy along the primary path and the other along the backup path. The destination node chooses the copy with the better quality. If a failure occurs on one of the two paths, the destination would still receive the data transmitted along the other path. With 1:1 protection, a copy of data is only sent along the primary path. The backup path is pre-calculated and reserved for the future when the primary path is failed.

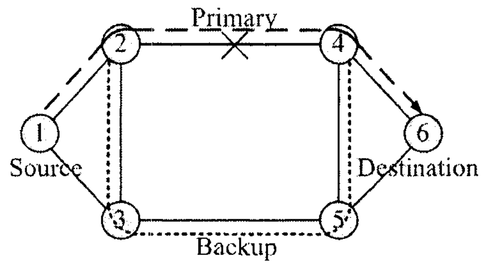
In restoration schemes, the backup resources can be pre-calculated but not allocated to each connection, or can be dynamically calculated and allocated as soon as a failure occurs. In contrast to the protection schemes, the restoration schemes are capable of sharing the backup capacity among different connections whose primary paths are failure-disjointed. Due to the sharing of the spare capacity, restoration schemes are usually more efficient than protection schemes in terms of resource utilization. Previous research studies in [2] and [3] have shown that shared restoration is the most efficient strategy for spare capacity allocation while still achieving full restoration for any single network component (e.g. link) failure. However, one drawback of fully reactive restoration schemes (where the backup resources are dynamically computed and allocated as soon as a failure occurs) is that there is no gaurantee that a backup path is available for a failed connection when a failure occurs. Therefore, our research concentrates on the shared restoration schemes where the backup resources are computed in advance but allocated only when failures occur.

1.1.2 End-to-end and Link Restoration Schemes

Restoration schemes are also divided into two major categories: end-to-end (path) restoration and link restoration. End-to-end restoration refers to a class of restoration techniques that the traffic traversing a failed link is rerouted over a replacement path (backup path) between the source and destination nodes (Figure 1.1 (a)). Backup resources for each connection are reserved on an end-to-end basis. In link restoration schemes (Figure 1.1 (b)), the traffic traversed the failed link is rerouted around the failed link. Backup resources are calculated separately for each individual link along the primary path. In link restoration schemes, the selection of the path between the end-nodes of the failed link is limited to the same wavelength as that of the primary path. Whereas in end-to-end restoration, the backup path could use a different wavelength channel because it is selected between the source and destination nodes. For this reason, end-to-end restoration shows better resource utilization and is more attractive than link restoration [5]. End-to-end shared restoration is thus primarily considered in this thesis.



(a) End-to-end



(b) link

Figure 1.1 End-to-end and Link Restoration Schemes

1.2 Multi-domain Network Survivability

The existing protection and restoration mechanisms have focused on the restoration of network traffic in the event of a physical link (or node) failure inside a single network domain. Efficient dynamic protection and restoration algorithms have been developed under single-domain networks environment. The majority of these algorithms are based on the exchange of detailed link-state information among the nodes inside the domain [2]-[11].

However, emerging multi-service data applications require high-bandwidth high-quality connectivity across multiple domains. In a multi-domain network environment, as illustrated in Figure 1.2, domains are typically controlled by different autonomous service providers. Considering the scalability of the network and the confidentiality of each domain, a domain may not wish to exchange detailed information about the state of its resources and the topology of the network with other domains. Solutions to network protection and restoration based on a detailed information exchange will not be feasible. It is necessary to develop a new generation of highly intelligent survivable routing mechanisms to compute end-to-end paths and to perform functions of restoration and bandwidth management across multiple domains. And, this is where the major contribution of this thesis lies.

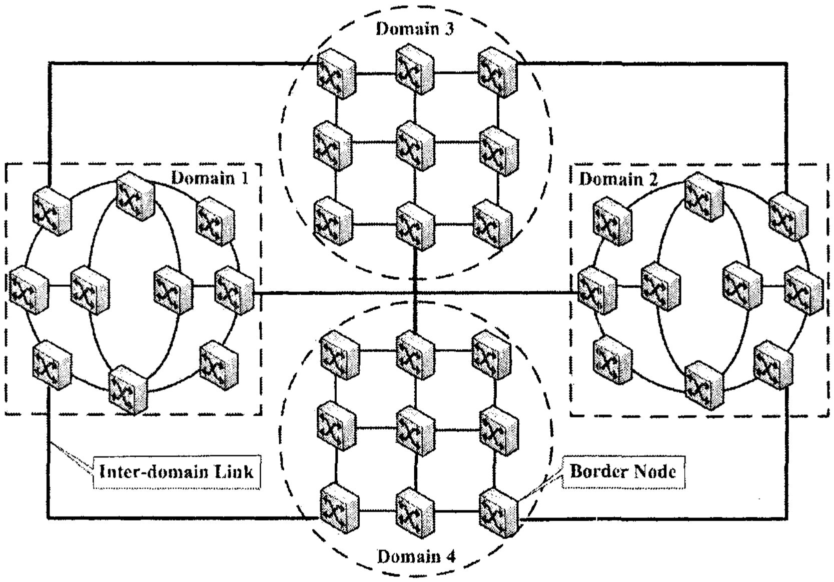


Figure 1.2 Multi-domain Network

1.3 Major Contribution of the Thesis

There are two challenges in developing efficient survivable routing mechanisms in multi-domain networks. First, in order to make the routing scalable, the link-state information that a domain advertises to other domains must be limited. Second, since domains are typically administered by different authorities, they will not provide their competitors with their confidential topology and state information [12].

To overcome these challenges, we introduce two network information abstraction models, upon which three end-to-end shared restoration schemes are designed in this thesis. The two information abstraction network models are referred to as *virtual path* model and *virtual node* model. In both information abstraction models, the link-state information is aggregated within each domain and only the aggregated information is advertised to other domains. The three restoration schemes we developed are referred to as: Link Disjointed Virtual Path (LDVP), Domain Disjointed Virtual Node (DDVP) and Link Disjointed Virtual Node (LDVN) shared restoration schemes.

These restoration schemes are designed to compute a pair of diverse primary and backup paths for a demand between any given pair of nodes in a multi-domain network. The primary path is dedicated to the demand, while the backup path is shared among different demands. The primary and backup paths for every demand will be link-disjointed when they are computed by using LDVP and LDVN schemes. However, they are domain-disjointed when DDVP scheme is used. Link-disjointed means that the paths have no links in common. Similarly, domain-disjointed paths have no domains in common.

With these schemes, the network traffic is divided into two categories: local and transit (remote). Local traffic is the traffic that is exchanged between two nodes inside a single domain. The local traffic is routed over the links of the domain's physical topology (i.e. over intra-domain links). Transit traffic is the traffic exchanged between two nodes in different domains. This traffic may thus cross one or more intermediate domains before reaching the destination domain.

The LDVP and DDVP schemes are designed to work over the virtual path network information abstraction model. With the virtual path model, the multi-domain network is topologically aggregated to become a single-domain network, called virtual path network, where each domain is abstracted by its border nodes interconnected by point-to-point virtual paths. Figure 1.3 illustrates a virtual path network created from the original multi-domain network shown in Figure 1.2. Virtual paths are paths that are computed (pre-planned) within each domain between the border nodes of that domain. Each domain will only advertise limited information (such as the available capacity) about its virtual paths to other domains. Therefore, all domains will have the same image of the virtual path network, which consists of the border nodes of all domains, the virtual paths interconnecting the border nodes, and the inter-domain links connecting the border nodes of the adjacent domains. A route for a traffic-demand to a remote domain is computed by the source domain over this virtual path network. No other information as to the nature and identity of the constituent links of each virtual path, or the extent of the search undertaken to compute the virtual paths is exchanged between the domains.

The LDVN scheme is however designed to work over the *virtual node* network information abstraction model. With the virtual node model, every domain is abstracted as a single virtual node with certain *internal capacity* that can be advertised to other domains. In order to route the transit traffic, the multi-domain network is hence topologically reduced to become a single-domain network, called virtual node network, as illustrated in Figure 1.4, which consists of virtual nodes interconnected by inter-domain links. Compared with the conventional least-cost path computation algorithms, the LDVN scheme not only associates a cost to every (inter-domain) link, but also associates a cost to every virtual node. The link/node costs are used to seek for a path (or paths in case of protection and restoration) that accumulates the least cost through the network. The link/node costs are dynamically computed as a function of the available transmission capacity (in the case of a link), or as a function of the internal capacity (in the case of a node). Once a path is computed over the virtual node network, every domain along the path is then responsible for determining an explicit route over its intra-domain links for that path.

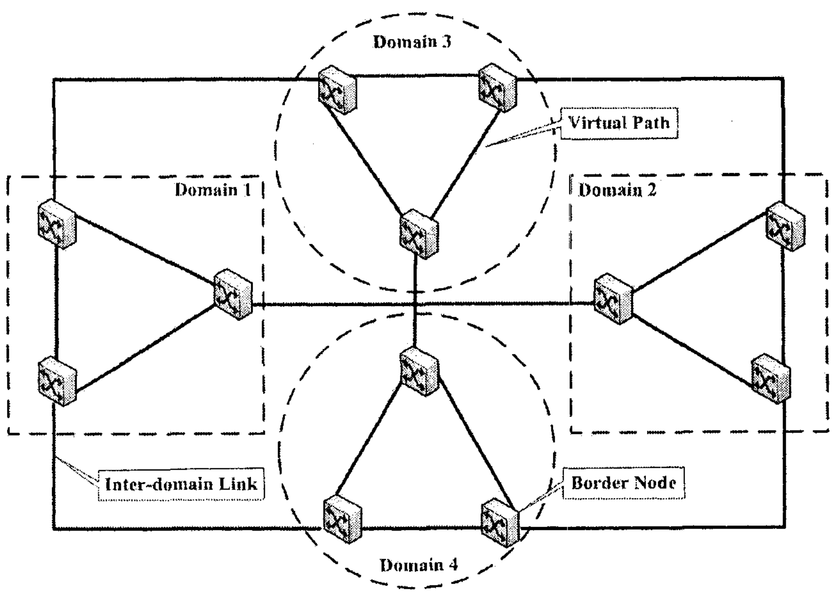


Figure 1.3 Virtual Path Network

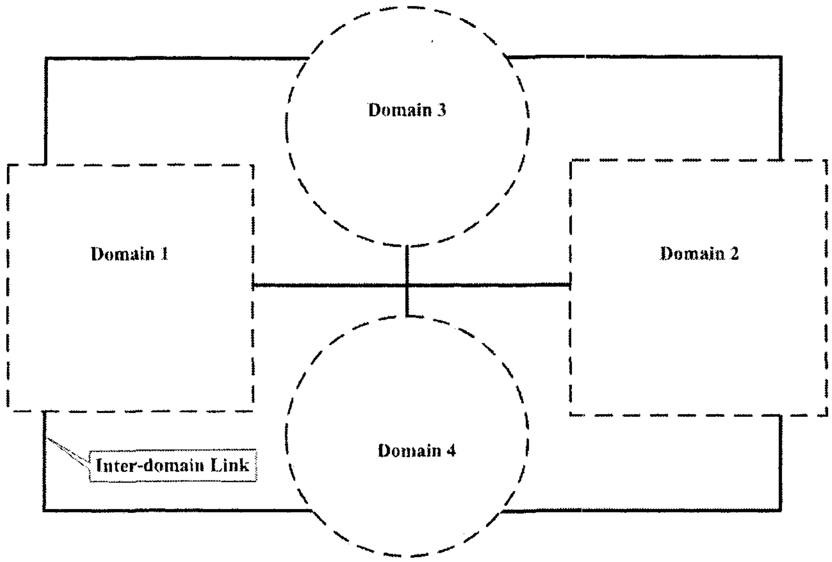


Figure 1.4 Virtual Node Network

1.4 Performance Evaluation

The proposed LDVP, DDVP and LDVN schemes have been simulated by using Python programming language on eclipse IDE. The simulation has been carried out over two multi-domain networks to evaluate and compare the capacity usage and restoration time of the proposed schemes. The two simulated multi-domain networks are: the NSF (National Science Foundation) network that is one of the representative North American backbone networks, and the European Optical Network (EON).

One advantage of the proposed schemes over the existing restoration schemes is that they require much less amount of link-state information to be advertised between the domains. This will reduce the routing message overhead and make the proposed schemes to be scalable to large multi-domain networks.

1.5 Outline of the Thesis

The rest of the thesis is organized as follows: Chapter 2 is a review of the existing research related to multi-domain networks survivability; Chapter 3 presents LDVP and DDVP end-to-end shared restoration schemes which are developed to work over the virtual path network model. First, the concept of the virtual path network model is presented. Then, the models for bandwidth recording and link cost assignment, and procedures used in LDVP and DDVP for computing a pair of diverse paths between a given pair of source-destination nodes are presented. Chapter 4 presents the LDVN scheme which is designed to work over the virtual node network model. Chapter 5 presents the simulation and performance evaluation of the proposed schemes. Chapter 6 provides the concluding remarks and the future work.

Chapter 2 Literature Review

2.1 Existing Research on Single-domain Networks

Many powerful protection and restoration algorithms have been developed under a single domain network environment [6]-[11]. References [6], [7] and [8] present some restoration schemes for single-domain networks and examine the capacity performance of these schemes. Reference [9] investigates different existing protection and restoration schemes, such as shared path protection and *p*-Cycle (pre-configured protection cycle) protection, and compares these schemes in terms of both capacity efficiency and recovery time. Reference [10] proposes a protection algorithm that takes QoS (Quality of Service) parameters into consideration. Reference [11] investigates a classic two-step shared restoration algorithm that aims to compute a pair of link-disjointed paths between a given pair of nodes. All of the algorithms introduced in these studies are based on the detailed link state information exchange among the network. They are not feasible in the scenario where the traffic needs to traverse more than one domain under a multi-domain network environment.

2.2 Existing Research on Multi-domain Networks

2.2.1 Information Aggregation Techniques

In a multi-domain network environment, domains are typically controlled by different autonomous service providers. Considering the scalability of the network and the confidentiality of each domain, a domain may not wish to exchange detailed information about the state of its resources and the topology of the network with other domains. Solutions to network protection and restoration based on a detailed information exchange are not feasible in multi-domain networks. One approach that has been commonly adopted to address this problem is to aggregate the internal information within each domain in a multi-domain network. Aggregation could hide the detailed information within each domain and reduce the amount of information advertised between the domains across the network. References [13] and [14] have introduced full mesh and symmetric star topology aggregation approaches. In the full mesh topology representation, each domain is represented by its border nodes and logical links coupling the border node pairs. The logical links are indeed the shortest paths between border node pairs. The aggregated state of a domain does not contain detailed information about the internal structure of the domain, but only contains information describing end-to-end properties, i.e. the routing cost, between any two border nodes. In the symmetric star topology representation, a logical node is introduced in each domain as the center of the star topology. Each domain is logically transformed to a star topology containing the logical links interconnecting the logical center node and each of the border nodes. The costs between any two border nodes are simply assumed to be the same and usually the average of the costs between all border node pairs. The aggregated state of a domain contains the border nodes and the routing cost between any two border nodes. Each domain only advertises its aggregated state information to other domains. The impact of topology aggregation on routing performance is also evaluated in the references [13] and [14]. The authors have found that, in general, full mesh representation is very accurate, and performs similar to flat non-hierarchical routing. The full mesh topology aggregation approach is thus adopted in the virtual path and virtual node network information abstraction models to be described in this thesis.

2.2.2 Routing Algorithms in Multi-domain Networks

Many research studies have explored routing algorithms in multi-domain networks. Reference [15] addresses the challenges of supporting Quality of Service (QoS) in Border Gateway Protocol (BGP), which has been the standard inter-domain routing protocol in the Internet. To address these challenges, they introduce an inter-domain QoS routing model that makes use of an "Inter-domain Routing Agent (IRDA)" in each domain to advertise QoS information. In [16], a "Path Computation Element" (PCE) in each domain is introduced, where an end-to-end path across domains is computed by the collaboration of PCEs in different domains. Reference [17] has presented "Route Sever" (RS) architecture for inter-domain QoS routing. In this architecture, each domain is abstracted as a number of pipes. Each pipe is associated with QoS parameters including delay, packet loss, available capacity, and cost. In [18], the game theory is used to analyze inter-domain routing in multi-domain networks. Reference [19] provides a review of the existing path computation schemes in multi-domain network environment. These schemes are categorized into PCE-based and per-domain path computation schemes. They introduce a new per-domain path computation scheme, Computation While Switching (CWS), which keeps finding a better path after successfully finding an initial path, thus resulting in an optimal or near-optimal path without assuming the availability of complete topology information.

We noticed that the scope of the above referenced papers is limited to the routing of transport connections (with no protection and restoration) in multi-domain networks. Network survivability has not been taken into consideration in their proposed routing schemes.

2.2.3 Survivability Mechanisms in Multi-domain Networks

In fact, only a limited number of research studies deal with the problem of multi-domain network survivability. A summary of these papers is given below. In [20], a multi-domain network protection mechanism is proposed based on the establishment of independent protection mechanisms within individual domains and merging at the domain boundaries. In comparison with end-to-end mechanisms, individual domain protection and restoration mechanisms are not capacity efficient because of the lack of detailed information about the transit traffic demand (such as the type of protection that the demand requested). In the absence of this detailed information, when a link fails, the local domain will try to recover all traffic traversed the failed link even for the demands that did not request protection.

Reference [21] investigates the use of *p*-Cycle in a multi-domain network. With their *p*-Cycle protection scheme, the multi-domain survivability problem is decomposed into two-levels: the lower intra-domain level and the upper inter-domain level resilience. At the lower level, the intra-domain failure is recovered within its domain. At the upper level, the network information is aggregated and each domain is represented as a node. The inter-domain links between these nodes are each assigned as an "on-cycle" or a "straddling" link of a *p*-Cycle. *p*-Cycles are pre-configured at the upper level and considered unchanged while the network is being operated. If an inter-domain link fails, the traffic will be routed counter-clockwise along the associated *p*-Cycle. Generally, *p*-Cycle schemes require a large amount of capacity in each domain [21]. The traffic on the failed inter-domain link could not be recovered if there is no available capacity to across a domain on the *p*-Cycle.

Reference [22] has introduced how to provide multi-domain optical network protection by using Hamiltonian Cycles. The basic idea presented in [22] is to partition a mesh network into a set of protection domains, and use one Hamiltonian Cycle to protect each domain. Primary and backup resource allocation is carried out in two separate steps. First, working (primary) paths for a set of demands are found by using a shortest path calculation algorithm. Then, the working network is partitioned into a set of protection domains and one Hamiltonian Cycle is found in each domain to protect its corresponding domain. In the protection scheme introduced in [22], the traffic is routed based on the global availability of the internal information of every domain. The confidentiality of each domain has not been taken into consideration. In addition, this protection scheme is link (local) protection. Link protection is not resource efficient compared with path (end-to-end) protection [2][3].

Chapter 3 LDVP and DDVP Shared Restoration Schemes

In this thesis, we propose three novel solutions to end-to-end shared restoration in multi-domain networks. These three solutions are referred to as: Link Disjointed Virtual Path (LDVP), Domain Disjointed Virtual Node (DDVP) and Link Disjointed Virtual Node (LDVN) shared restoration schemes. LDVP and DDVP are developed over virtual path abstraction network, while LDVN is developed over virtual node abstraction network. In this chapter, we are going to introduce virtual path network and the restoration schemes---LDVP and DDVP schemes---developed over it. Virtual node network and LDVN restoration scheme will be presented in the next chapter, chapter 4.

This chapter is organized as follows: Section 3.1 gives an brief introduction on how to aggregate a multi-domain network into a single-domain virtual path network and how to use the aggregated virtual path network to implement end-to-end restoration of the traffic exchanged between two nodes located in different domains. Section 3.2 presents the mathematical network model used to configure virtual path network, to record the reserved bandwidth and to compute the shared backup bandwidth. Section 3.3 explains the procedures and algorithms of dynamic primary and backup path computation with LDVP and DDVP schemes.

3.1 End-to-end Restoration over Virtual Path Network

LDVP is designed to find a pair of link-disjointed paths between any given pair of nodes located in different domains in a multi-domain network environment. DDVP is designed to find a pair of domain-disjointed paths between these nodes. In both of the schemes, the network traffic is divided into two categories: local and transit (remote) traffic. Local traffic is the traffic that is exchanged between two nodes inside a single domain. The local traffic is routed over the links of the domain's physical topology. Transit traffic is the traffic exchanged between two nodes in different domains. This traffic may thus cross one or more intermediate domains before reaching the destination domain.

In order to route the transit traffic, the multi-domain network is topologically aggregated to become a single-domain network, called virtual path network, in which each domain is represented by its border nodes that are interconnected by point-to-point virtual paths. The border nodes are the nodes that have links to the nodes in the neighboring domains. Virtual paths are paths that are computed within each domain between the border nodes of that domain, according to a shortest-path constraint to be described. Figure 3.2 illustrates a virtual path network created from the original multi-domain network shown in Figure 3.1.

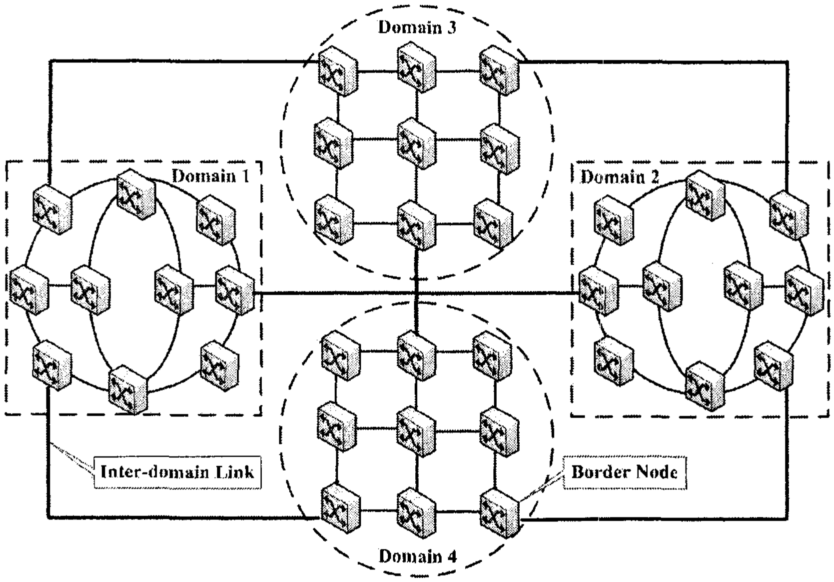


Figure 3.1 Multi-domain Network

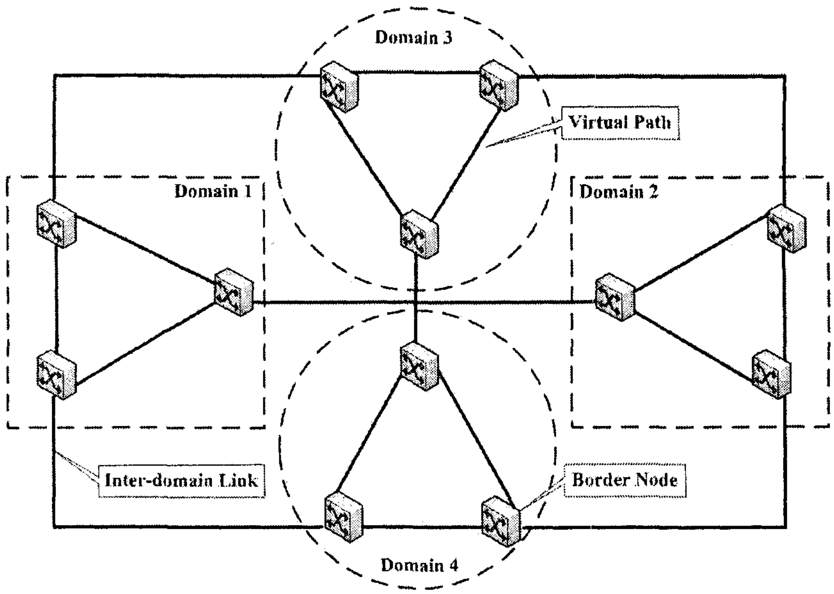


Figure 3.2 Virtual Path Network

At the transit level, each domain's physical network is thus replaced by a virtual path network consisting of virtual paths interconnecting the border nodes of that domain. Each domain will only exchange limited information about its virtual paths and their border nodes to other domains in the multi-domain network. More precisely, in LDVP scheme, each domain exchanges the available capacity on each of its virtual paths and the information about the link-disjointedness of its virtual paths to other domains. In DDVP scheme, only the available capacity on each of its virtual paths is exchanged between domains. Therefore, all domains will have the same image of the virtual path network, which consists of: 1) The border nodes of each domain; 2) The virtual paths interconnecting each pair of border nodes inside each domain; 3) The inter-domain links connecting the border nodes of the adjacent domains (see Figure 3.2). Corresponding to this virtual path network, there will be two global capacity matrices that record the reserved primary and backup bandwidths on the virtual paths and inter-domain links. Every domain has a copy of these matrices and must synchronize its own copy with other domains.

A route for the transit traffic is computed by the source domain over this virtual path network. No other information as to the nature and identity of the constituent links of each virtual path, or the extent of the search undertaken to compute the virtual paths is exchanged between the domains. A route from a source node to a destination node can be divided into three segments. The first segment is a path from the source node to a border node in the source domain. The second segment starts at the border node in the source domain and terminates at a border node in the destination domain. This segment may traverse zero, one or more intermediate domains. The third segment is a path from the border node in the destination domain to the final destination node in the destination domain.

The source domain has access to the detailed link-state information inside that domain in order to compute the first segment. It can also compute the second route-segment over the virtual path network by using the information provided by global capacity matrices. The source domain cannot however compute the third segment of the route, because it has no topological information about the destination domain---beyond what provided by the virtual paths---to compute a route over the local (physical) links in the destination domain. Hence, for both of the primary and backup path computations, the source domain first computes segments one and two from the source node to the shortest border node in the destination domain. The border node then computes the third segment from itself to the destination node, and concatenates the three segments to form an end-to-end path from the source node to the destination node. Once a source domain gets the end-to-end path for the transit traffic, it updates its own copy of the global capacity matrices and informs other domains to synchronize their databases with the changes made. Each domain is then responsible for mapping the changes in the virtual path's reserved capacity to the reserved capacity of its constituent physical links.

Both of the LDVP and DDVP schemes are divided into four stages of execution. At the initialization and network configuration stage, every domain's authority computes the virtual paths within its own domain, and then exchanges information about these virtual paths with other domains. After the initialization and configuration stage, all domains replace the local physical network by their border nodes and the virtual paths between them. Every domain will have an abstract view of the network by putting the virtual paths and inter-domain links together. At the second and third stages, the primary path and the backup path are computed respectively. Finally, the capacities of the links along the computed primary and backup paths are updated. The details, such as the mathematical network models and path computation algorithms, of the two schemes are described in the following sections.

3.2 Mathematical Network Models for LDVP and DDVP

3.2.1 Network Model for Local Traffic

In the local network environment, each domain d records the backup bandwidth reserved on its links for the local traffic in a private matrix U_d.

$$U_d = \begin{bmatrix} 0 & u_{12}^d & u_{13}^d & \dots & u_{1N_d}^d \\ u_{21}^d & 0 & u_{23}^d & \dots & u_{2N_d}^d \\ u_{31}^d & u_{32}^d & 0 & \dots & u_{3N_d}^d \\ \dots & & & & \\ u_{N_d1}^d & u_{N_d2}^d & u_{N_d3}^d & \dots & 0 \end{bmatrix} \tag{3.1}$$

Element u^d_{mn} is the amount of backup bandwidth needed on n if m fails. Both m and n are physical links in domain d, N_d is the total number of links in domain d.

3.2.2 Network Model for Transit Traffic (Configuration of Virtual Path Network)

At the network configuration stage, each domain computes the virtual paths inside that domain by using the Dijkstra's algorithm based on the minimum of number of hops routing criteria. The domains then assigns capacity to each of these virtual paths, and advertises these virtual paths to all other domains. On the basis of the advertised information, each domain creates the virtual path network consisting of the virtual paths inside all the network-domains and the inter-domain links between them.

The following information about virtual paths has to be exchanged between domains to support our proposal: 1) the identity of the border-nodes terminating each virtual path, 2) the available capacity on each virtual path, and 3) the link-disjointedness relationship between any two virtual paths in the same domain. There is a matrix L_d for each domain d to record the link-disjointed relationship between any two virtual paths in domain d,

$$L_d = \begin{bmatrix} 0 & l^d_{12} & l^d_{13} & \cdots & l^d_{1J_d} \\ l^d_{21} & 0 & l^d_{23} & \cdots & l^d_{2J_d} \\ l^d_{31} & l^d_{32} & 0 & \cdots & l^d_{3J_d} \\ \cdots & & & & \\ l^d_{J_d1} & l^d_{J_d2} & l^d_{J_d3} & \cdots & 0 \end{bmatrix} \tag{3.2}$$

J_d is the number of virtual paths in domain d. If virtual paths i and j are link-disjointed, the value of l_{ij}^d will be 1; otherwise, it will be 0. The detailed information, such as the internal network topology and the number of hops of a virtual path, is hidden from one domain to the other.

In the transit network environment, there is a global matrix K (shown below) to record the backup bandwidth reserved on virtual paths and inter-domain links for the transit traffic. Element k_{ij} in K is the amount of backup bandwidth needed on j if i fails. Both i and j are virtual paths and/or inter-domain links. All domains have a copy of this matrix. This matrix is advertised frequently.

$$K = \begin{bmatrix} k_{11} & k_{12} & k_{13} & \dots & k_{1J} \\ k_{21} & k_{22} & k_{23} & \dots & k_{2J} \\ k_{31} & k_{32} & k_{33} & \dots & k_{3J} \\ \dots & & & & \\ k_{J1} & k_{J2} & k_{J3} & \dots & k_{JJ} \end{bmatrix} \tag{3.3}$$

3.2.3 Shared Backup Bandwidth Calculation

The total amount of backup bandwidth needed on j to restore the transit traffic (denoted by B_j) is indeed the maximum of all elements in column j of matrix K:

$$B_j = \max_{\forall i \leq J} k_{ij} \tag{3.4}$$

If j is a virtual path in domain d, B_j will also be needed on every component (physical) link n of j. Because, in general, link n can be on more than one virtual path, we denote by $S_{VP}(n)$ the set of all virtual paths that cross link n. Hence, the total backup bandwidth needed on link n for the transit traffic is:

$$B_n^{d,T} = \sum_{j \in S_{VP}(n)} B_j \tag{3.5}$$

The total amount of backup bandwidth needed on link n to restore the local traffic (denoted by $B_n^{d,L}$) is indeed the maximum of all elements in column n of matrix U_d in equation (3.1). That is:

$$B_n^{d,L} = \max_{\forall n \leq N_d} u_{mn}^d \tag{3.6}$$

The total amount of backup bandwidth needed on link n (denoted by B_n^d) is therefore the sum of $B_n^{d,L}$ and $B_n^{d,T}$.

$$B_n^d = B_n^{d,L} + B_n^{d,T} \tag{3.7}$$

3.3 Dynamic Path Computation Algorithms

LDVP and DDVP restoration schemes are developed over the same abstraction network model, virtual path network. LDVP scheme is designed to provide a pair of link-disjointed primary and backup paths for every demand, whereas DDVP scheme is designed to provide domain-disjointed paths for the demand. Therefore, LDVP and DDVP schemes use different path computation algorithms over the virtual path network in order to find a pair of primary and backup path for each demand. These algorithms are described in Sections 3.3.1 and 3.3.2 respectively.

3.3.1 Dynamic Path Computation Algorithm in LDVP

The objective of this algorithm is to find a pair of link-disjointed paths between two given nodes located in different domains in a multi-domain network environment. The source domain has access to the information about the source local network and the intermediate virtual path network (outside of the source and destination domains). However, the source domain does not have access to the detailed internal information about the destination local network. Therefore, for both of the primary and backup path computations, the source node uses the physical network of the source domain and the virtual path network outside the source and destination domains to compute a path from the source node to every border node in the destination domain. Among these paths, the least-cost path is selected, and a path setup request is sent to the border-node in the destination domain that terminates the selected path. The border-node will use the physical network of the destination domain to compute the path-segment from the selected border node to the actual destination node. For every arriving demand r in a source domain d, a pair of link-disjointed paths will be computed dynamically using the following two-step restoration algorithm.

Step 1 (Primary Path Computation): for links in the source domain, the cost of choosing link n on the primary path is determined according to the following function:

$$W_p^{src}(n) = \begin{cases} 1 & b \le A_n^{src} \\ \infty & otherwise \end{cases}, \forall n \in [0,1,......N_{src}-1] \tag{3.8}$$

N_{src} is the total number of links in the source domain, b is the bandwidth that demand r requests, and A_n^{src} is the available capacity on link n in the source domain. $W_p^{src}(n)$ is the cost of choosing link n in the source domain to be on the primary path of demand r. The cost of link n is set to 1 if there is enough available capacity on the link to accommodate demand r. Otherwise, it is set to ∞.

For inter-domain links and virtual paths outside of the source and destination domains, the cost is determined according to the following function:

$$W_p(j) = \begin{cases} 1 & b \le A_j \\ \infty & otherwise \end{cases}, \forall j \in [0,1,......J_r - 1] \tag{3.9}$$

Where j denotes an inter-domain link or a virtual path, J_r is the total number of virtual paths and inter-domain links outside of the source and destination domains of demand r, and A_j is the available capacity on j. If j is a virtual path, A_j is the minimum available capacity of all component links of j. $W_p(j)$ is the cost of choosing j to be on the primary path of demand r. The cost of j is set to 1 if there is enough available capacity on j to accommodate demand r. Otherwise, it is set to ∞.

For the links in the destination domain, the link cost is determined according to the following function:

$$W_p^{des}(n) = \begin{cases} 1 & b \le A_n^{des} \\ \infty & otherwise \end{cases}, \forall n \in [0,1,......N_{des} - 1] \tag{3.10}$$

N_{des} is the total number of links in the destination domain, and A_n^{des} is the available capacity on link n in the destination domain. $W_p^{des}(n)$ is the cost of choosing link n in the destination domain to be on the primary path of demand r.

The cost of link n is set to 1 if there is enough available capacity on link n to accommodate demand r. Otherwise, it is set to ∞.

The source node uses the source physical network and the virtual path network outside the source and destination domains to compute one optimum path from the source node to every ingress border node in the destination domain by using the Dijkstra's algorithm according to the cost functions (3.8) and (3.9). The least cost path is selected as the primary path from the source node to the selected border node in the destination domain. Next, the source node sends a message to the selected border node in the destination domain that contains the identity of the actual destination node, the bandwidth requested by the newly arrived demand, and the protection type of the demand. The border node will compute an optimum path from the border node to the destination node in the destination domain by using Dijkstra's algorithm according to the link cost function (3.10). Finally, if there exists a path from the border node to the destination node in the destination domain, the destination node will send a message back to the source via the border node to confirm the path setup. Otherwise, the demand will be rejected.

Step 2 (Backup Path Computation): if there exists a primary path for the demand, a backup path will be computed according to the scheme described below. For the links in the source domain, the link cost for backup path computation is determined according to the following function:

$$W_b^{src}(n) = \begin{cases} \infty & n \in S_p^{src}(r) \\ \varepsilon & T_n^{src,L} \le B_n^{src,L} \\ T_n^{src,L} - B_n^{src,L} & 0 < T_n^{src,L} - B_n^{src,L} \le A_n^{src} \\ \infty & otherwise \end{cases} \tag{3.11}$$

The term '*src*' denotes the source domain. $W_b^{src}(n)$ is the cost to choose link *n* in the source domain on the backup path. $S_p^{src}(r)$ is the set of links in the source domain that are on the primary path of demand *r*. $T_n^{src,L}$ is the maximum amount of backup bandwidth required on link *n* if a link in $S_p^{src}(r)$ fails. It follows that $T_n^{src,L}$ will simply be:

$$T_n^{src,L} = b + \max_{\forall n \in S_p^{src}(r)} [u_{mn}^{src}]. \tag{3.12}$$

$B_n^{src,L}$ is the total amount of backup bandwidth needed on link *n* in the source domain to restore the local traffic (as shown in equation (3.6)).

In equation (3.11), if link *n* is in $S_p^{src}(r)$, the cost of link *n* is set to ∞. Otherwise, the cost is set to a very small value ε (0 < ε << 1) if $T_n^{src,L}$ is less than or equal to $B_n^{src,L}$. In this case, demand *r* can be restored on link *n* without need to reserve any additional backup bandwidth on this link. If neither of the above conditions is satisfied, the cost is set to $T_n^{src,L} - B_n^{src,L}$ if this quantity is not larger than the available capacity on link *n*. In this case, $T_n^{src,L} - B_n^{src,L}$ is the amount of additional backup bandwidth required on link *n* in order to restore demand *r*. If the available capacity on link *n* is not adequate to accommodate this additional bandwidth the cost of link *n* is set to ∞ in the fourth term.

For inter-domain links and virtual paths outside of the source and destination domains, the link cost for backup path computation is determined according to the following function:

$$W_b(j) = \begin{cases} \infty & j \in S_p(r) \\ \infty & \forall d \in M_P(r) \& \forall i \in S_p(r) : i, j \in V(d) \& l_{ij}^d = 0 \\ \varepsilon & T_j \le B_j \\ T_j - B_j & 0 < T_j - B_j \le A_j \\ \infty & otherwise \end{cases} \tag{3.13}$$

$W_b(j)$ is the cost of choosing *j* to be on the backup path of demand *r*. $S_p(r)$ is the set of all virtual paths and inter-domain links that are on the primary path of demand *r*. $M_P(r)$ is the set of all intermediate domains of the primary path of demand *r*. $V(d)$ is the set of virtual paths in domain *d*. T_j is the maximum amount of backup bandwidth required on *j* if a virtual path or an inter-domain link in $S_p(r)$ fails:

$$T_j = b + \max_{\forall i \in S_p(r)} [k_{ij}] \tag{3.14}$$

B_j is the total amount of backup bandwidth needed on j to restore the transit traffic (as shown in equation (3.4)).

In equation (3.13), the cost of j is set to ∞ if it is already on the primary path. The cost of j is also set to ∞ if it is not link-disjointed with any of the virtual paths on the primary path. If neither of the above conditions is satisfied, the cost is set to a very small value (ε, $0 < \varepsilon < 1$) if T_j is not larger than B_j. In this case, demand r can be restored on j without need to reserve any additional backup bandwidth on j. Otherwise, the cost is set to $T_j - B_j$ if this quantity is less than or equal to the available capacity on j. In this case, $T_j - B_j$ is the amount of additional backup bandwidth required on j in order to restore demand r. If the available capacity on j is not adequate to accommodate this additional bandwidth, the cost of j is set to ∞ in the fifth term.

For the links in the destination domain, the link cost function is defined below in (3.15), which is similar to function (3.11) used for the links in the source domain:

$$W_b^{des}(n) = \begin{cases} \infty & n \in S_p^{des}(r) \\ \varepsilon & T_n^{des,L} \leq B_n^{des,L} \\ T_n^{des,L} - B_n^{des,L} & 0 < T_n^{des,L} - B_n^{des,L} \leq A_n^{des} \\ \infty & otherwise \end{cases} \tag{3.15}$$

The term '*des*' denotes the destination domain. $W_b^{des}(n)$ is the cost of choosing link n in the destination domain to be on the backup path. $S_p^{des}(r)$ is the set of links in the destination domain that are on the primary path of demand r. $T_n^{des,L}$ is the maximum amount of backup bandwidth required on link n if a link in $S_p^{des}(r)$ fails. It follows that $T_n^{des,L}$ will simply be:

$$T_n^{des,L} = b + \max_{\forall n \in S_p^{des}(r)} [u_{mn}^{desc}]. \tag{3.16}$$

$B_n^{des,L}$ is the total amount of backup bandwidth needed on link n in the destination domain to restore the local traffic (as shown in equation (3.6)).

The cost of link n is set to ∞ if link n is in $S_p^{des}(r)$. The cost is set to a very small value ($\varepsilon, 0 < \varepsilon < 1$) if $T_n^{des,L}$ is not larger than $B_n^{des,L}$. In this case, demand r can be restored on link n without need to reserve any additional backup bandwidth on this link. If neither of the above conditions is satisfied, the cost is set to $T_n^{des,L} - B_n^{des,L}$ if this quantity is less than or equal to the available capacity on link n. In this case, $T_n^{des,L}$ – $B_n^{des,L}$ is the amount of additional backup bandwidth required on link n in order to restore demand r. If the available capacity on link n is not adequate to accommodate this additional bandwidth the cost of link n is set to ∞ in the fourth term.

The source node uses the source physical network and the virtual path network outside the source and destination domains to compute an optimum backup path from the source node to every ingress border node in the destination domain by using the Dijkstra's algorithm according to the cost functions (3.11) and (3.13). The least-cost path is selected as the backup path from the source node to the border node of the destination domain. Next, the source node sends a message to the selected border node in the destination domain to request a computation of a backup path from the border node to the destination node in the destination domain. The border node will compute an optimum backup path from the border node to the destination node by using the Dijkstra's algorithm according to the link cost function (3.15). Finally, if there exists a backup path from the border node to the destination node in the destination domain, the border node will send a message back to the source to confirm the path setup. Otherwise, the demand will be rejected.

3.3.2 Dynamic Path Computation Algorithm in DDVP

DDVP scheme computes to a pair of domain-disjointed paths between two given nodes using the same four stages of processing as used in LDVP. We only describe the differences between LDVP and DDVP schemes below.

Because with DDVP, the primary and backup paths will be domain-disjointed, the information about the link-disjointedness relationship between virtual paths in a domain will not be used by the path computation authority. Therefore, domains do not need to store and advertise the content of matrix (3.2).

For the backup path computation, DDVP uses the cost function (3.17) described below for the inter-domain links and transit virtual paths, instead of the cost function (3.13) used in LDVP.

$$W_b(j) = \begin{cases} \infty & j \in S_p(r) \\ \infty & \forall d \in M_P(r) : j \in I(d) \\ \varepsilon & T_j \leq B_j \\ T_j - B_j & 0 < T_j - B_j \leq A_j \\ \infty & otherwise \end{cases} \tag{3.17}$$

The cost of inter-domain link or virtual path j is set to ∞ if it is already on the primary path of demand r. With the second term, the cost of j is also set to ∞ if it is an inter-domain link that emanates from a domain that is on the primary path of demand r. $I(d)$ indeed denotes the set of all inter-domain links emanating from domain d. With the third term, the cost is set to a very small value (ε, $0 < \varepsilon < 1$) if T_j is less than or equal to B_j. In this case, demand r can be restored on j without need to reserve any additional backup bandwidth on j. If neither of the above conditions is satisfied, the cost is set to $T_j - B_j$ if this quantity is not larger than the available capacity on j. In this case, $T_j - B_j$ is the amount of additional backup bandwidth required on j in order to restore demand r. If the available capacity on j is not adequate to accommodate this additional bandwidth the cost of j is set to ∞ in the fifth term.

3.3.3 Capacities Update

Once the paths are found for the newly arrived demand r, the total reserved primary bandwidth on links along the primary path and the total reserved backup bandwidth on links along the backup path are updated. The procedure for updating the reserved primary bandwidth is straightforward: the requested bandwidth b of demand r is simply added to the total primary bandwidth already reserved on each link along the primary path. For virtual paths on the primary path, the reserved bandwidth b should be added to each link on the virtual paths; For all the intra-domain and inter-domain links on the primary path, the bandwidth b is added to the total primary bandwidth already reserved on these links.

Let $S_l(j)$ denote the set of intra-domain links along the virtual path j; P_n^d denotes the total primary bandwidth already reserved on intra-domain link n in domain d; P_i denotes the total primary bandwidth already reserved on an inter-domain link i;

$$\forall j \in S_p(r) \,\&\, \forall n \in S_l(j): \; P_n^d \leftarrow P_n^d + b$$

$$\forall n \in S_p^{src}(r) \; : \; P_n^{src} \leftarrow P_n^{src} + b$$

$$\forall n \in S_p^{des}(r) \; : \; P_n^{des} \leftarrow P_n^{des} + b$$

$$\forall i \in S_p(r) : \; P_i \leftarrow P_i + b \qquad (3.18)$$

The procedure for updating the backup bandwidth is as follows. The total reserved backup bandwidth on every virtual path and inter-domain link along the backup path is updated via updating the corresponding elements in global matrix K in equation (3.3). Bandwidth b is added to element k_{ij} for every inter-domain link or virtual path i along the primary path and every inter-domain link or virtual path j along the backup path. That is:

$$\forall i \in S_p(r) \ \& \ \forall j \in S_b(r) : k_{ij} \leftarrow k_{ij} + b \qquad (3.19)$$

Where $S_p(r)$ is the set of inter-domain links and virtual paths on the primary path, which includes the set of all virtual paths that are overlapped with the links on the primary path in the source and destination domain, and the inter-domain links and virtual paths on the primary path outside of the source and destination domain. $S_b(r)$ is the set of inter-domain links and virtual paths on the backup path, which includes the set of all virtual path that are overlapped with the link on the backup path in the source and destination domain, and the inter-domain links and virtual paths on the backup path outside of the source and destination domain.

Once the elements in matrix K are updated, the new total reserved backup bandwidth on the backup inter-domain links and virtual paths in $S_b(r)$ can be obtained from equation (3.4). On some of these links or virtual paths, the new value of this quantity may be the same as the old value before demand r arrived. If j is such a link or a virtual path, T_j must have been less than or equal to B_j when the backup path was computed.

In the source and destination domain, the total amount of backup bandwidth needed on links should also be updated by updating the corresponding local matrix U_d in equation (3.1) of each domain.

$$\forall m \in S_p^{src}(r) \ \& \ \forall n \in S_b^{src}(r): u_{mn}^{src} \leftarrow u_{mn}^{src} + b$$

$$\forall m \in S_p^{des}(r) \ \& \ \forall n \in S_b^{des}(r): u_{mn}^{des} \leftarrow u_{mn}^{des} + b \tag{3.20}$$

Where *src* and *des* denote the source and destination domain. $S_p^d(r)$ is the set of links on the primary path in domain d, $S_b^d(r)$ is the set of links on the backup path in domain d. Once the elements in matrix U_{src} and U_{des} are updated, the new total reserved backup bandwidth on the links to restore the local traffic can be obtained from equation (3.6); The new total reserved backup bandwidth on the links within each domain to restore the transit traffic can be obtained from equation (3.5); The total amount of reserved backup bandwidth needed on link n on the backup path can be calculated form equations (3.7).

Chapter 4 LDVN Shared Restoration Scheme

Link Disjointed Virtual Node (LDVN) restoration is the third end-to-end shared restoration scheme we proposed in this thesis. LDVN is designed to find a pair of link-disjointed paths between any given pair of nodes located in different domains. As with LDVP and DDVP schemes, the network traffic with LDVN scheme is divided into two categories as well: local and transit (remote) traffic. Local traffic is the traffic that is exchanged between two nodes inside a single domain. The local traffic is routed over the links of the domain's physical topology (i.e. over intra-domain links). Transit traffic is the traffic exchanged between two nodes in different domains. This traffic may thus cross one or more intermediate domains before reaching the destination domain.

LDVN is designed and developed over virtual node abstraction network in which every domain is abstracted as a single virtual node with certain internal capacity that can be advertised to other domains. Figure 4.2 illustrates a virtual node network created from the original multi-domain network shown in Figure 4.1. In order to route the transit traffic, the multi-domain network is hence topologically reduced to a single-domain network, called virtual node network, which consists of virtual nodes interconnected by inter-domain links. Compared with the conventional least-cost path computation algorithms, the LDVN algorithm not only associates a cost to every (inter-domain) link, but also associates a cost to every virtual node. The link/node costs are used to seek for a path (or paths in case of protection) that accumulates the least cost through the network. The link/node costs are dynamically computed as a function of the available transmission capacity (in the case of a link), or as a function of the internal capacity (in the case of a node). Once a path is computed over the virtual node network, every domain along the path is then responsible for determining an explicit route over its intra-domain links for that path. The details, such as the mathematical network models and path computation algorithms, of LDVN scheme are described in the following subsections.

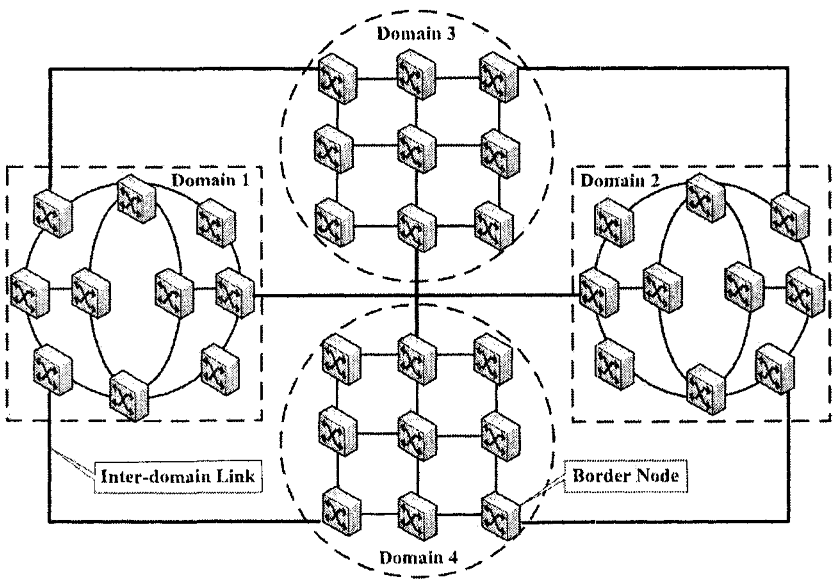


Figure 4.1 Multi-domain Network

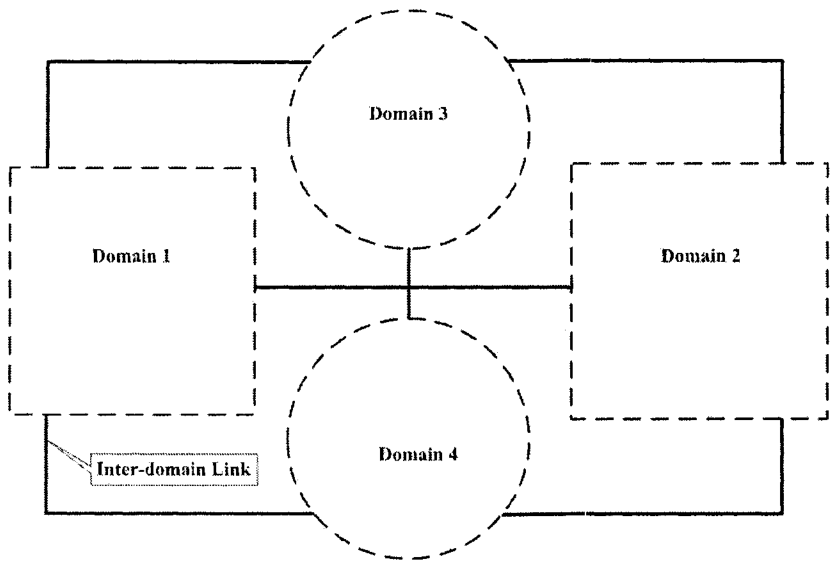


Figure 4.2 Virtual Node Network

4.1 Mathematical Network Models for LDVN

4.1.1 Virtual Node Network Model

With the LDVN scheme, no virtual path or pre-determined tunnel is used inside a domain to route the transit traffic. Instead, every domain is modeled as a single virtual node with certain internal capacity that can be advertised to other domains. The initial internal capacity of each virtual node is computed at the network configuration stage using the following procedure executed by every domain d in the multi-domain network.

Procedure 1 (Virtual Node's Internal Capacity Computation): between every pair of border-nodes (i, j) in domain d, a path with the maximum capacity is found. This is a path that traverses the links (inside domain d) with the highest available bandwidth. Although the problem of computing a path with the maximum capacity has been invetigated by many researchers, we used the Modified Dijkstra Algorithm presented in Appendix A to do so [23]. Now, Let us denote by $c_{ij}^{\max}$ the capacity of the maximum-capacity path between the border-nodes (i, j). We define the "internal capacity" of the virtual node d (denoted by C_d^I) to be the minimum of $c_{ij}^{\max}$'s over all border-node pairs (i, j) in domain d.

Domain d advertises C_d^I to all other domains, which they interpret as specifying the available capacity in domain d to carry the transit traffic. How this transit traffic will be carried over this domain is not relevant to other domains; it is strictly left to the discretion of domain d as to which path it will choose to carry the transit traffic across its domain. Note that **Procedure 1** must be executed every time that a connection (primary or backup) is computed and accepted for a new transit traffic in domain d. In case of any change to the current value of C_d^I, the new value must be advertised.

Once all the domains become aware of the internal capacity of each other, they can create the image of the virtual node network, which will consist of virtual nodes (each of which replaces the corresponding domain) interconnected by inter-domain links. Every domain will have the same image of the virtual node network, which will be used to compute a pair of link-disjointed paths between every pair of nodes in the multi-domain network. Figure 4.2 illustrates a virtual node network created from the original multi-domain network shown in Figure 4.1.

Corresponding to this virtual node network, there will be a global capacity matrix (denoted by K) that records the backup bandwidth reserved on each inter-domain link or inside each virtual node in the virtual node network. Matrix K is defined as below:

$$K = \begin{bmatrix} k_{11} & k_{12} & k_{13} & \dots & k_{1J} \\ k_{21} & k_{22} & k_{23} & \dots & k_{2J} \\ k_{31} & k_{32} & k_{33} & \dots & k_{3J} \\ \dots & & & & \\ k_{J1} & k_{J2} & k_{J3} & \dots & k_{JJ} \end{bmatrix} \tag{4.1}$$

Parameter J represents the total number of inter-domain links and virtual nodes in the virtual node network. Element k_{ij} is the amount of backup bandwidth needed on j if i fails. Both i and j are each an inter-domain link or a virtual node. Every domain maintains and synchronizes a copy of this matrix. The total amount of backup bandwidth (B_j) needed on j to protect the transit traffic is indeed the maximum of all elements in column j of matrix K:

$$B_j = \max_{\forall i \le J}[k_{ij}] \tag{4.2}$$

4.1.2 Intra-domain Network Model

Every domain d also keeps two internal capacity-related matrices to account for the reserved backup bandwidth on its intra-domain links: one matrix for the local traffic, and one matrix for the transit traffic. For the local traffic, the backup bandwidth reserved on the intra-domain links of domain d is recorded in a private matrix U_d shown below.

$$U_d = \begin{bmatrix} 0 & u_{12}^d & u_{13}^d & \dots & u_{1N_d}^d \\ u_{21}^d & 0 & u_{23}^d & \dots & u_{2N_d}^d \\ u_{31}^d & u_{32}^d & 0 & \dots & u_{3N_d}^d \\ \dots & & & & \\ u_{N_d1}^d & u_{N_d2}^d & u_{N_d3}^d & \dots & 0 \end{bmatrix} \tag{4.3}$$

$$B_j^{d,L} = \max_{\forall i \le N_d} [u_{ij}^d] \tag{4.4}$$

Element u_{ij}^d is the amount of backup bandwidth needed on j if i fails. Both i and j are intra-domain links inside domain d. N_d is the number of intra-domain links in domain d. The total amount of backup bandwidth needed on an intra-domain link j in domain d to protect the local traffic is denoted by $B_j^{d,L}$, which is the maximum of all elements in column j of matrix U_d (see equation (4.4)).

For the transit traffic, the backup bandwidth reserved on intra-domain links of domain d is recorded in the private matrix V_d shown in (4.5) below. Element v_{ij}^d is the amount of backup bandwidth needed on intra-domain link j in domain d if an inter-domain link or virtual node i fails. Therefore, each element of this matrix actually maps the backup bandwidth reserved at the virtual node network level (higher level) to the backup bandwidth needed at the intra-domain level (lower level).

$$V_d = \begin{bmatrix} v_{11}^d & v_{12}^d & v_{13}^d & \cdots & v_{1N_d}^d \\ v_{21}^d & v_{22}^d & v_{23}^d & \cdots & v_{2N_d}^d \\ v_{31}^d & v_{32}^d & v_{33}^d & \cdots & v_{3N_d}^d \\ \cdots & & & & \\ v_{J1}^d & v_{J2}^d & v_{J3}^d & \cdots & v_{JN_d}^d \end{bmatrix} \quad (4.5)$$

$$B_j^{d,T} = \max_{\forall i \le J} [v_{ij}^d] \quad (4.6)$$

The total amount of backup bandwidth needed on an intra-domain link j in domain d to protect the transit traffic is denoted by $B_j^{d,T}$, which is the maximum of all elements in column j of matrix V_d, see (4.6).

4.2 Dynamic Path Computation in LDVN

For a newly arrived demand r requesting b units of bandwidth, the source node uses a two-step algorithm described below to compute a pair of link-disjointed working and shared backup paths. Both paths will be computed at two different levels. First, at the higher level (level- 1), a least-cost path is computed between the source and destination virtual nodes over the virtual node network. This path consists of a consecutive sequence of virtual nodes interconnected by inter-domain links, starting from the source virtual node and ending at the destination virtual node. At the second level, each of the virtual nodes (domains) along the path will be responsible for determining an internal (intra-domain) path-segment between the two border-nodes of the level-1 path in that domain.

4.2.1 Step 1: Primary Path Computation

The level-1 primary path is computed by executing the Dijkstra's algorithm over the virtual node network with the following inter-domain link cost (weight) assignment:

$$W_P(l) = \begin{cases} \infty & b > A_l \\ \infty & \exists d : l \in I(d) \,\&\, b > C_d^I \\ 1 & otherwise \end{cases} \quad (4.7)$$

$W_P(l)$ is the cost of choosing an inter-domain link l to be on the path; A_l is the available capacity on l; d is a virtual node; C_d^I is the available internal capacity in d; and $I(d)$ is the set of all inter-domain links terminating on d. The first condition ensures that the inter-domain link l will not be selected if it does not have enough capacity to accommodate demand r. The second condition ensures that link l will not be selected if it terminates on a domain (virtual node) that does not have enough internal capacity to accommodate demand r. That is: the cost of inter-domain link l is set to ∞ , if there exists a virtual node d that terminates l (i.e. $l \in I(d)$) and the available internal capacity in d is less than the bandwidth requested by demand r $(b > C_d^l)$.

After computing the level-1 primary path, each virtual node along the computed path is responsible for mapping that path to an intra-domain path-segment inside its domain. For the intra-domain primary path computation within domain d, the cost of each intra-domain link is determined according to the following equation:

$$W_p^d(n) = \begin{cases} 1 & b \le A_n^d \\ \infty & otherwise \end{cases} \tag{4.8}$$

$W_p^d(n)$ is the cost of choosing the intra-domain link n in domain d to be on the primary path of demand r, and A_n^d is the available capacity on link n. The Dijkstra's algorithm is executed over the network-topology of domain d with the cost assignment (4.8) to determine the least-cost intra-domain path between the ingress and egress border-nodes of domain d that are on the level-1 path. Note that, in general, this intra-domain path may not coincide with the maximum capacity path found by **Procedure 1** between these ingress and egress border-nodes. In any case, regardless of whether the two paths overlap (partially, completely, or none), **Procedure 1** must be executed by domain d every time that a new intra-domain path-segment is established for a primary or backup connection.

4.2.2 Step 2: Backup Path Computation

The backup path computation follows the same two-level procedure as described for the primary path computation. However, the cost functions for the backup path computation are different from those used for the primary path computation. The link cost functions for the level-1 backup path computation are defined as follows:

$$W_b(l) = \begin{cases} \infty & l \in S_p(r) \\ \infty & \exists d : l \in I(d) \,\&\, (T_d - B_d) > C_d^I \\ \varepsilon & T_l < B_l \\ T_l - B_l & 0 < T_l - B_l < A_l \\ \infty & otherwise \end{cases} \tag{4.9}$$

$$T_j = b + \max_{\forall i \in S_p(r)} [k_{ij}] \tag{4.10}$$

$W_b(l)$ is the cost of choosing an inter-domain link l to be on the backup path for demand r; $S_p(r)$ is the set of all virtual nodes and inter-domain links along the primary path of demand r; C_d^I is the available internal capacity of virtual node d; $I(d)$ is the set of inter-domain links terminating on d; T_d / T_l is the maximum amount of backup bandwidth required on d / l if a virtual node or an inter-domain link in $S_p(r)$ fails. Both T_d and T_l are obtained from equation (4.10). B_d / B_l is the total backup bandwidth reserved on d / l, which is obtained from equation (4.2).

In (4.9), $W_b(l)$ is set to ∞ if the inter-domain link l is already on the primary path. $W_b(l)$ is also set to ∞ if l terminates on a virtual node d that does not have enough backup capacity to restore demand r (that is: $W_b(l)$ is ∞ , if there exists a virtual node d such that $l \in I(d)$ and $T_d - B_d > C_d^l$). With the third term, the cost $W_b(l)$ is set to a very small value ($\varepsilon, 0 < \varepsilon < 1$) if $T_l \leq B_l$. In this case, demand r can be restored on link l without need to reserve any additional backup bandwidth on this link. If neither of the above conditions is satisfied, the cost is set to $(T_l - B_l)$, which is the amount of additional backup bandwidth required on l in order to restore demand r, if this amount is available. Otherwise, $W_b(l)$ is set to ∞ in the fifth term.

After the level-1 backup path is computed over the virtual node network, every domain along that path is responsible for computing an intra-domain backup path-segment between the two border-nodes of the level-1 backup path in that domain. In general, a virtual node can be on both the primary path and the backup path of demand r. The reason for this is that, with LDVN scheme, the primary and backup paths of demands are required to be link-disjointed, not domain-disjointed or node-disjointed.

Therefore, if a virtual node d turns out to be on both the primary and backup paths of demand r, the link cost function used to compute the intra-domain backup path-segment inside domain d will be derived as follows:

$$W_b^d(n) = \begin{cases} \infty & n \in S_p^d(r) \\ \varepsilon & T_n^{d,L} \le B_n^{d,L} \\ T_n^{d,L} - B_n^{d,L} & 0 < T_n^{d,L} - B_n^{d,L} \le A_n^d \\ \infty & otherwise. \end{cases} \tag{4.11}$$

$$T_n^{d,L} = b + \max_{\forall m \in S_p^d(r)} [u_{mn}^d] \tag{4.12}$$

$W_b^d(n)$ is the cost of choosing intra-domain link n in domain d to be on the backup path of demand r, $S_p^d(r)$ is the set of intra-domain links in domain d that are along the primary path of demand r. This set was determined in Step 1 when the intra-domain primary path-segment for demand r was computed by domain d. A_n^d is the available capacity on n; $T_n^{d,L}$ is the maximum amount of backup bandwidth required on n if an intra-domain link in $S_p^d(r)$ fails (see (4.12)); u_{mn}^d is an element of matrix (4.3); and $B_n^{d,L}$ is obtained from (4.4). Condition 1 in (4.11) ensures that none of the intra-domain links in domain d that are on the demand r's primary path-segment will be selected as a backup link. With condition 2, the cost of link n is set to a very small value (ε, $0 < \varepsilon < 1$) if on this link demand r can be restored without need to reserve any additional backup bandwidth. With condition 3, the cost of link n is set to the amount of additional backup bandwidth required in order to restore demand r. Finally, if this amount is not available, the cost is set to ∞ in the forth term.

If, however, domain d is not on the primary path of demand r, the link cost function used to compute the intra-domain backup path inside domain d will be defined as follows:

$$W_b^d(n) = \begin{cases} \varepsilon & T_n^{d,T} \le B_n^{d,T} \\ T_n^{d,T} - B_n^{d,T} & 0 < T_n^{d,T} - B_n^{d,T} \le A_n^d \\ \infty & otherwise \end{cases} \quad (4.13)$$

$$T_n^{d,T} = b + \max_{\forall m \in S_P(r)} [v_{mn}^d] \quad (4.14)$$

$T_n^{d,T}$ is the maximum amount of backup bandwidth required on the intra-domain link n if an inter-domain link or a virtual node along the primary path of demand r fails (i.e. if any member of the set $S_p(r)$ fails). v_{mn}^d is an element of matrix (4.5); and $B_n^{d,T}$ is obtained from (4.6).

4.2.3 Capacities Update

The reserved capacities of each link along the paths are updated after both the primary and backup paths are computed for the newly arrived demand r. At the virtual node network level, the primary and backup paths traverse zero, one, or more intermediate virtual nodes (domains) interconnected by inter-domain links to reach the destination. Within each of these domains, there is a path-segment computed over intra-domain links. Hence, the reserved capacities are updated globally over the transit network and locally within each domain along the paths.

For the primary path, the capacity of every inter-domain link along the primary path is updated by adding the requested bandwidth b to the reserved primary capacity on that link. Each domain along the primary path adds the requested bandwidth b to the reserved primary capacity of every link along the primary path-segment.

$$\forall d \in S_{p,n}(r) \,\&\, \forall n \in S_p^d(r) : P_n^d \leftarrow P_n^d + b$$

$$\forall i \in S_p(r) : P_i \leftarrow P_i + b \tag{4.15}$$

$S_{p,n}(r)$ denotes the set of virtual nodes along the primary path of demand r; P_n^d denotes the total primary bandwidth already reserved on the intra-domain link n in domain d; P_i denotes the total primary bandwidth already reserved on an inter-domain link i.

Due to backup capacity sharing scheme, the backup capacity on each backup link is not updated directly by adding the requested bandwidth to the reserved backup capacity on that link. As shown in equations (4.1), (4.3) and (4.5), the global matrix K and the local matrices U_d and V_d of each domain d have been designed to record the reserved backup capacities on network links. The backup capacity on each link along a computed backup path is thus updated by updating the corresponding elements of the above matrices according to the following procedures.

The backup capacities in virtual nodes and on inter-domain links are updated by updating the elements in matrix (4.1).

$$\forall i \in S_p(r) \ \& \ \forall j \in S_b(r) : k_{ij} \leftarrow k_{ij} + b \tag{4.16}$$

$S_p(r)$ is the set of all virtual nodes and inter-domain links along the primary path of demand r. $S_b(r)$ is the set of all virtual nodes and inter-domain links along the backup path of demand r. For every i in set $S_p(r)$ and every j in set $S_b(r)$, element k_{ij} of the global matrix K is updated by adding the requested bandwidth b to it. After updating all corresponding elements of global matrix K, the reserved backup capacity B_j on each virtual node or inter-domain link j can be calculated from equation (4.2).

The backup capacities along the backup path-segments within domains are updated via updating the internal matrices U_d and V_d (see equation (4.3) and (4.5)) within each domain. Matrix U_d is used to record the backup capacities on intra-domain links to restore the local traffic. Matrix V_d is for the transit traffic. The selection of the internal matrices to be updated within domain d depends on whether d is on both of the primary and backup paths or not.

If a domain d is selected on both of the primary and backup path, the matrix for the local traffic, U_d, is updated. In this case, because domain d has the detailed information about the primary path-segment within itself when it computes the internal backup path-segment, the transit traffic is treated as local traffic by domain d when it traverses this domain. Therefore, the matrix for the local traffic, U_d, is updated when the intra-domain links' backup capacities are updated. For every domain d is on both the primary and backup paths, for every intra-domain link m along the primary path-segment and for every intra-domain link n along the backup path-segment within d, the element u_{mn}^{d} of matrix U_d is updated by adding the requested bandwidth b to it.

$$\forall d \in S_b(r) \cap S_p(r) \,\&\, \forall m \in S_p^d(r) \,\&\, \forall n \in S_b^d(r): u_{mn}^d \leftarrow u_{mn}^d + b \tag{4.17}$$

Where $S_p^d(r)$ and $S_b^d(r)$ denote the primary and backup path-segments within d. After all corresponding elements in matrix U_d are updated, $B_n^{d,L}$, the reserved backup capacity on each intra-domain link n inside domain d to recover the local traffic can be calculated according to equation (4.4). Furthermore, B_d (calculated form equation (4.2)) is the total amount of backup bandwidth needed on domain (virtual node) d to protect the transit traffic. In other words, the amount of bandwidth B_d needed on each intra-domain link along the backup path-segment within d. The amount of reserved backup bandwidth on intra-domain link n to protect demand r, the value of $B_n^{d,L}$, should be larger than or at least equal to the value of B_d. Hence, the value of $B_n^{d,L}$ for every link n in $S_b^d(r)$ is updated by taking the maximum value between $B_n^{d,L}$ and B_d (see equation (4.18)).

$$\forall d \in S_b(r) \cap S_p(r) \,\&\, \forall n \in S_b^d(r) \;: B_n^{d,L} = B_d \quad if \quad B_d > B_n^{d,L} \tag{4.18}$$

On the other hand, if a domain d is just on the backup path of demand r, for every link n along the backup path-segment in domain d, the reserved backup capacity for transit traffic ($B_n^{d,T}$) is updated by updating the corresponding elements in matrix V_d (see equation 4.19).

$$\forall d \in S_b(r) \,\&\, d \notin S_p(r) \,\&\, \forall i \in S_p(r) \;\&\; \forall n \in S_b^d(r):\; v_{in}^d \leftarrow v_{in}^d + b \tag{4.19}$$

For every domain d on the backup path but not on the primary path of demand r, for every virtual node or inter-domain link i in set $S_p(r)$ and for every intra-domain link n in set $S_b^d(r)$, the backup bandwidth needed on n if i fails (element v_{in}^d in matrix V_d) is updated by adding the requested bandwidth b to it.

Chapter 5 Simulation and Performance Evaluation

We have simulated the proposed LDVP, DDVP and LDVN schemes by using Python programming language on eclipse IDE. The simulation has been carried out over two multi-domain networks to evaluate and compare the proposed schemes in terms of capacity usage and restoration time to be described in the following sections. The first simulated multi-domain network is a multi-domain network based on the NSF (National Science Foundation) network. NSF is one of the representative North American backbone networks. The second simulated network is based on the European Optical Network (EON). The characteristics of the simulated networks, the performance evaluation metrics and the simulation results are presented in the following sections in this chapter.

5.1 Simulated Networks

The proposed end-to-end shared restoration schemes---LDVP, DDVP and LDVN schemes---have been simulated on two multi-domain networks: a multi-domain network based on the NSF network that is named NSF-based Network and a network based on the European Optical Network named EON-based Network in this thesis.

5.1.1 NSF-based Network

Figure 5.1 shows the topology of the multi-domain NSF-based network used in this thesis. Squares in Figure 5.1 represent domains. Each domain corresponds to a node (a major city in the U.S.) in the original NSF network. We modeled each domain as a metro network covering the entire city, with the metro network topology shown within the cloud in Figure 5.1. Open circles represent internal nodes inside each domain (metro network), while solid circles represent the border nodes that connect that domain to the neighboring domains.

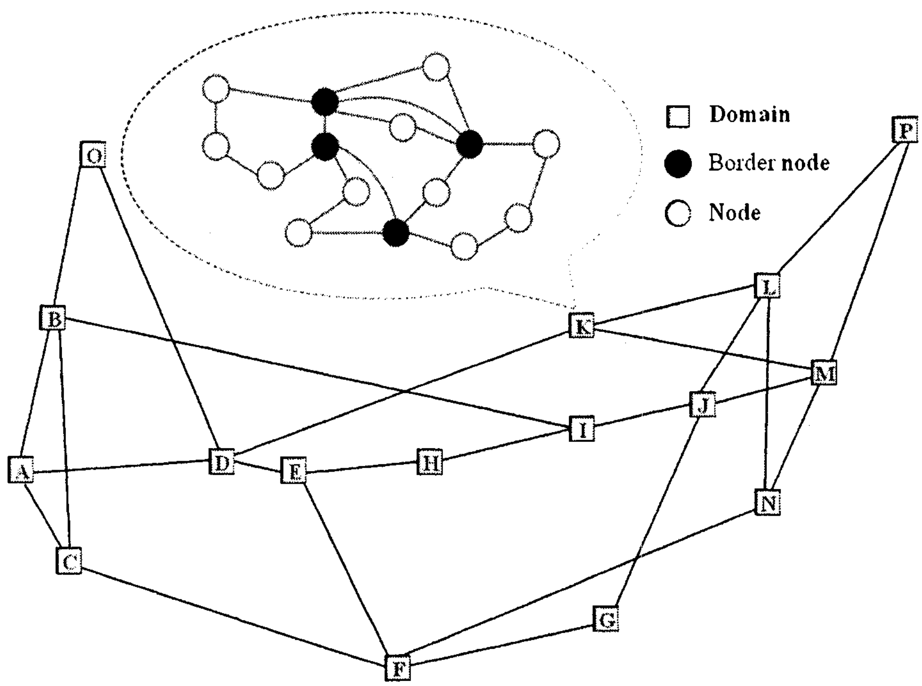


Figure 5.1 A Multi-domain Network Based on NSF Network

The NSF-based network contains 16 domains interconnected by 25 bidirectional inter-domain links. All domains have identical intra-domain network topology, shown within the cloud in Figure 5.1 and also reproduced in Figure 5.2. Every domain consists of 15 nodes interconnected by 20 intra-domain links. In each intra-domain network, the nodes with the highest degree are

Node	Node	Length (km)	Node	Node	Length (km)	Node	Node	Length (km)
A	B	750	D	K	3000	K	M	1200
A	C	750	E	F	750	J	L	750
A	D	1200	E	H	600	J	M	1200
B	C	1200	H	I	600	L	N	1200
B	O	750	F	G	1500	L	P	750
B	I	3000	F	N	3000	P	M	1200
C	F	1500	G	J	1050	M	N	600
O	D	1200	I	J	600	Average		1182
D	E	600	K	L	600			

Table 5.1 NSF Network Cable Length

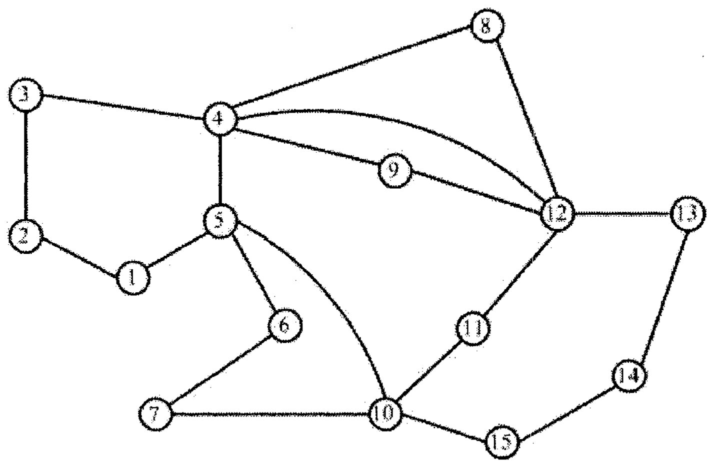


Figure 5.2 Intra-domain Network Topology in NSF-based Network

selected as the border nodes. The degree of a node is defined to be the number of links incident to that node. For instance, in the intra-domain network shown in Figure 5.2, the highest node's degree is five and the second highest degree is four. Each of the nodes 4 and 12 has a degree of five, whereas node 10 has a degree of four. The number of border nodes within each domain is equal to the degree of that domain. The degree of a domain is defined to be the number of inter-domain links incident to that domain. For example, nodes 4 and 12 will be the border nodes of a domain with degree two; nodes 4, 5, 10, and12 will be the border nodes of a domain with degree four.

5.1.2 EON-based Network

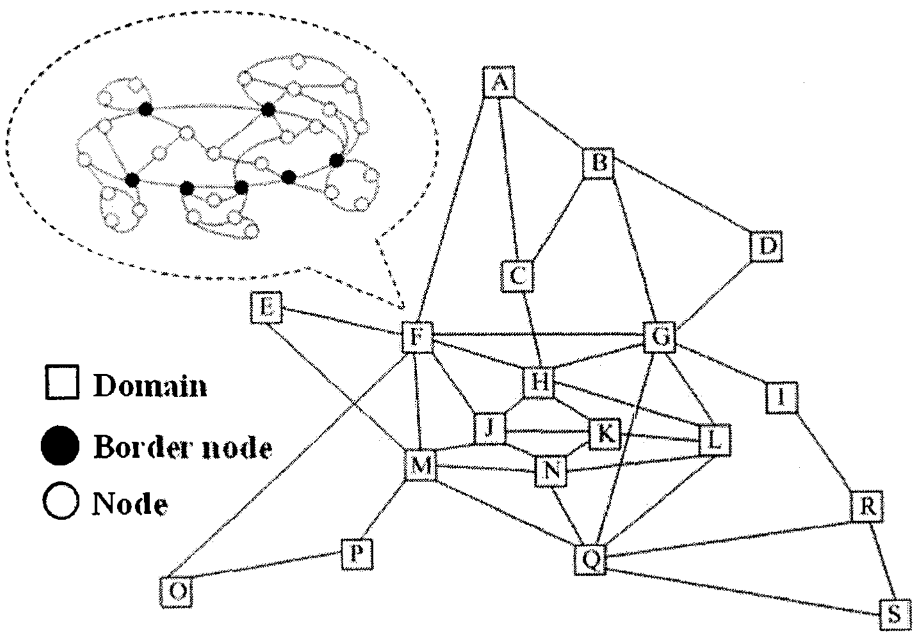


Figure 5.3 A Multi-domain Network Based on European Optical Network

The second simulated network is a multi-domain network based on the European Optical Network that is referred to as EON-based Network. As shown in Figure 5.3, EON-based Network contains 19 domains interconnected by 38 inter-domain links. Each domain in the EON-based Network represents a real network in a metropolitan area, consisting of 33 nodes interconnected by 52 links

(Figure 5.4) [29]. The squares in Figure 5.3 represent domains, the empty circles represent nodes within domains and solid circles represent the border nodes.

Node	Node	Length (km)	Node	Node	Length (km)	Node	Node	Length (km)
A	B	485	F	M	261	K	L	268
A	C	914	F	O	1709	K	N	276
A	F	1086	G	H	280	L	N	227
B	C	621	G	I	443	L	Q	517
B	D	668	G	L	489	M	N	966
B	G	633	G	Q	922	M	P	1053
C	H	413	H	J	754	M	Q	1106
D	G	515	H	K	524	N	Q	489
E	F	319	H	L	770	O	P	503
E	M	341	I	R	630	Q	R	894
F	G	721	J	K	682	Q	S	1052
F	H	652	J	M	439	R	S	523
F	J	492	J	N	547	**Average**		636.4

Table 5.2 European Optical Network Cable Length

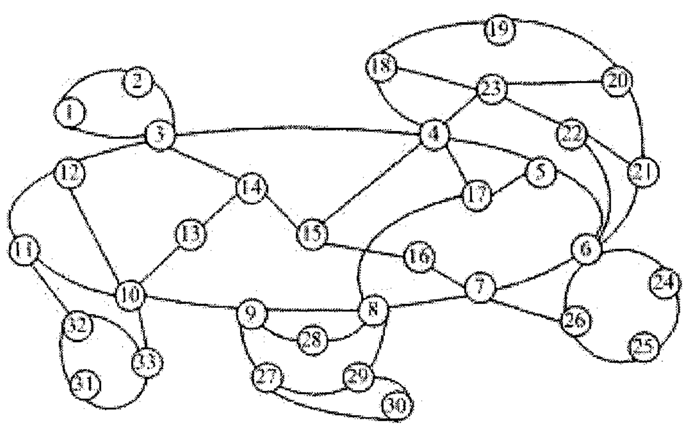


Figure 5.4 Intra-domain Network Topology in EON-based Network

5.2 Performance Evaluation Metrics

5.2.1 Capacity Usage

We used the following metrics to evaluate the capacity usage performance of the proposed schemes:

1) The average reserved primary capacity per inter-domain link, which is the sum of the reserved primary capacity on every inter-domain link divided by the number of inter-domain links. Let i denote an inter-domain link, N denote the number of inter-domain links in a multi-domain network, and P_i denote the total reserved capacity for all the primary connections traversing link i, the average reserved primary capacity per inter-domain link is $(\sum_{i=1\cdots N} P_i)/N$;

2) The average reserved primary capacity per intra-domain link, which is the sum of the average reserved primary capacity in every domain divided by the number of domains in the network. If P_n^d denotes the total primary bandwidth reserved on an intra-domain link n in domain d, and N_d is the number of intra-domain links in domain d, the average reserved primary capacity per intra-domain link in domain d will be $(\sum_{n=1\cdots N_d} P_n^d)/N_d$. The average reserved primary capacity per intra-domain link in the multi-domain network is the sum of $(\sum_{n=1\cdots N_d} P_n^d)/N_d$ over all domains divided by the number of domains;

3) The average reserved backup capacity per inter-domain link. If B_i denote

the reserved backup capacity on *i*, $(\sum_{i=1\cdots N} B_i)/N$ will be the average reserved backup capacity per inter-domain link;

4) The average reserved backup capacity per intra-domain link. If B_n^d denote the total backup bandwidth already reserved on intra-domain link *n* in domain *d*, the average reserved backup capacity per intra-domain link in domain *d* will be $(\sum_{n=1\cdots N_d} B_n^d)/N_d$. The average reserved backup capacity per intra-domain link is the sum of $(\sum_{n=1\cdots N_d} B_n^d)/N_d$ over all domains divided by the number of domains;

5) The average inter-domain link load. The average inter-domain link load is the sum of the load on every inter-domain link divided by the number of inter-domain links. The load on an inter-domain link is defined to be the sum of the reserved primary and backup capacities on the link divided by the total capacity of that link. Let ρ_i denote the load on link *i* and C_i denote the total capacity on *i*, then $\rho_i = (P_i + B_i) / C_i$ and the average inter-domain link load is $(\sum_{i=1\cdots N} \rho_i)/N$;

6) The average intra-domain link load, which is the sum of the average link load in every domain *d* divided by the number of domains. If C_n^d is the total capacity on an intra-domain link *n* in domain *d*, ρ_n^d is the load on *n*, then ρ_n^d is $(P_n^d + B_n^d)/C_n^d$ and the average link load in domain *d*

is $(\sum_{n=1\cdots N_d} \rho_n^d)/N_d$. The average intra-domain link load is the sum of $(\sum_{n=1\cdots N_d} \rho_n^d)/N_d$ over all domains divided by the number of domains;

7) The number of blocked demands in the network. The network will accept an arrived demand r if it can find a primary path (in case of no protection) or primary and backup paths (in case of protection) that the demand has requested. Otherwise, the demand will be blocked (or rejected). The number of blocked demands is the number of demands that are blocked by the network.

5.2.2 Restoration Time

When a link in the network fails, all the connections traversing the failed link are affected. The connections traversing the failed link are called failed connections in this thesis. For each failed connection requesting restoration services, the restoration time is the time taken from the instant a link fails to the instant the connection is rerouted over its predetermined backup path. In references [3] and [24], a formulation of restoration time has been presented for a single-domain network. In this thesis, we generalize that formulation to include restoration over a multi-domain network. The restoration time formula presented next is applicable to all of the three restoration schemes (LDVP, DDVP and LDVN) presented in this thesis.

It is desirable and indeed expected that the end-nodes of the failed link should detect the failure. Once an end-node of the failed link detects the failure, it can identify the identities of the failed connections in its information database. Figure 5.5 shows the steps to restore a failed connection for the proposed shared restoration schemes after a link failure is detected by the source end-node of the failed link. (It is assumed that the control network is reliable, i.e., control messages will never be lost; and the transmission time of control message can be neglected in comparison to the link propagation delay [3][25].) First, the source end-node of the failed link sends a *notification* message to notify the source node of the failed connection along the primary path. Because LDVP, DDVP and LDVN are shared restoration schemes, the backup paths are predetermined but the cross-connects along the backup path are not configured until the failure occurs. Hence, upon receiving the failure notification message, the source node of the failed connection sends a *request* (REQ) message along the backup path to the destination node in order to notify and configure the intermediate cross-connects along the backup path. The destination node will return an *acknowledgement* (ACK) message back to the source node after it receives the REQ message and finishes configuring its own cross-connect. Once the restoration procedure is completed, the traffic on the failed connection can be rerouted over the backup path.

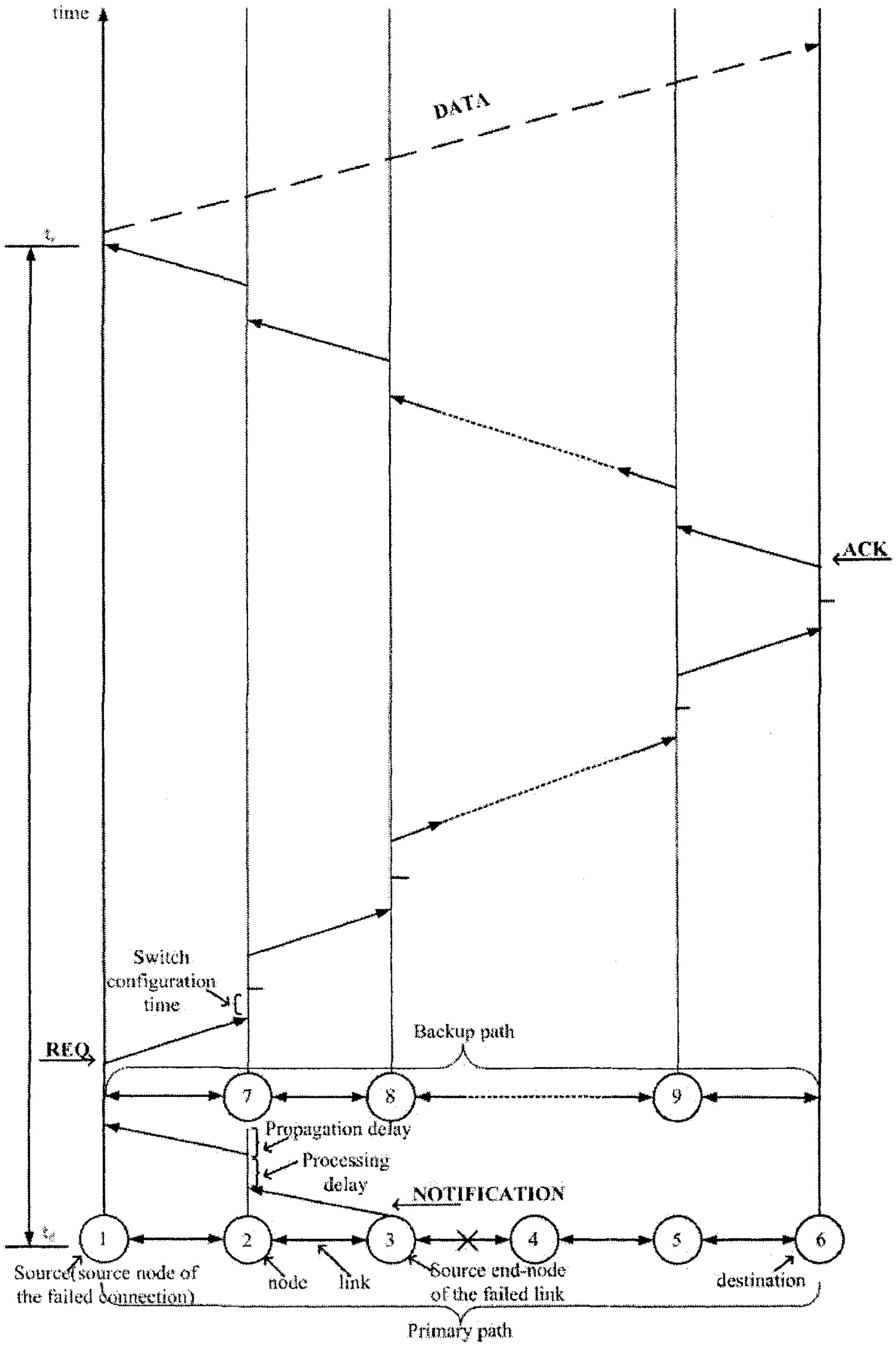


Figure 5.5 Restoration Process

We use the following notations to formulate the restoration time for the proposed restoration schemes in multi-domain networks.

d: Domain;

$t(r)$: Restoration time for a failed connection r;

t_d: Failure detection time;

$t_n(r)$: The time taken to notify the source-node of the failed connection r about the link failure event;

$t_s(r)$: The time taken to send a REQ message from the source-node of the failed connection r to the destination node and to configure the cross-connects along the backup path;

$t_a(r)$: The time taken to send an ACK message from the destination node to the source node of connection r;

P: Processing time at each node. (P is a fixed number. The case when a node receives many control message causes the increasing of the queuing delay is not considered in this analysis);

D_1: Propagation delay per inter-domain link. Propagation delay on a link is proportional to the length of the link;

D_2: Propagation delay per intra-domain link;

C: Cross-connect configuration time;

N: Number of inter-domain links from the source end of the failed link to the source end of the failed connection r;

S_D^p: Set of all domains from the source end of the failed link to the source end of the failed connection r except the domain in which the failed link is located;

n_d: Number of intra-domain links along the primary path of the failed connection r in each domain d in set S_D^p;

n_f: Number of intra-domain links along the primary path of the failed connection r from the source end of the failed link to the border node in the domain where the failure occurs;

M: Number of inter-domain links along the backup path of connection r;

S_D^b: Set of all domains along the backup path;

m_d: Number of intra-domain links along the backup path in each domain d in set S_D^b;

The restoration time for a failed connection r is:

$$t(r) = t_d + t_n(r) + t_s(r) + t_a(r) \tag{5-1}$$

where,

$$t_n(r) = N \times D_1 + n_f \times D_2 + (n_f + 1) \times P + \sum_{\forall d \in S_D^p} [n_d \times D_2 + (n_d + 1) \times P];$$

$$t_s(r) = M \times D_1 + \sum_{\forall d \in S_D^b} [(m_d + 1) \times C + m_d \times D_2 + (m_d + 1) \times P];$$

$$t_a(r) = M \times D_1 + \sum_{\forall d \in S_D^b} [m_d \times D_2 + (m_d + 1) \times P].$$

If $S_F(i)$ is the set of all failed connections that have requested restoration services when link i fails, the time taken to restore all these connections (or total restoration time) is:

$$t = t_d + \sum_{\forall r \in S_F(i)} [t_n(r) + t_s(r) + t_a(r)] \quad (5\text{-}2)$$

The average restoration time for each link is the total restoration time (5-2) divided by the number of failed connections of that link. The average restoration time for a network is the sum of average restoration time on every link divided by the total number of links in the network.

5.3 Simulation Results

We perform the simulation for every restoration scheme over two different network topologies described in section 5.1. The capacity and restoration time metrics of every restoration scheme are evaluated by using the evaluation metrics presented in section 5.2. Section 5.3.1 presents the parameters and simulation results on the NSF-based network. Section 5.3.2 presents the parameters and results on the EON-based Network.

5.3.1 NSF-based Network

We used two sets of link capacities to evaluate the capacity performance of the proposed schemes. In the first set of simulations (set 1), the capacities of inter-domain and intra-domain links are all set to 2.5 Gbps (OC-48), and the number of generated demands is varied from 50 to 350 at a step of 50. In the second set of simulations (set 2), the capacities of inter-domain links are set to 10

Gbps (OC-192), whereas the capacities of intra-domain links are set to 2.5 Gbps. The number of generated demands is varied from 50 to 600 at a step of 50. All demands request an identical amount of 100 Mbps bandwidth, which is the current fast Ethernet data rate. All demands request restoration services. Only the transit traffic is simulated in the experiments, which means that the source and destination nodes of a demand are always located in different domains. For each demand, the source and destination domains are generated uniformly randomly over all domains. The source node and the destination node inside the source and destination domains are also generated uniformly randomly over all the nodes in the source and destination domains.

With the NSF-based network, the inter-domain link length is set to the actual fiber length between the two end-nodes in the NSF network (see Table 5.1), and the intra-domain link length is set to 20 kilometers to fit the typical metropolitan networks where the diameter is around 100 kilometers [26]. The failure detection time t_d is 500 μs; the processing time P at each node is set to a fixed value 10 μs. The cross-connect configuration time C is either set to a low value of 10 μs or to a high value of 500 μs. These parameter sets are in line with the values reported in the literature, as well as with the expected range of values that can be achieved with today's technology [3] [24] [27] [28].

Figure 5.6 – Figure 5.10 show the simulation results with the link capacities in set 1 in the NSF network. Figure 5.6 illustrates the average reserved primary and backup capacities per inter-domain link, whereas Figure 5.7 shows the

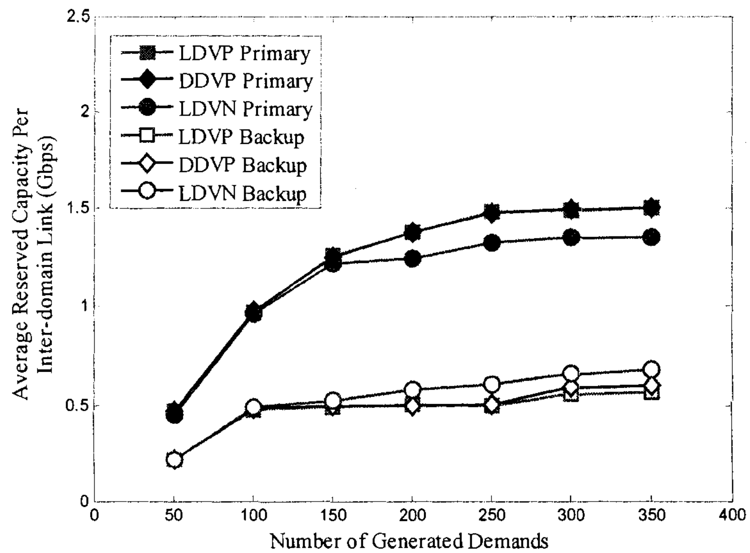


Figure 5.6 Average Reserved Capacity per Inter-domain Link

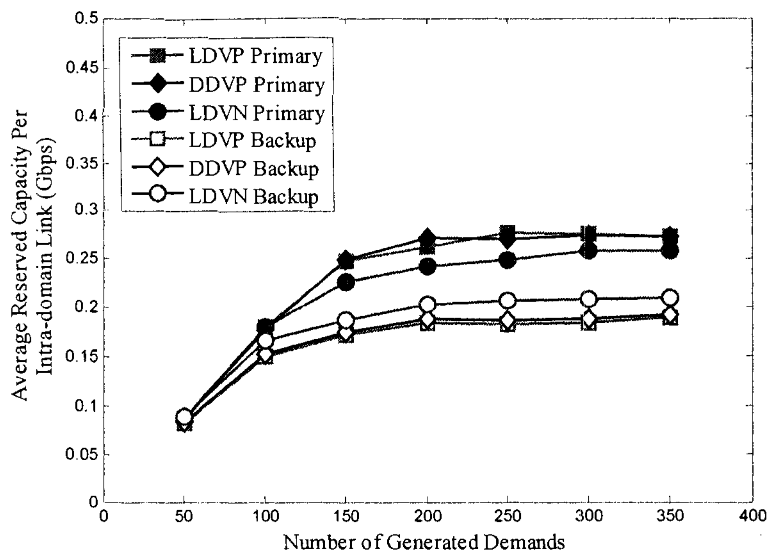


Figure 5.7 Average Reserved Capacity per Intra-domain Link

corresponding capacities per intra-domain link, each as a function of the number of generated demands. As can be seen from both figures, the average reserved primary capacity is always higher than the average reserved backup capacity with all the three schemes (LDVP, DDVP, and LDVN). This is due to the fact that the backup bandwidth is shared among different failure-disjoint primary paths, whereas the primary bandwidth is not.

In Figure 5.6 and Figure 5.7, we see that the curves for LDVP and DDVP schemes are very close to each other. We also see that the LDVN scheme always consumes less primary capacity on both inter-domain and intra-domain links than the LDVP and DDVP schemes. One reason for this could be that the LDVP and DDVP schemes accepted more number of demands than the LDVN scheme, for the same number of generated demands. Figure 5.8 indeed shows the number of blocked (rejected) demands for these schemes, as a function of the number generated demands in the network. However, the switchover occurs when the backup capacity on both link-types is considered: the LDVN scheme always

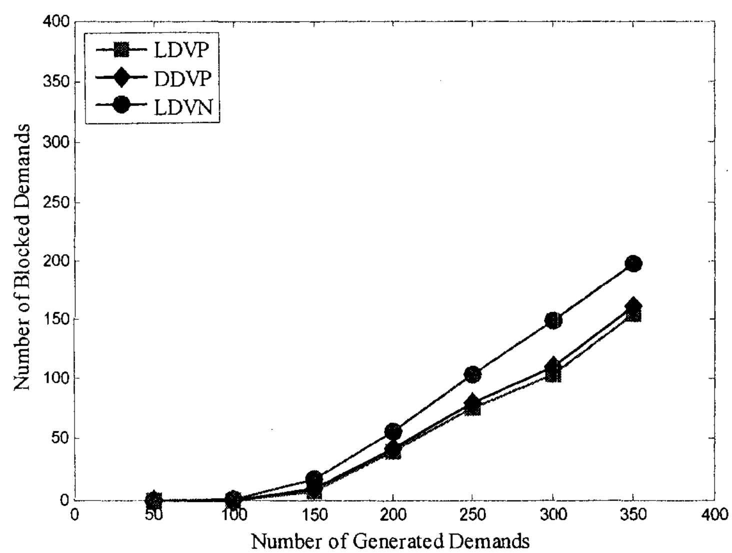


Figure 5.8 Number of Blocked Demands

consumes more backup capacity than the LDVP and DDVP schemes on both inter-domain and intra-domain links. One reason for this could be that with the LDVP and DDVP schemes, the backup path-segments for the transit traffic in the intermediate domains must follow the pre-established virtual paths in those domains. Whereas, with the LDVN scheme, there is no pre-established virtual path inside a domain, which provides more degree of freedom to LDVN to route backup path-segments over intra-domain links that have not been used (shared) by other connections before.

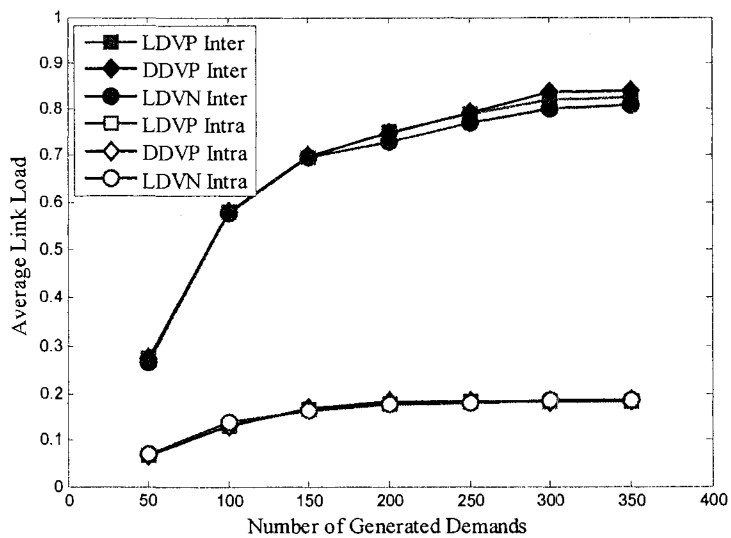


Figure 5.9 Average Link Load

Figure 5.9 illustrates the average link load as a function of the number of generated demands. The average load on inter-domain links is always higher than the average load on intra-domain links with all of the three restoration schemes.

In fact, the intra-domain links are not saturated, as the maximum load on these links does not exceed beyond 30%. One reason for this is that, in this set of simulation parameters, all inter-domain and intra-domain links have equal capacity, and all demands are transit. Therefore, with this setting, the inter-domain links will be the bottleneck, not allowing the intra-domain links to become saturated.

Figure 5.10 illustrates the average restoration time for each of the proposed restoration schemes as a function of the number of generated demands when the cross-connect configuration time is 10 μs and 500 μs. The figure indicates that

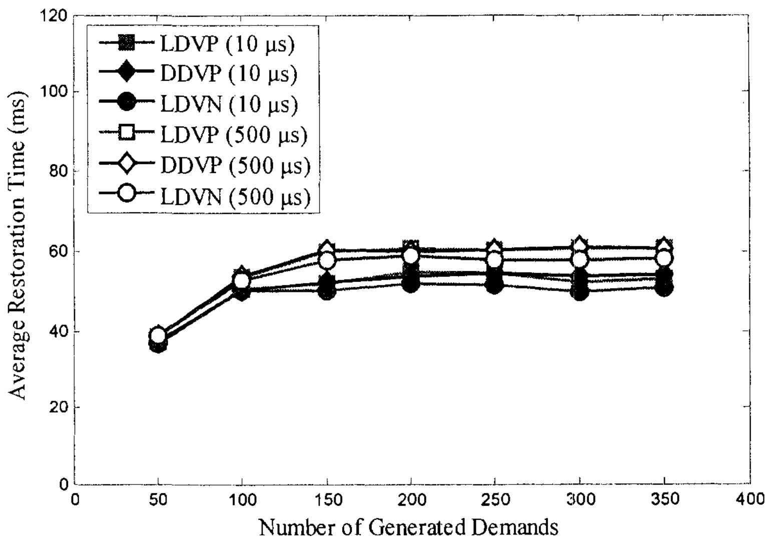


Figure 5.10 Average Restoration Time

the three restoration schemes yield virtually the same restoration time (on average) for the same number of generated demands and for the same value of configuration time. When the cross-connect configuration time is 500 μs, the average restoration time is slightly higher than the corresponding parameter when the configuration time is set to 10 μs. This is due to the fact that the time taken to configure all the cross-connects along the backup path is longer when the configuration time is at a higher value of 500 μs. Figure 5.10 shows that the restoration time initially increases as the number of generated demands changes from a low value, but eventually settles down to a value of around 60 ms, which is within the target service recovery in today's networks.

In the second set of simulations, the capacity of inter-domain links has been increased to 10 Gbps, while the capacity of intra-domain links is kept at 2.5 Gbps. Figure 5.11 – Figure 5.15 show the corresponding results. The results shown in Figure 5.11, Figure 5.12, Figure 5.13 and Figure 5.15 are very similar to those shown in Figure 5.6, Figure 5.7, Figure 5.8 and Figure 5.10. However, due to the increased inter-domain link capacity, the network accepts more demands and has a higher value of average reserved capacity and average restoration time than it does in the first set of simulation. Figure 5.14 shows that the average load on an intra-domain link is close to that on the inter-domain links, which is different than the results shown in Figure 5.9 where the average inter-domain link load is much higher than the average intra-domain link load. When the capacity on inter-domain links is increased to 10 Gbps, the inter-domain links will not be the bottleneck, allowing the intra-domain links to become saturated.

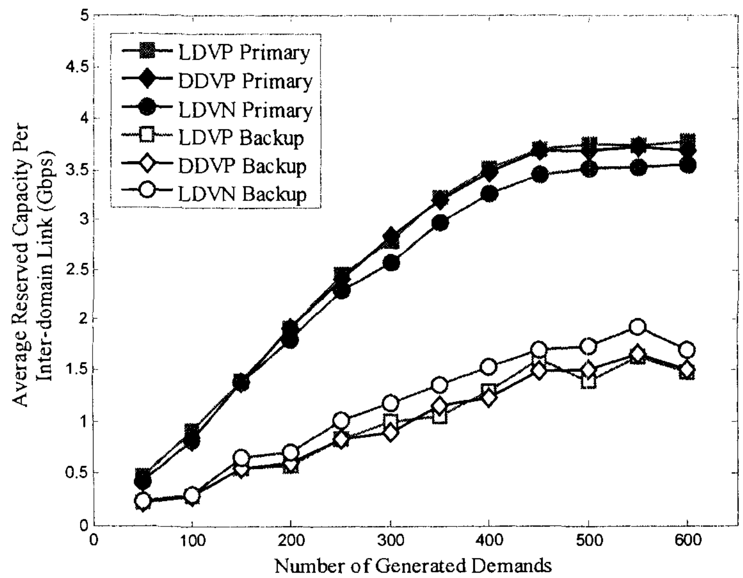


Figure 5.11 Average Reserved Capacity per Inter-domain Link

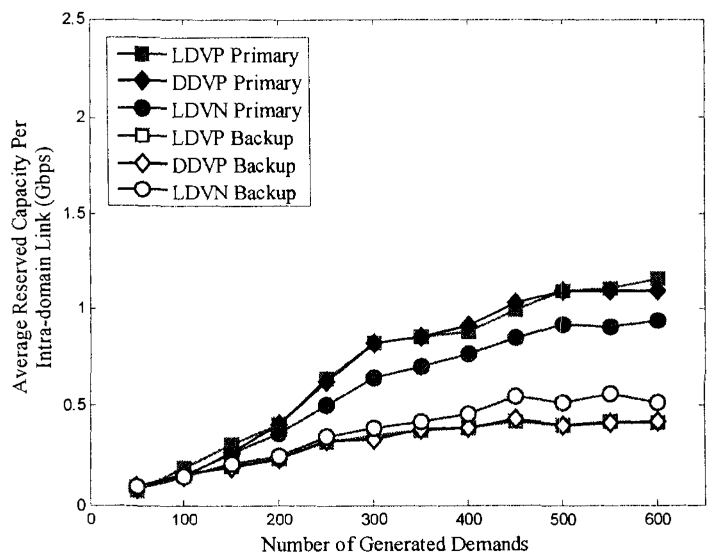


Figure 5.12 Average Reserved Capacity per Intra-domain Link

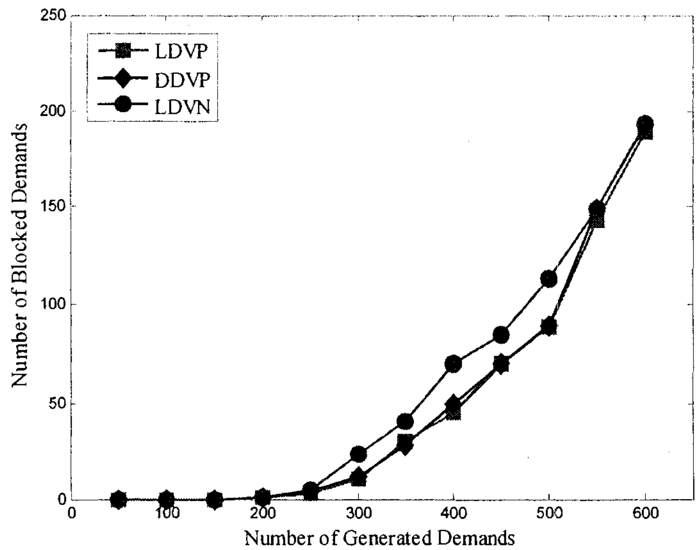


Figure 5.13 Number of Blocked Demands

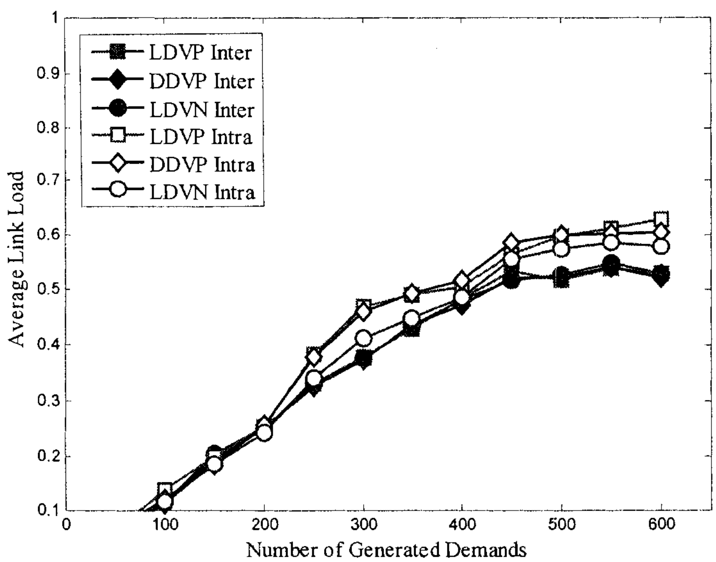


Figure 5.14 Average Link Load

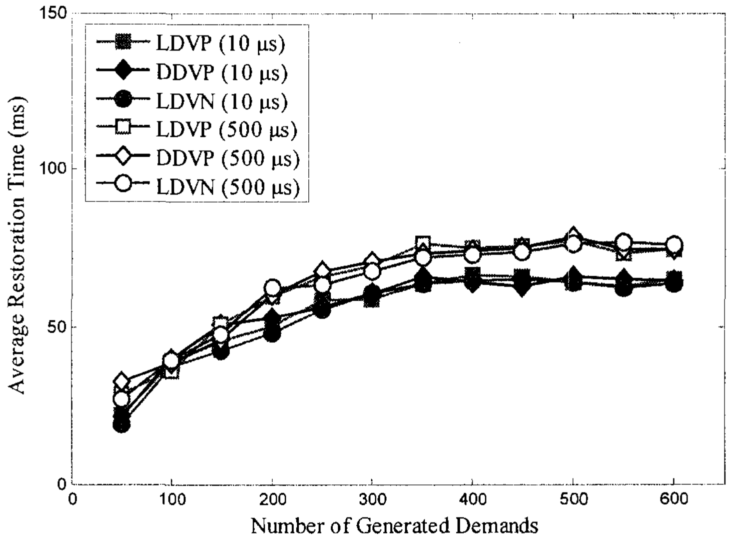


Figure 5.15 Average Restoration Time

5.3.2 EON-based Network

In the simulation experiments on the EON-based network, the parameters excluding the inter-domain link length, which is the real EON cable length shown in Table 5.2, have been set to the same values as those in the simulation on the NSF-based network. Figure 5.16 – Figure 5.20 illustrates the corresponding results for capacities Set 1 in which the capacities of inter-domain and intra-domain links are all set to 2.5 Gbps, and the number of generated demands is varied from 50 to 550 at a step of 100. Figure 5.21 – Figure 5.25 illustrates the corresponding results for the capacities Set 2 in which the capacities of inter-domain links are set to 10 Gbps, whereas the capacities of intra-domain links are set to 2.5 Gbps. In Set 2, the number of generated demands is varied from 50 to 1050 at a step of 100. Comparing with the corresponding Figures (Figure 5.6 – Figure 5.15) of NSF-based network, Figure 5.16 – Figure 5.25 shows the similar characteristics of the proposed restoration schemes. We have found that these restoration schemes have reached the similar performance in terms of capacity usage when the simulation has been performed on different simulated networks---NSF-based and EON-based networks. Meanwhile, as shown in Figure 5.20 and Figure 5.25, the restoration time achieved by the proposed restoration schemes over the EON-based network is around 40 ms which is shorter than the value (60 ms) over the NSF-based network. The reason for this could be the shorter propagation delay on inter-domain links in the EON-based network than it in the NSF-based network. (The propagation delay on a link is proportional to the length of the link. And The average EON cable length is 636.4 kilometers which is smaller than the average NSF network cable length, 1182 kilometers.)

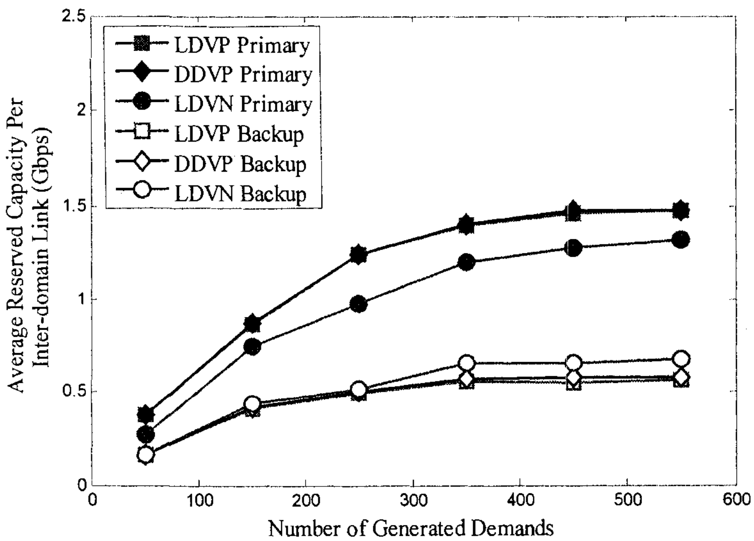


Figure 5.16 Average Reserved Capacity per Inter-domain Link

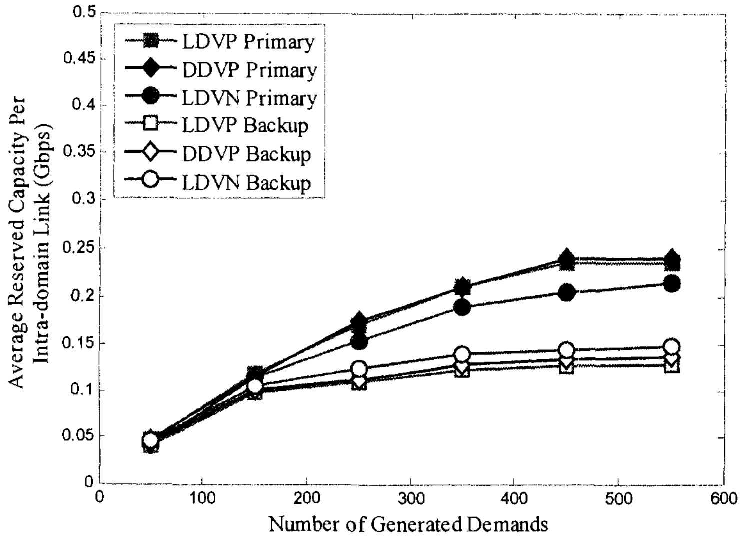


Figure 5.17 Average Reserved Capacity per Intra-domain Link

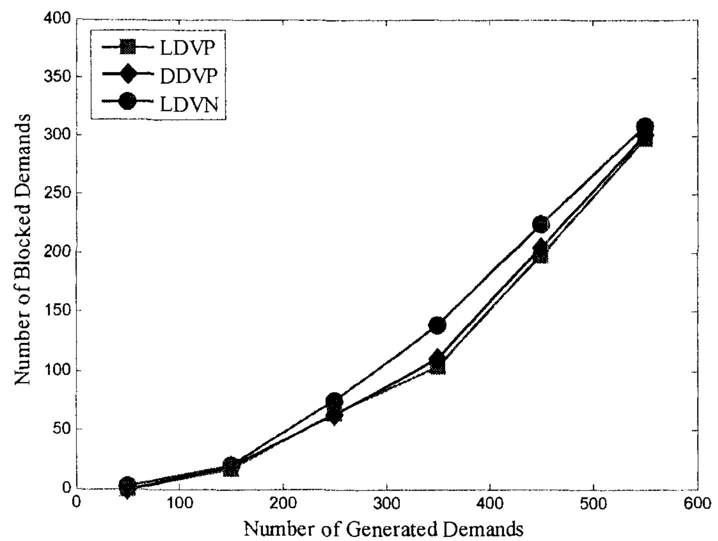


Figure 5.18 Number of Blocked Demands

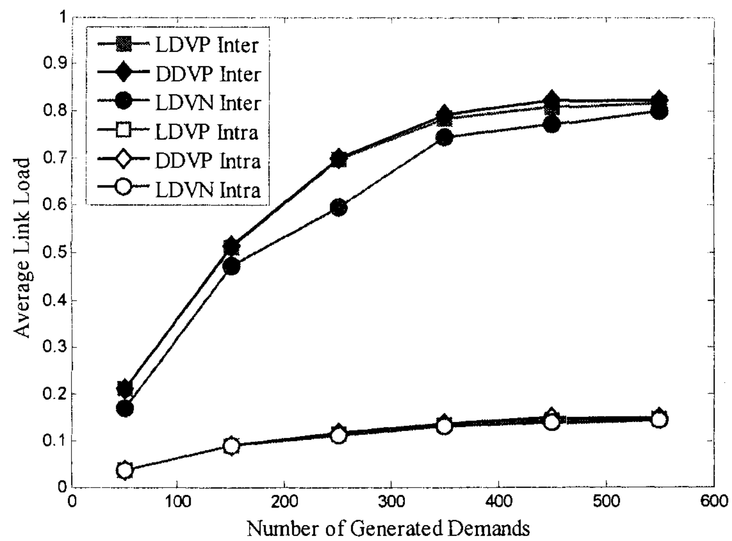


Figure 5.19 Average Link Load

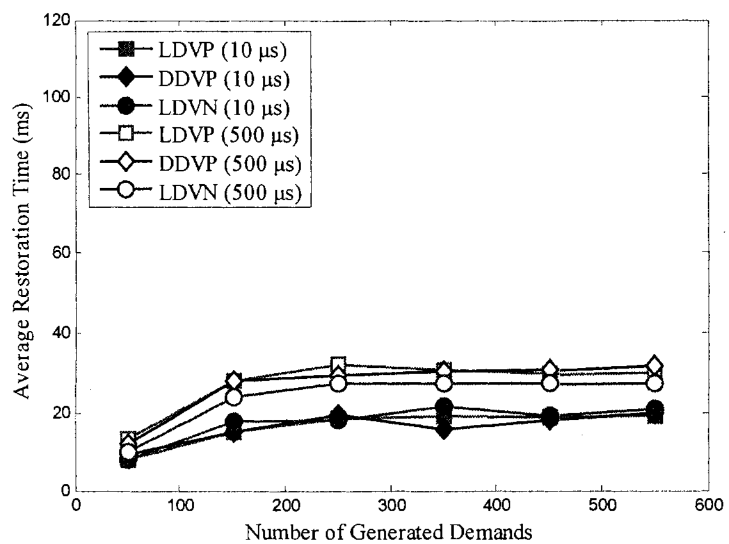


Figure 5.20 Average Restoration Time

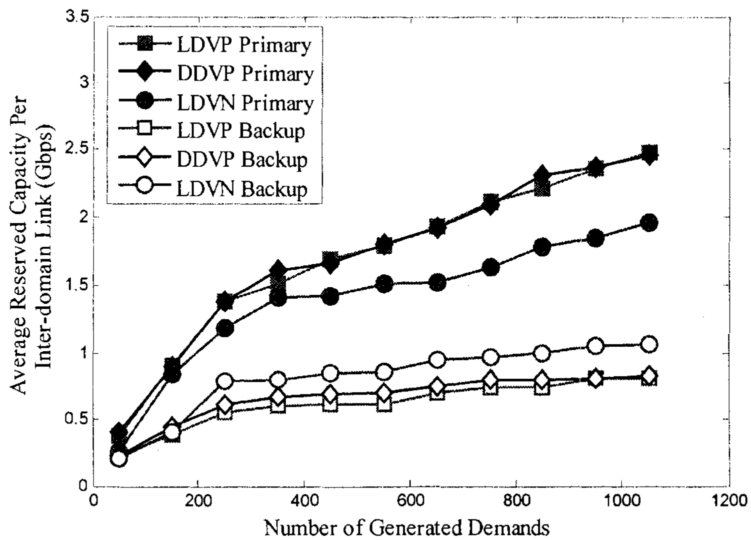


Figure 5.21 Average Reserved Capacity per Inter-domain Link

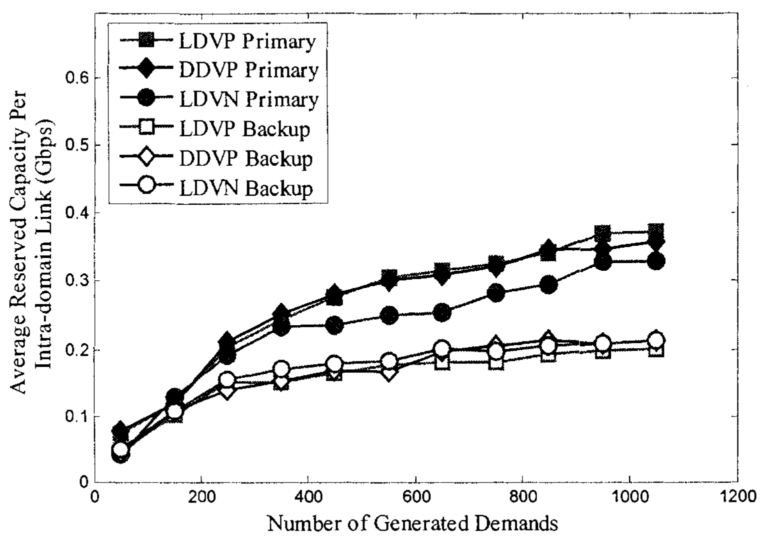


Figure 5.22 Average Reserved Capacity per Intra-domain Link

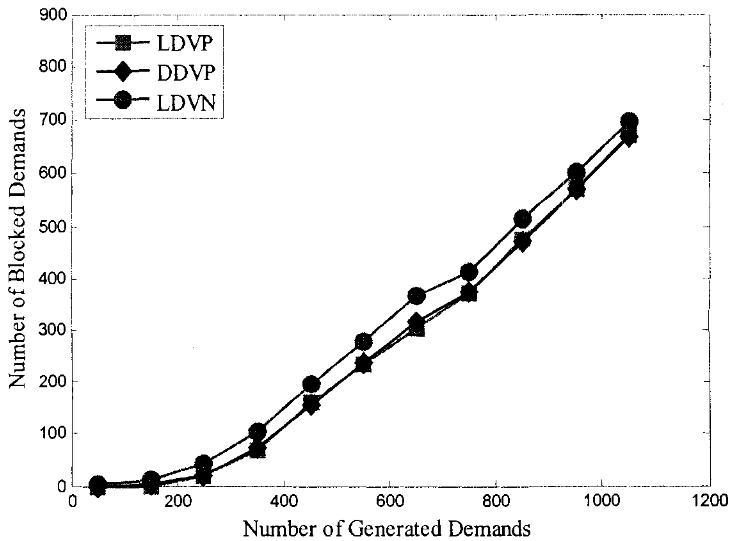


Figure 5.23 Number of Blocked Demands

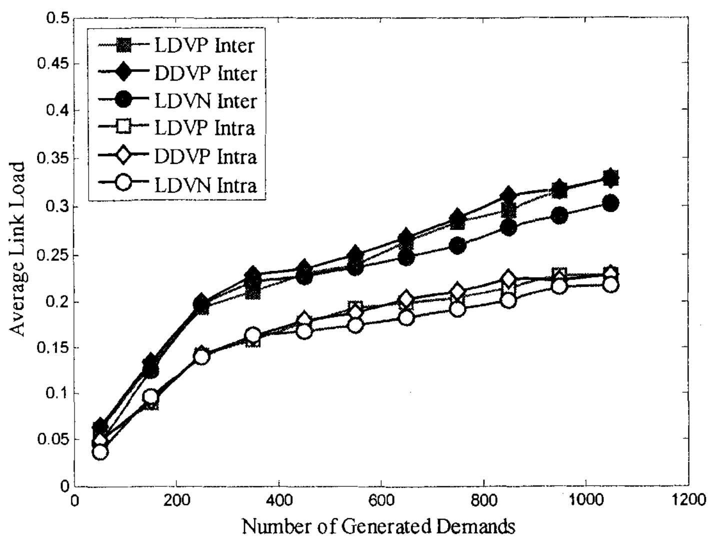


Figure 5.24 Average Link Load

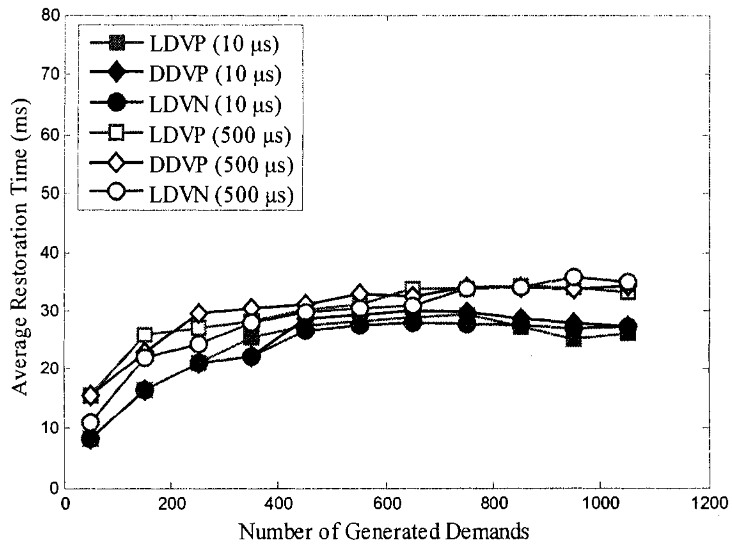


Figure 5.25 Average Restoration Time

Chapter 6 Conclusion and Future Work

6.1 Conclusion

We proposed three end-to-end capacity-constrained shared restoration schemes for computing failure-disjoint primary and backup paths between a pair of source and destination nodes in a multi-domain network environment. These schemes are referred to as Link Disjointed Virtual Path (LDVP), Domain Disjointed Virtual Path (DDVP) and Link Disjointed Virtual Node (LDVN) shared restoration schemes. With both LDVP and DDVP schemes, the multi-domain network is topologically aggregated to become a single-domain virtual path network, in which each domain is represented by its border nodes interconnected by point-to-point virtual paths. With LDVN scheme, each domain is abstracted as a virtual node. The multi-domain network is modeled as a single-domain virtual node network containing a set of virtual nodes interconnected by inter-domain links. For each demand, the end-to-end paths are computed on the aggregated multi-domain network. The primary path is dedicated to each connection, while the backup resources are shared among different failure-disjointed connections. The primary and backup paths are "link-disjointed" when they are computed by using LDVP or LDVN scheme, while they are domain-disjointed when DDVP scheme is used.

We have simulated the proposed restoration schemes on the NSF network and a multi-domain European network. We evaluated the performance of the proposed restoration schemes in terms of the capacity usage and the restoration time. We have found that the LDVP and DDVP schemes consume similar amount of capacity on average, but they have slightly more average reserved primary capacity and less average reserved backup capacity than the LDVN scheme. The average reserved primary capacity is always higher than the average reserved backup capacity with all the schemes. We also have found that these schemes yield similar amounts of time on average to restore a single-link failure in a multi-domain network. One advantage of the proposed schemes over the existing restoration schemes is that they require much less amount of link-state information to be advertised between the domains. This will reduce the routing message overhead and make the proposed algorithms to be scalable to large multi-domain networks.

6.2 Future Work

In this thesis, we proposed three capacity-constrained shared restoration schemes in multi-domain networks upon virtual path or virtual node abstraction network models. In the proposed schemes, the abstracted information within each domain is advertised frequently to other domains and every domain must maintain the same abstract image of the multi-domain network. The optimum level of the computed paths is affected by the advertising frequency and the level of information aggregation. We plan to enhance the routing mechanisms of the proposed schemes in the future by examining the reasonable trade-off between the advertising frequency and the level of link-state information aggregation for the different restoration schemes.

In addition, with the proposed shared restoration schemes, the backup resources are pre-calculated but allocated as soon as a failure occurs. Because the cross-connects along the pre-planned backup path are only configured upon the failure occurrence, the efficient signaling protocols are required to provide fast and effective restoration services. However, as illustrated in Chapter 5, we used the conventional straightforward restoration process when we evaluated restoration time for the proposed schemes. Therefore, another objective in our future works is to design new restoration signaling protocols unlike the conventional straightforward one to reduce the restoration time and improve the restoration efficiency.

References

[1] ITU-T Study Group 15, "Optical Transport Networks & Technologies Standardization Work Plan", http://www.itu.int/ITU-T/studygroups/com15/otn/index.html

[2] S. Ramamurthy, L. Sahasrabuddhe, and B. Mukherjee, "Survivable WDM Mesh Networks, Part I - Protection," Proc. of IEEE INFOCOM'99, Vol. 2, March 1999, pp. 744 - 751.

[3] S. Ramamurthy, L. Sahasrabuddhe, and B. Mukherjee, "Survivable WDM Mesh Networks, Part II - Restoration," Proc. of IEEE ICC'99, June 1999, pp. 2023- 2030.

[4] L. Sahasrabuddhe, S. Ramamurthy, and B. Mukherjee, "Fault Management in IP-Over-WDM Networks: WDM Protection Versus IP Restoration", IEEE Journal on Selected Areas in Communications, Vol. 20, Issue 1, January. 2002, pp 21-33.

[5] S. Yuan and J. P. Jue, "Dynamic Lightpath Protection in WDM Mesh Networks under Risk-Disjoint Constraint", Proc. of IEEE Globecom Conference, Vol. 3, pp 2019-2023, Nov. 29- Dec. 3, 2004, Dallas, Texas, USA.

[6] R. Iraschko, M. MacGregor, and W. D. Grover, "Optimal Capacity Placement for Path Restoration in STM or ATM Mesh-Survivable Networks", IEEE/ACM Transactions on Networking, Vol. 6, No. 3, June 1998, pp 325- 336.

[7] R. Ramamurthy, Z. Bogdanowicz, S. Samieian, D. Saha and B. Rajagopalan,"Capacity Performance of Dynamic Provisioning in Optical Networks", Journal of Lightwave Technology, Vol. 19, Issue.1, January 2001, pp. 40-48.

[8] M. Kodialam, T. V. Lakshman, "Dynamic Routing of Bandwidth Guaranteed Tunnels with Restoration", Proc. of IEEE INFOCOM' 2000, Vol. 2, March 2000, pp. 902-911.

[9] W. He, A. K. Somani, "Comparison of Protection Mechanisms: Capacity Efficiency and Recovery Time", Proc. of IEEE ICC'07, June, 2007, pp. 2218-2223, Glasgow, Scotland.

[10]W. Wen, S. J. Ben Yoo, B. Mukherjee, "Quality-of-Service Protection in MPLS Control WDM Mesh Networks", Photonic Network Communications, Vol. 4, July-December, 2002, pp. 297-320.

[11]H. Naser and H.T. Mouftah, "Enhanced Pool Sharing: A Constrained-based Routing Protocol for Shared Mesh Restoration Networks", OSA Journal of Optical Networking, Vol. 3, Issue.5, May 2004, pp. 303-323.

[12]D. Larrabeiti, R. Romeral, I. Soto, M. Uruena, T. Cinkler, J. Szigeti and J. Tapolcai, "Multi-domain Issues of Resilience", Proc. Of the 7th International Conference on Transparent Optical Networks, Vol. 1, July 2005, pp. 375–380.

[13]F. Hao, E. Zegura, "On scalable QoS routing: performance evaluation of topology aggregation", Proc. of IEEE INFOCOM'2000, Vol. 1, March 2000, pp. 147-156 .

[14]Y. Tang, S. Chen and Y. Ling, "State Aggregation of Large Network Domains", Computer Communications, Vol.30 No.4, February, 2007, pp.873-885.

[15]M. Yannuzzi, S. Sanchez Lopez, X. Masip Bruin, J. Sole-Pareta and J. Domingo Pascua , "A combined intra-domain and inter-domain QoS routing model for optical networks," Proc. of Conference on Optical Network Design and Modeling, Feburary 7-9, 2005, pp 197- 203, Milan, Italy.

[16]P. Torab, B. Jabbari, Q. Xu, S. Gong, X. Yang , T. Lehman, C. Tracy and J. Sobieski, "On Cooperative Inter-Domain Path Computation," Proc. Of IEEE Symposium on Computers and Communications (ISCC '06), June 26-29, 2006, pp 511- 518, Sardinia, Italy.

[17]H.A. Mantar, "A QoS routing architecture for multi-domain networks," Proc. of the Int'l Symposium on Computer Networks (ISCN 2006), pp 236- 241, June 16-18, 2006, Istanbul, Turkey.

[18]K. Loja, J. Szigeti and T. Cinkler, "Inter-domain routing in multiprovider optical networks: game theory and simulations," Proc. of the 1st Conference on Next Generation Internet Networks (NGI 2005), April 18-20, 2005, pp 157- 164, Rome, Italy.

[19]F. Aslam, Z. Afzal, A. Farrel, "Interdomain path computation: Challenges and Solutions for Label Switched Networks" IEEE Communications Magazine, Vol. 45, No. 10, October 2007, pp: 94 – 101.

[20]C. Huang and D. Messier, "A Fast and Scalable Inter-domain MPLS Protection Mechanism", Journal of Communications and Networks, Vol. 6, Issue. 1, March, 2004.

[21]A., Farkas, J. Szigeti and T. Cinkler, "P-cycle based protection schemes for multi-domain networks", Proc. of the 5th International Workshop on Design of Reliable Communication Networks (DRCN 2005), October 16- 19, 2005, Naples, Italy.

[22]H. Huang and J. A. Copeland, "Multi-domain Mesh Optical Network Protection using Hamiltonian cycles", Proc. of Workshop on High Performance Switching and routing, Merging Optical and IP Technologies, May 26-29, 2002, pp 83- 87, Kobe, Japan.

[23]S. Sahni, N. Rao, S. Ranka, Y. Li, E. Jung and N. Kamath, "Bandwidth Scheduling and Path Computation Algorithms for Connection-Oriented Networks", Proc. of the Sixth International Conference on Networking (ICN '07), April 22-28, 2007, pp 47-47, Sainte-Luce, Martinique.

[24]C. Assi, Y. Ye, A. Shami, S. Dixit and M. Ali, "Efficient Path Selection and Fast Restoration Algorithms for Shared Restorable Optical Networks", Proc. of IEEE International Conference on Communications, ICC '03, vol.2, pp. 1412- 1416, May 11-15, 2003. Alaska, USA.

[25]W. Stallings, "Data and Computer Communications", 7th edition, Pearson Education, Inc., Person Prentice Hall, Upper Saddle River, New Jersey.

[26]Sheeshia, S.; Yang Chen; Anand, V.; Chunming Qiao, "Performance comparison of OBS and SONET in metropolitan ring networks", IEEE Journal on Selected Areas in Communications, Vol. 22, Issue 8, October 2004, pp. 1474 - 1482.

[27]G.Sahin, S.Subramaniam, "Providing Quality-of-protection Classes Through Control-Message Scheduling in DWDM Mesh Networks with Capacity Sharing", IEEE Journal on Selected Areas in Communications, Vol. 22, Issue 9, November. 2004, pp. 1846 – 1858.

[28]Z.Yao, J. Fitchett, K. Felske, "A real-time approach for fast failure restoration in dynamic optical networks", Proc. of Design of Reliable Communication Networks (DRCN) 2003, pp. 131 – 138, October 19-22, 2003, Banff, Alberta, Canada.

[29]M. Herzberg, F. Shleifer, R. Ring and O Zolberg, "Applying OD Cycles to Multi-domain, Multi-service Survivable Networks", Proc. of 3rd EURO-NGI Conference on Next Generation Internet Networks, pp. 80-87, May 21-23 2007, Trondheim, Norway.

Appendix A: The Modified Dijkstra Algorithm

The widest path (the maximum-available-bandwidth path) computation is achieved by modifying the Dijkstra's shortest path algorithm. The following notation is used in presenting the modified Dijkstra algorithm: s is the source node; d is the destination node; $prev[i]$ is the previous node of node i along the maximum bandwidth path; $b[i][j]$ is the bandwidth available on the link connecting nodes i and j; $bw[i]$ is the maximum bandwidth along paths from the source s to i; and n is the total number of nodes. The modified Dijkstra's algorithm, for finding the shortest widest path from a specified source node and destination node, can be expressed in the following program:

MaxBandwidth(s,d,prev)

{

$bw[i] = b[s][i],\ 1 \le i \le n.$

$prev[i] = s,\ 1 \le i \le n.$

$prev[s] = 0.$

Initialize L to be a list with all nodes other than s.

for (i = 1, $i < n - 1$, i++)

{

Delete a node w from L with maximum bw.

if ($w == d$) return.

for (each u adjacent from w)

if ($bw[u] < \min\{bw[w], b[w][u]\}$)

```
            { bw[u] =min{bw[w], b[w][u]}.
            prev[u] = w.
            }
      }
}
```

The maximum bandwidth along paths from the source *s* to node *i* is assigned a value of 0 initially if node *i* is not adjacent to the source node. Otherwise, it is initialized as the bandwidth of the link connecting nodes *i* and *s*. As the final path to a node *i* is determined, it is assigned a value equal to the maximum available bandwidth between node *i* and *s*.